Б А К А Л А В Р И А Т

А.Г. Жихарев, О.А. Зимовец, М.Ф. Тубольцев, А.А. Кондратенко

ТЕОРИЯ СИСТЕМ И СИСТЕМНЫЙ АНАЛИЗ

Под редакцией **С.И. Маторина**

Рекомендовано
Экспертным советом УМО в системе
ВО и СПО в качестве **учебника**
для направлений бакалавриата
«Прикладная информатика», «Бизнес-информатика»,
«Информационные системы и технологии»,
«Системный анализ и управление»

BOOK.ru
ЭЛЕКТРОННО-БИБЛИОТЕЧНАЯ СИСТЕМА
КНОРУС • МОСКВА • 2021

УДК 001.51:519.876.5:004.94(075.8)
ББК 32.81:22.16я73
С34

Рецензенты:
А.Б. Петровский, главный научный сотрудник, руководитель отдела теории принятия решений Федерального исследовательского центра «Информатика и управление» РАН, профессор Волгоградского государственного технического университета, профессор Белгородского государственного технологического университета им. В.Г. Шухова, д-р техн. наук, проф.,
А.В. Заболеева-Зотова, начальник управления Российского фонда фундаментальных исследований, профессор Московского физико-технического института (национального исследовательского университета), д-р техн. наук, проф.

С34 **Теория систем и системный анализ** : учебник / А.Г. Жихарев, О.А. Зимовец, М.Ф. Тубольцев, А.А. Кондратенко ; под ред. С.И. Маторина. — Москва : КНОРУС, 2020. — 456 с. — (Бакалавриат)

ISBN 978-5-406-07767-2

В учебнике рассмотрены системные исследования как самостоятельное научное направление. Выделены основные виды системного подхода. Предложен оригинальный системно-объектный подход, позволяющий построить новую теорию систем. Изложены основные положения и функциональные возможности новой теории систем, позволяющей получать новые нетривиальные результаты. Дано формальное описание системы в терминах «Узел–Функция–Объект». Представлены известные и новые системно-объектные средства графоаналитического моделирования и анализа сложных систем. Показаны примеры практического применения нового инструментария.

Для студентов и аспирантов университетов, обучающихся по направлениям «Системный анализ и управление», «Прикладная информатика», «Бизнес-информатика», «Информационные системы и технологии». Книга будет также полезна консультантам по принятию решений, бизнес-аналитикам, ИТ-специалистам.

Соответствует ФГОС ВО последнего поколения.

Для бизнес-аналитиков, ИТ-специалистов, студентов университетов, обучающихся по направлениям «Прикладная информатика», «Бизнес-информатика», «Информационные системы и технологии», «Системный анализ и управление».

УДК 001.51:519.876.5:004.94(075.8)
ББК 32.81:22.16я73

Маторин Сергей Игоревич, Жихарев Александр Геннадиевич
Зимовец Ольга Анатольевна, Тубольцев Михаил Федорович
Кондратенко Анна Алексеевна

ТЕОРИЯ СИСТЕМ И СИСТЕМНЫЙ АНАЛИЗ

Изд. № 544820. Подписано в печать 19.09.2020. Формат 60×90/16.
Гарнитура «Times». Усл. печ. л. 28,5. Уч.-изд. л. 25,37. Тираж 500 экз.

ООО «Издательство «КноРус».
117218, г. Москва, ул. Кедрова, д. 14, корп. 2.
Тел.: +7 (495) 741-46-28.
E-mail: welcome@knorus.ru www.knorus.ru

Отпечатано в АО «Т8 Издательские Технологии».
109316, г. Москва, Волгоградский проспект, д. 42, корп. 5.
Тел.: +7 (495) 221-89-80.

ISBN 978-5-406-07767-2

АВТОРСКИЙ КОЛЛЕКТИВ

Маторин Сергей Игоревич – д-р техн. наук, профессор, заместитель генерального директора по науке и инновациям ЗАО «СофтКоннект» (г. Белгород), профессор кафедры информационных и робототехнических систем НИУ «Белгородский государственный университет», профессор кафедры информационных систем и технологий Белгородского университета кооперации, экономики и права.

Жихарев Александр Геннадиевич – канд. техн. наук, доцент, доцент кафедры информационных и робототехнических систем НИУ «Белгородский государственный университет».

Зимовец Ольга Анатольевна – канд. техн. наук, доцент кафедры информационных и робототехнических систем НИУ «Белгородский государственный университет».

Тубольцев Михаил Федорович – канд. техн. наук, доцент, ведущий научный сотрудник ЗАО «СофтКоннект (г. Белгород).

Кондратенко Анна Алексеевна – канд. техн. наук, специалист отдела бизнес-консультирования в ООО «Мотивэа» (г. Белгород).

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Во второй половине XX века в обиход учёных, инженеров, аналитиков прочно вошли такие понятия как «системный подход», «теория систем», «системный анализ», ядром которых является понятие «система». Интенсивно проводимые в первое время системные исследования сменились этапом осмысления полученных результатов. Основными причинами стали отличия в объяснении разными исследователями свойства системности вплоть до утверждения об отсутствии у понятия «система» онтологического статуса. Отсутствие общепринятого понимания и четкого определения системы привело к торможению системных исследований, которые в настоящее время свелись, в основном, к переписыванию полученных в прошлом результатов. Создание же конструктивной, а не описательной теории систем отложено на неопределённое время.

Основная идея книги – показать общую картину системных исследований, провести анализ накопившихся проблем, затрудняющих получение новых результатов; представить новый системно-объектный подход, позволяющий получить нетривиальные результаты и заложить основы конструктивной теории систем; описать известные и новые системно-объектные средства моделирования и анализа сложных систем; дать примеры практического использования теоретических результатов. Книга состоит из трех частей.

В первой части «Системный подход» описана суть системных исследований как самостоятельного научного направления. Выделены основные виды системного подхода. Предложен оригинальный системно-объектный подход, позволяющий построить новую теорию систем.

Во второй части «Теория систем» рассмотрены начала новой теории систем. Показаны функциональные возможности теории, позволяющие учитывать общесистемные принципы и закономерности, объяснять сложные явления природы и общества. Изложены подходы к формализации и моделированию систем. Представлены примеры использования нового теоретического инструментария для представления знаний, моделирования административных процедур, финансовых систем, природных и общественных процессов.

В третьей части «Технологии системного анализа» представлены известные и новые средства графоаналитического моделирования и анализа систем. Описаны хорошо известные средства структурно-функционального моделирования: диаграммы потоков данных DFD, методология компьютерного моделирования в стандартах IDEF0 и

IDEF3. Описаны относительно новые средства объектно-ориентированного моделирования: унифицированный язык моделирования UML, система моделирования и обозначения бизнес-процессов BPMN. Подробно рассмотрены оригинальные графоаналитические средства структурного и имитационного системно-объектного моделирования: программный инструментарий для моделирования в терминах «Узел-Функция-Объект», соответствующие Интернет-ресурсы.

Особенностью книги является системное изложение материала, что позволяет сформировать у читателя целостное представление о системных исследованиях с учётом особенностей всех частей этого научного направления. Читатель имеет возможность полноценно освоить графоаналитические средства системного анализа, широко используемые, например, при организационном проектировании и реинжиниринге бизнес-процессов, проектировании информационных систем. Текст книги, кроме разделов, связанных с формализацией, написан достаточно простым языком, доступным для читателей с различными профессиональными интересами и уровнями подготовки.

Представленный в учебнике материал прошел апробацию при чтении курсов лекций по дисциплинам «Теория систем и системный анализ», «Системный подход и системное моделирование», «Современные проблемы системного анализа и управления» в бакалавриате, магистратуре и аспирантуре по направлениям подготовки «Системный анализ, управление и обработка информации», «Прикладная информатика», «Бизнес-информатика», «Информационные системы и технологии». Системно-объектный подход, теоретический и программный инструментарий были использованы авторами при выполнении бизнес-проектов, диссертационных исследований, научных работ, поддержанных грантами Российского фонда фундаментальных исследований (18-07-00310, 18-07-00355, 19-07-00111, 19-07-00290).

Новый системно-объектный подход зародился под влиянием личного общения некоторых авторов с выдающимися отечественными учёными – Г.П. Мельниковым и Ю.А. Шрейдером. Содержащиеся в книге материалы являются результатами многолетней исследовательской и преподавательской деятельности членов авторского коллектива в области системных исследований. Авторы выражают свою благодарность Российскому фонду фундаментальных исследований, поддержавшему значительную часть научных работ грантами. Авторы признательны профессору А.Б. Петровскому и профессору Е.Г. Жилякову за поддержку работ в научной и образовательной сферах, без которой бы этот учебник не появился.

I. СИСТЕМНЫЙ ПОДХОД

1. Системные исследования (системный подход, теория систем, системный анализ)

Системные исследования представляют собой совокупность теоретических построений, концепций и методов, в которых объект исследования рассматривается как система.

Очевидно, что в этом определении и, таким образом, во всех системных исследованиях ключевым понятием является понятие «*система*». Оно, естественно, будет подробно рассмотрено в дальнейшем. В данном разделе рассмотрим особенности системных исследований в целом, их составные части и структуру.

Как относительно молодое научное направление «системные исследования» окончательно еще не сформировалось. В связи с этим разные представители научного сообщества (и связанные, и не связанные с этими исследованиями) воспринимают и оценивают их по-разному. От полного неприятия системности как явления действительности до включения в круг системных исследований научных теорий ничего общего с ними не имеющих. С точки зрения авторов это связано с тем, что в современном обществе до конца еще не сформировано системное мышление. Данное обстоятельство обуславливает необходимость целенаправленного обучения системному подходу и системному мышлению.

Опустим историю возникновения системных исследований, так как этот вопрос многократно освещен в различных публикация, включая Интернет (например, см. Википедию). Отметим только, что общие представления о системности в человеческом обществе и сознании возникли очень давно в рамках размышлений о сущности вещей и явлений. Например, еще Будда учил своих учеников пониманию сущности вещей, предлагая им разбирать и собирать колесницу и выяснить, таким образом, что делает колесницу колесницей, т.е. в чем ее сущность. По сути дела, он говорил о системном эффекте, возникающем у целого объекта при взаимодействии его частей (теперь это называется «эмерджентность»), и прививал своим ученикам системное мышление. Таким образом, история возникновения и развития системных исследований показывает, что системный подход есть результат отражения в сознании людей существующей системности реальной действительности.

Из множества существующих описаний системных исследований наиболее содержательным является, по мнению авторов, выделение в них трех следующих аспектов:

- разработка теоретических основ ***системного подхода***;
- построение адекватного системному подходу теоретического аппарата (***теории систем***);
- приложение системных идей и методов (***системный анализ***).

Принцип системного исследования объекта состоит, во-первых, в выявлении его эмерджентных свойств (целостных, не сводимых к свойствам частей) и, во-вторых, в рассмотрении его с позиции системы большего масштаба.

Цель системного исследования – выявление целостных свойств объекта, причины их возникновения и механизмов обеспечения.

Содержание системного исследования заключается в решении двух основных проблем:

- проблемы системного выделения объекта, изучения взаимосвязей его элементов, эмерджентных свойств, механизмов его функционирования и развития;
- проблемы системного моделирования объекта по некоторым заданным свойствам; решение этой проблемы необходимо для создания новых или преобразования существующих систем.

Системное исследование реализуется посредством системного подхода, теоретических системных построений и системного анализа.

1.1. Структура системных исследований как самостоятельного научного направления

Для понимания структуры собственно системных исследований рассмотрим основные компоненты любого самостоятельного научного направления и их взаимодействие, обеспечивающее его функциональную целостность.

Схема взаимодействия компонент самостоятельного научного направления представлена на рисунке 1.1. Описание схемы заимствовано из работ Г.П. Мельникова. На схеме, кроме блоков (компонент научного направления или научной дисциплины), представлено еще два типа связей между этими компонентами. Связи типа «что» или «по данным», по которым осуществляется информационное взаимодействие компонент (обмен знаниями), изображены одинарными стрелками. Связи типа «как» или «по управлению», по которым

осуществляется методологическое или методическое управление компонентами, изображены двойными стрелками.

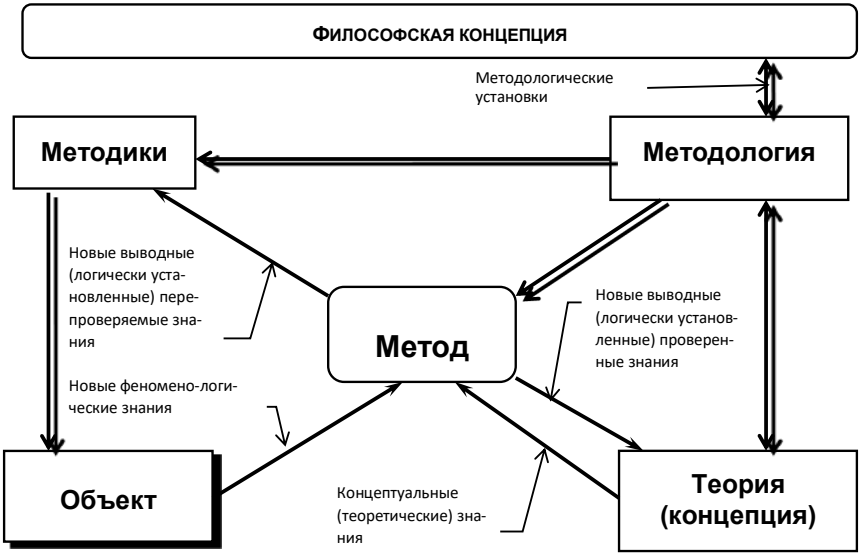


Рис. 1.1. Схема взаимодействия компонент самостоятельного научного направления

Цель исследовательской деятельности – познание наблюдаемых и потенциальных свойств ***объекта*** исследований, являющегося первым необходимым компонентом научного направления. Результат исследовательской деятельности – знания его (объекта) текущего состояния, предыстории становления и умения на достаточно серьезных основаниях прогнозировать его будущее и делать выводы о его поведении, в том числе и в необычных для него условиях.

При рассмотрении научной дисциплины как системы в ее структуре функционально выделяется два вида знаний, обеспечивающих взаимодействия ее компонент. Во-первых, это ***феноменологические*** знания, основанные на учете того, что дано непосредственно в наблюдении за объектом исследования как явлением, феноменом. Во-вторых, это ***логические*** знания, вырабатываемые на основе феноменологических, с учетом знания законов, отражающих причинно-следственные связи явлений. При равной степени полноты и достоверности знаний об изучаемом объекте предпочтительнее тот путь исследования, который позволяет обойтись меньшей долей феноменологически получаемых знаний и, соответственно, большую долю знаний об объекте получать логически.

Отработанные приемы наблюдения и эксперимента для получения новых феноменологических, эмпирических знаний об объекте представляют собой ***методики*** исследований, также являющиеся необходимым компонентом самостоятельного научного направления.

Получение на основании феноменологических знаний новых логически устанавливаемых (выводных) может быть обеспечено только с помощью опорных, априорных знаний более универсального характера. Накопление, хранение и передачу таких знаний об объекте обеспечивает еще один необходимый компонент – ***концепция*** или ***теория***, обслуживающая данное научное направление.

Выведение на основе новых феноменологических знаний и ранее установленных концептуальных знаний новых логически устанавливаемых знаний о данном объекте обеспечивается следующим компонентом научной дисциплины – ***методом*** исследования. Метод представляет собой способ теоретического освоения наблюдаемого и выявленного в эксперименте, ориентированный на определенную феноменологию и концепцию.

Любое самостоятельное научное направление, представляя собой определенную целостность конкретно-научных знаний, не существует, тем не менее, сама по себе. Она функционирует в среде общенаучных знаний, которые периодически привлекаются данной дисциплиной, что обеспечивает формирование навыков и приемов философского осмысления и анализа специальных концепций, теорий, методов и методик данной науки. Эти специфические навыки, возникающие из потребностей философского осмысления проблем конкретной науки и конкретно-научного осмысления философских категорий и законов, формируют особый компонент данной науки – ***методологию***. Методология обеспечивает эффективность всей научной деятельности за счет согласования и усовершенствования всех ее компонентов, особенно в «нештатных» ситуациях.

О существовании самостоятельной научной дисциплины (научного направления) можно говорить в тех случаях, когда достаточно четко сформировались границы ее ***объекта***, сложились ***методики*** получения новых феноменологических знаний, существует ***концепция*** или ***теория*** данной дисциплины. Для существования самостоятельной научной дисциплины, кроме того, должен существовать ***метод*** (или набор методов) данной научной дисциплины и должна сформироваться определенная ***методология***. Лишь в этом случае возможна эффективная исследовательская деятельность, имеющая в своем составе

все необходимые компоненты: объект, методики, метод, теорию или, хотя бы, концепцию и методологию.

Рассмотрим теперь структуру собственно системных исследований как самостоятельного научного направления, впервые предложенную в работе [Бондаренко, 1998] и уточненную в работе [Маторин, 2016, 1].

Схема на рисунке 1.2 иллюстрирует функции каждого из компонент современных системных исследований по аналогии с функциями ***объекта, теории (концепции), метода, методики*** и ***методологии*** любого самостоятельного научного направления.

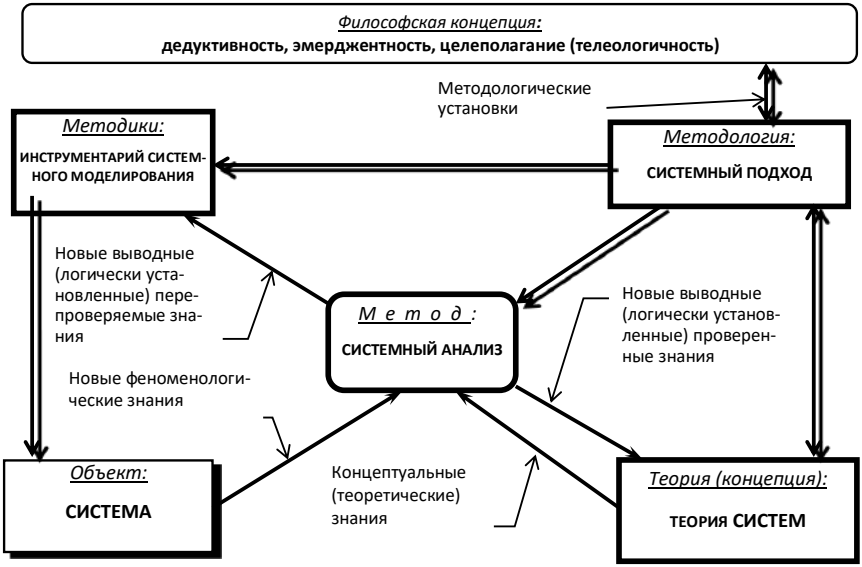


Рис. 1.2. Схема взаимодействия компонент системных исследований

Главной целью исследовательской деятельности является познание свойств и сущности объекта, который в рамках системных исследований представляет собой ***систему***, как предельную абстракцию объекта исследований произвольной природы.

Накопление, хранение и передачу знаний об объекте (системе) обеспечивает концепция или теория, обслуживающая данное научное направление, которая в рамках системных исследований представляет собой ***общую*** или ***абстрактную теорию систем***, как теорию (концепцию) предельной степени абстракции. Из множества существующих

определений этой теории можно упомянуть следующее. ***Общая теория систем*** – это научная и методологическая концепция исследования объектов, представляющих собой системы. Она тесно связана с системным подходом и является конкретизацией его принципов и методов. Первый вариант общей теории систем был выдвинут Людвигом фон Берталанфи, основная идея которого состоит в признании изоморфизма законов, управляющих функционированием системных объектов.

Получение новых феноменологических, эмпирических знаний об объекте (системе) обеспечивают методики исследований, которые в рамках системных исследований должны представлять собой ***инструментарий системного моделирования***, как единственно возможный способ получения данных о предельно абстрактном объекте, то есть системе.

Выведение на основе новых феноменологических и ранее установленных концептуальных знаний новых знаний, логически устанавливаемых, о данном объекте обеспечивается методом исследования. Метод, как средство выведения новых знаний о предельно абстрактных объектах (системах), в рамках системных исследований является ***системным анализом***. Из множества существующих определений этого анализа можно упомянуть следующее. ***Системный анализ*** – это совокупность приемов научного познания, представляющая собой последовательность действий по установлению структурных связей между переменными или элементами исследуемой системы. Он опирается на комплекс общенаучных, экспериментальных, естественнонаучных методов. Системный анализ возник в эпоху разработки компьютерной техники. Успех его применения при решении сложных задач во многом определяется современными возможностями информационных технологий.

Эффективность всей исследовательской деятельности в рассматриваемом направлении за счет согласования и усовершенствования всех ее компонентов, особенно в «нештатных» ситуациях, обеспечивается методологией, которая в рамках системных исследований представляет собой ***системный подход.*** Из множества существующих определений этого подхода можно упомянуть следующие. ***Системный подход*** – это методологическое направление исследования, в основе которого лежит рассмотрение объекта как целостного множества элементов в совокупности отношений и связей между ними, то есть рассмотрение объекта как системы. Говоря о ***системном подходе,*** можно говорить о некотором способе организации наших действий, таком, который охватывает любой род деятельности, выявляя закономерности и взаимосвязи с целью их более эффективного использования.

При этом ***системный подход*** является не столько способом решения задач, сколько способом их постановки. Как говорится, «Правильно заданный вопрос – половина ответа». Это качественно более высокий, нежели просто предметный, способ познания.

Рассмотренный способ представления структуры взаимодействия компонентов системных исследований позволяет предположить, что с увеличением сложности, общности, абстрактности объекта исследований методы и методики исследований должны все в большей степени соответствовать системному подходу и системному анализу. При этом и теоретический аппарат системных исследований, и анализ, и инструментарий будут действительно системными только в том случае, если они опираются на действительно системный подход.

1.2. Основные принципы системного подхода и их эволюция

Для понимания принципов системного подхода и особенно их эволюции необходимо учесть смену научной парадигмы и переход науки от аналитического этапа развития к современному (ноосферному). Этот переход, заключающийся в изменении основных научных принципов и подходов (таблица 1.1), обусловлен непрекращающимся непрерывным ростом научного знания, по поводу существования которого в истории науки имеются веские и неоспоримые доказательства. При этом возникновение и становление системных исследований, в том числе, системного подхода рассматривается как один из признаков смены научной парадигмы.

Широкое внедрение системного подхода во всех сферах научной, конструкторской и управленческой деятельности, как и любая существенная перестройка, является процессом длительным и противоречивым, обусловленным, в первую очередь, продолжением формирования и развития самого системного подхода. В ходе своего развития и совершенствования системный подход (традиционно ***системно-структурный***) не сразу обретает признаки и черты, окончательно свободные от влияния аналитического этапа развития науки и полностью соответствующие современному ноосферному этапу. Среди существующих современных концепций системного подхода такими чертами обладает, например, концепция, которая называется ***системологической*** или просто ***системологией*** [Мельников, 1978].

Таблица 1.1

Смена научной парадигмы

Аналитическая наука	Современная наука
Индуктивность (операционализм): Зависимость свойств целого от свойств частей и изучение целого по его частям. Выведение знаний на измерительной экспериментальной основе	***Дедуктивность:*** Зависимость свойств частей от свойств целого и изучение частей, исходя из целостного характера объектов. Выведение знаний на модельной основе
Элементаризм (редукционизм): Первопричины всех явлений в микромире. Изучение, в первую очередь, морфологических признаков строения и состава	***Эмерджентность (нередукционизм):*** Возникновение у целого свойств, не выводимых на основании свойств частей. Изучение, в первую очередь, функциональных признаков
Антителеологичность: Спонтанность (не целенаправленность) всех процессов и явлений. Необходимость и достаточность законов физики для объяснения любых явлений	***Целеполагания (телеологичность):*** Возможность использования категории цели для изучения объектов произвольной природы. Исследование целенаправленных взаимодействий объектов друг с другом
Дифференциация знаний: Узкая специализация научных исследований и ограничение количества одновременно учитываемых факторов	***Интеграция знаний:*** Междисциплинарный характер научных исследований и многоаспектное, многофакторное рассмотрение объекта

Данная концепция обладает существенными преимуществами, выгодно отличающими ее от известных ранее концепций и уже привычных. Эти преимущества могут быть проиллюстрированы в ходе сравнительного анализа фундаментальных принципов, лежащих в основе любой концепции системного подхода. Это связано с тем, что системный подход, как существенный аспект ноосферного этапа развития науки, в целом, и системных исследований, в частности, принципиально ориентирован на применение для выполнения своих задач ***диалектических принципов: системности, целостности, иерархичности и развития*** [Гвишиани, 1979]. Сравнительный анализ применения основополагающих диалектических принципов традиционным системным подходом и современным системологическим, результаты которого приведены в таблице 1.2, показывает, что первый в значительной степени продолжает использовать методологические установки прошедшего аналитического этапа развития науки и, следовательно, малоэффективен и бесперспективен в современных условиях.

Таблица 1.2

Сравнение системных подходов

Традиционный системный (системно-структурный) подход	Современный системный (системологический) подход
Принцип системности: Понятие системы многозначно (более 40 определений). Понятия «мера» или «степень системности» отсутствуют. Объекты рассматриваются как системы только при определённых условиях [Перегудов, 1989]	***Принцип системности:*** Понятие системы однозначно (предельно абстрактно). Введены понятия «мера» или «степень системности» для рассмотрения любого объекта как системы, имеющей определённую меру [Мельников, 1978]
Принцип целостности и многоаспектности: Учитывается либо только структурная целостность, когда природа связываемых элементов считается несущественной, либо только субстанциальная целостность, когда не учитываются связи между частями целого [Месарович, 1978; Клир, 1990]	***Принцип целостности и многоаспектности:*** Учитывается комплексный характер целостности системы, что обеспечивает взаимосогласование структуры и субстанции системы при её взаимодействии со средой. Основной аспект целостности – функциональный [Косарев, 1978]
Принцип иерархичности: Учитывается иерархичность только внутренней структуры системы, что обеспечивает исследование объектов методом, так называемого, «серого (светлого) ящика» [Полищук, 1989]	***Принцип иерархичности:*** Учитывается, в том числе иерархичность структуры внешней для системы среды, что обеспечивает исследование объектов методом «всё более и более светлого ящика» [Мельников, 1978; Полищук, 1989]
Принцип развития: Рассматриваются только статические параметры системы. Не рассматриваются причины возникновения системы и этапы её становления. Нет понятия «адаптация системы» [Перегудов, 1989; Косарев, 1978]	***Принцип развития:*** Рассматриваются, в том числе, динамические характеристики системы, что обеспечивает понимание причин её возникновения и этапов становления. Введено понятие адаптации системы [Мельников, 1978]

Современный системный подход (системологический), последовательно применяя принципы диалектики, методологически однозначно соответствует новому ноосферному этапу и, следовательно, обладает более высокой эффективностью и перспективностью при решении современных научных и практических задач. Основным результатом последовательного применения принципов диалектики системологией является ее более выраженная и более универсальная методологическая направленность.

Традиционный системный (системно-структурный) подход может функционировать в качестве методологического средства преодоления барьеров конкретного научного познания при возникновении «нештатных ситуаций» или «методологических порочных кругов», но ограничено [Мельников, 1978]. Системология же представляет собой, по сути дела, ориентированное на методологическое использование изложение понятий и принципов диалектики. ***Интерпретированная в терминах конкретной науки системология, может выполнять в этой науке методологические функции неограниченно***, «в том числе при выборе таких оптимальных методов исследования частных задач, которые ... откроют возможность рассмотрения объекта исследования с разных позиций без утраты целостного и сущностного о нем представления» [Мельников, 1989, с. 42].

Названные преимущества системологии позволяют широко применять ее при исследовании слабоструктурированных и слабо формализуемых проблемных областей, представляющих собой основное поле деятельности современной науки.

1.3. Проблемы традиционного системного (системно-структурного) подхода и системно-структурного анализа

Основной проблемой традиционного системного (системно-структурного) подхода, обуславливающей все остальные его проблемы и, в том числе, проблемы системного анализа, является неосуществленный до сих пор окончательный отрыв системного подхода от теоретико-множественного.

Сравнительный анализ системного (системно-структурного) и теоретико-множественного подходов однозначно свидетельствует о том, что эти подходы принципиально противоположны и ни один из них не сводим к другому, что обусловлено, в частности, следующим [Шрейдер, 1982]:

- Во-первых, в концепции множества изначально заложена первичность элемента (части) по отношению к множеству (целому). Множество существует тогда и только тогда, когда тем или другим образом заданы его элементы. В системной же концепции первичным является понятие системы (целого), которая уже потом может быть (а может и не быть) представлена в виде совокупности взаимодействующих частей.

- Во-вторых, теоретико-множественный подход характеризуется абсолютной неразборчивостью, т.е. позволяет рассматривать как одно множество любую совокупность любых явлений (с учетом известных парадоксов). Системный же подход претендует на рассмотрение действительности (предметной области) в виде естественно взаимодействующих системных образований, что накладывает определенные ограничения на представление совокупности явлений в виде одной системы.
- В-третьих, теоретико-множественный подход (как следует из выше сказанного) характеризуется гносеологичностью, так как реальные объекты не имеют теоретико-множественной природы. Системный же подход, по своему замыслу, ориентирован на описание целостной природы реальных объектов и, таким образом характеризуется онтологичностью, так как реальные объекты имеют системную природу.

Применение, следовательно, для описания системных отношений (методов и процедур системного анализа) аппарата теории множеств или другого, сводимого к теоретико-множественному, фактически сводит на нет специфические особенности и преимущества системного подхода [Безматерных].

Не удивительно, что сложные явления и процессы (т.е. то, что действительно имеет системную природу) оказываются «не по зубам» традиционному системному анализу. Все что формализовано, на сегодняшний день, под вывеской системного подхода и системного анализа могло быть сделано в принципе под другой (какой-либо теоретико-множественной), так как фактически сделано обычными традиционными формальными средствами, не описывающими специфические системные отношения.

Второй существенной проблемой традиционного системного (системно-структурного) подхода и, следовательно, системного анализа является формализация его понятий без учета их специфического содержания.

По поводу роли формального аппарата и математических методов в системных исследованиях отмечается, например, в [Никаноров, 1969; Гиг, 1981], что основное содержание системного анализа заключено не в формальном математическом аппарате, описывающем «системы» и «решение проблем» и не в специальных математических методах, … а в его концептуальном, т.е. понятийном аппарате, в его идеях, подходе и установках. При этом в работе [Гиг, 1981, с. 73–74] подчёркивается,

что «реальные системы не полностью поддаются описанию с помощью математических моделей» и что аналитические (т.е. формальные) методы непригодны для изучения живых и, следовательно, социальных (организационных) систем. Кроме того, «использование математики переносит акцент с содержания на структуру явления» [там же, с. 86]. При этом теоретические системные построения, основанные на результатах физико-математических наук, автор упомянутой работы называет *теориями жёстких систем*, «применение которых к экономическим и организационным системам позволяет создать количественные модели чрезвычайно бедные, однако, по своему содержанию» [там же, с. 103]. Более того, там же подчёркивается, что «если теория связана только с понятиями структуры и цели и не связана с понятиями субстанции и содержания, то бесполезно ожидать появления конкретных полезных приложений такой системной теории» [там же].

При применении методов математической статистики зачастую предполагается независимость, одинаковая распределенность и нормальность используемых совокупностей случайных величин [Колесников, 1]. Однако, «данные предположения, как правило, не выполняются, и это обстоятельство может приводить к потере точности и достоверности результатов моделирования …. Хотя в моделировании существуют приемы сведения данных к виду, пригодному для использования традиционных методов статистики, эти методы часто носят эвристический характер либо приспособлены для изучения частной модели» [Колесников, 1, с. 19].

В работе [Бусленко, 1978] отмечается, что анализ и моделирование систем с помощью средств моделирования случайных процессов или теории массового обслуживания «…зачастую носят кустарный характер и уже не соответствуют современным запросам практики. Они приводят, по мере возрастания сложности задач, к значительному увеличению трудоемкости подготовки моделей…. Эти обстоятельства снижают эффективность имитационного моделирования и препятствуют его широкому распространению как инструмента повседневного использования…» [Бусленко, 1978, с. 202].

Трудности практического использования моделей математического программирования связаны, прежде всего, с обеспечением полноты, точности и достоверности исходных данных, необходимых для модели, с учетом динамики функционирования системы и с многокритериальным характером выбора варианта структуры [Емельянов, 2014].

При этом весьма показательным является непринятие стандарта математического моделирования – IDEF2, которое «было вполне естественным, так как при реализации математической модели приходилось либо жертвовать ее точностью, либо ... самой возможностью что-то моделировать, ввиду чего никогда нельзя было с полной уверенностью говорить о соответствии математической модели функциональной» [Колесников, 2].

Таким образом, именно использование чисто формальных математических средств является одной из причин определенной ограниченности существующих методов системного анализа организационных систем и их несоответствия содержательному по своей природе объектно-ориентированному подходу к проектированию ИС. С этой точки зрения весьма существенным для данного исследования является введение еще В.М. Глушковым понятия *обобщённых динамических систем* (организмы, предприятия, государства и т.д.), которые принципиально не могут быть описаны классической математикой [Глушков, 1972].

Кроме того, объекты, исследуемые средствами системного подхода и системного анализа, как правило, являются не строго и не точно определенными. Это позволяет оставаться актуальным известному предупреждению Н. Винера о том, что применение точных формул к вольно определяемым понятиям есть не что иное, как обман и пустая трата времени [Винер, 1960], а также резкому суждению А. Эйнштейна о том, что математика может доказать что угодно и использоваться как отличнейшее средство водит за нос даже самого себя, а главное состоит в содержании, а не в математике [Гернек, 1966].

Третьей проблемой традиционного системного (системно-структурного) подхода и, основанных на нем системного анализа и теории систем, является не развитость средств *анализа* и *синтеза* объектов как систем. Основной причиной существования данной проблемы является то, что само понятие «система» до сих пор остается дискуссионным.

Любая же научная теория, особенно обслуживаемая формальной системой, для выполнения своих многообразных функций (информативной, систематизирующей, прогностической, объяснительной) должна обладать средствами, позволяющими осуществлять процедуры анализа и синтеза описываемых данной теорией объектов. Совершенно очевидно, что теоретико-множественный подход в лице теории множеств (с четко определенным понятием «множество») такими сред-

ствами обладает, что и подвигает, так сказать «пользователей», на применение этой теории для решения конкретных задач, в частности задач системного анализа.

В рамках же традиционных вариантов системного подхода (их анализ, например, в работе [Агошкова, 1998]) используется более 40 определений; например, уже в работе [Уемов, 1978] собрано 35. При этом эти определения не обуславливают (не задают) конкретной возможности для проведения операций анализа или синтеза объектов как систем, т.е. с учетом специфических системных отношений. Данные операции либо вообще остаются без определения, либо определяются с помощью теоретико-множественных средств с потерей возможности целостного представления системы. А, как известно, «системные представления, построенные при игнорировании признака целостности, оказываются неэффективными или даже дают заведомо ошибочные результаты» [Агошкова, 1998, с. 73].

Например, в одном из самых первых вариантов системного подхода система определялась как средство решения проблем, но конструктивных подходов к анализу и синтезу таких средств не предлагалось [Оптнер, 1969]. В более поздних вариантах система рассматривается как упорядоченное определенным образом множество взаимосвязанных между собой компонент той или иной природы, характеризующееся единством (целостностью), которое выражается в интегральных свойствах и функциях множества [Садовский, 1974]. При этом заданная возможность теоретико-множественного анализа и синтеза таких систем (как множеств) исключает конструктивное определение специфической целостности системы и причины появления её интегральных, т.е. собственно системных свойств. В теории организации, например, система определяется как целое, созданное из частей и элементов, для целенаправленной деятельности. При этом системе приписывается стремление к сохранению структуры, потребность в управлении и сложная зависимость свойств от входящих в нее элементов [Смирнов, 1998]. Все эти характеристики, однако, имеют в рамках данной теории исключительно лингвистическое описание.

Совершенно очевидно, что в названных концепциях способы анализа и синтеза систем с учетом их специфических системных отношений не заданы, не определены. Такая же ситуация существует и в рамках формализованных системных построений. Практически все формальные определения системы или непосредственно, или после своего раскрытия имеют теоретико-множественный характер, в рамках которого либо затруднено проведение анализа и синтеза систем, либо

фактически не учитывается целостность этих систем. Довольно большая коллекция таких определений представлена, например, в работах [Волкова, 2006; Волкова, 2015].

В классическом труде [Месарович, 1978], описывающим так называемый феноменологический системный подход, исходное целостное представление системы в виде декартова произведения множеств входных (X) и выходных (Y) объектов: $S \subset X \times Y$, является представлением «черного ящика», процедура анализа которого как системы, естественно, не определена. Для обеспечения анализа и синтеза систем в названной работе вводится функциональное представление системы: $S: X \rightarrow Y$. Уточнение последнего представления осуществляется путем введения понятия глобальной реакции системы: $R: (C \times X) \rightarrow Y$, которое, в свою очередь, зависит от множества внутренних состояний (C) системы. Введение в рассмотрение множества внутренних состояний обеспечивает возможность анализа и синтеза функциональных систем в представлении [там же] теоретико-множественными средствами, однако, ценой отказа от целостного системного представления. Последнее обстоятельство обусловлено тем, что для определения системы (при таком ее понимании) исследователь, по сути дела, обязан рассматривать внутренности данной системы и не ее целостность.

Таким образом, включение в формальное определение системы кроме (или вместо) множества ее элементов множества свойств или состояний, а также различных вариантов отношений между ними ничего не меняет по существу данной проблемы. Например, в работе [Бусленко, 1978] система моделируется с помощью, так называемых, *кусочно-линейных агрегатов,* образовываемых с помощью трех множеств: входов, выходов и состояний. Решая, безусловно, с помощью такого подхода ряд практических задач для некоторых классов систем, тем не менее, невозможно обеспечить целостное рассмотрение системы и «инкапсуляцию» ее свойств, необходимую в ходе объектно-ориентированного проектирования ИС.

В еще одной классической работе [Клир, 1990], в которой система вообще рассматривается не как реальная вещь, а как абстрагирование или отображение некоторых свойств реального объекта, формальное представление системы осуществляется путем задания множества свойств объекта и их проявлений, а также так называемой «базы» и множества ее элементов. При этом база может состоять из элементов исследуемого объекта (системы) со всеми вытекающими отсюда и

отмеченными выше достоинствами и недостатками теоретико-множественного подхода. В других случаях база может представлять собой некоторые контекстные ситуационные характеристики и, тогда, процедуры анализа и синтеза таких систем не определяются.

Даже если применяются, чисто логические методы исследования систем или мощный алгебраический аппарат, сама система все равно определяется через понятие множества каких-либо входящих в систему сущностей [например: Левин, 1987]. Включение же в определение системы понятия среды [например: Хомяков, 2006; Бурков, 1999] фактически еще более усложняет процедуры анализа и синтеза объектов как систем, так как затрудняет определение границ системы.

При этом необходимо иметь в виду, что представление системы в виде множества ее взаимосвязанных элементов (свойств, состояний и т.п.) рано или поздно приводит, в конце концов, к паре множеств: множеству элементов (свойств, состояний и т.п.) и множеству связей (отношений), заданному на первом множестве. Это, в свою очередь, приводит к пониманию системы как модели, т.е. как некоторому средству описания реально существующих объектов. В настоящее же время общепризнанно, что системность является неотъемлемым атрибутом действительности, т.е. имеет не гносеологический, а онтологический статус [например: Шрейдер, 1982].

Таким образом, предполагаемая в традиционных подходах к системе её структурированность является не более чем декларацией, так как путей конкретной реализации структурирования объекта как системы (а не как множества) не указывается. При этом в теории множеств возможность анализа задана и однозначно определена. Она задана в самой концепции множества, согласно которой множество задается через его элементы, т.е. между множеством и его элементами существуют, хотя и примитивные, но однозначно оговоренные отношения. Таким образом, пути и способы теоретико-множественного анализа оказываются четко определенными, что находит свое воплощение в формальном аппарате, в первую очередь, в операции «$\in$«. Отношение же целое – часть (система – подсистема (элемент)) в упомянутых системных подходах не определено. Утверждается только факт его существования. Это приводит к тому, что способ анализа системы определяется, по сути дела, прихотью аналитика, обусловленной, конечно, решаемой задачей, но не приобретающей от этого определенности и объективности [Калашников, 1980].

Например, во множестве «Города России» любой исследователь в ходе анализа будет выявлять именно населенные пункты, имеющие

статус города, а не что-либо ещё; во множестве «Животные заповедников» – животных (не их части, не их сообщества и т.п.), про которых известно, что они живут в заповедниках. В ходе же системного анализа, например, системы «Город» или системы «Биоценоз» разные аналитики выявят разные части (подсистемы), в зависимости не только от целей анализа, но и от своих субъективных предпочтений [например: Игнатьев, 1998]. Причем, с точки зрения именно традиционного системного подхода, все они будут иметь для своих действий одинаковые основания, т.е. полное их отсутствие, так как путь (способ) анализа системы в рамках упомянутых и наиболее распространенных системных концепций априорно не определен. В публикациях аналитиков об этом так прямо и говорится: «Это значит, что мы можем увидеть, выделить и изучить в городе столько систем, сколько захотим или сколько требует реальная практика управления» [Заец, 1990, с. 93].

По отношению к процедуре синтеза в традиционных системных подходах (в сравнении с теоретико-множественным) существует аналогичное положение. В рамках теоретико-множественного подхода отношение части к целому строго задано заранее оговоренными правилами «сборки» частей в целое в виде соответствующих операций над множествами. В рамках же обсуждаемых системных подходов процедура синтеза системы в нечто еще более целое опять становится зависящей от субъекта исследования, его понимания текущей ситуации, так как простое декларирование функционирования системы в среде или подчиненности системы (выходов системы) некоторой цели, если не оговариваются отношения со средой или что это за цели, чьи они, откуда берутся, остается не более чем лозунгом, ровным счетом ничего не дающим для обоснованного проведения операции синтеза. Данная ситуация в системных исследованиях даже получила свое особое наименование: *системный эффект* или *эмерджентность* свойств целого. При этом признается, что в ходе синтеза целого (системы) из его частей (подсистем) возникают принципиально новые свойства, но механизм синтеза или, как его еще называют, «*алгоритм сборки*» остается до сих пор тайной за семью печатями [например: Моисеев, 1989]. И в этом нет ничего удивительно, так как в наиболее распространенных системных подходах процедура синтеза (путь соединения объектов как систем, а не как множеств) не задана в принципе.

Например, множество «Города России» можно однозначно описать (синтезировать) путем объединения множеств «Города области…»; множество «Животные заповедников» – путем объединения

множеств «Животные заповедника №…». Синтезирование же системы «Город» из подсистем населенного пункта такого вида или системы «Биоценоз» из частей, представляющих собой различные виды организмов, проживающих на данной территории, в значительной степени является эвристической процедурой, которую разные исследователи могут и будут выполнять по-разному.

Четвертой проблемой традиционного системного (системно-структурного) подхода и анализа является то, что ни один вариант такого системного подхода и ни один, основанный на нем, метод системного анализа не использует понятия класса при осуществлении своих процедур и построении моделей.

Т.е. в существующих теоретических построениях по поводу системного подхода не учитываются различные пути проявления системности. Говоря о системах, как правило, имеют в виду только конкретные объекты, явления. При этом еще в работе [Шрейдер, 1982] была обоснована необходимость учета в рамках системного подхода и теории систем не только конкретных объектов. В соответствии с упомянутой работой в зависимости от пути проявления целостности, как основного признака системности, имеет смысл рассматривать два вида систем: ***внутренние*** и ***внешние***.

Внутренняя система (наш термин: ***система-явление***) – это целостное образование (конкретный объект), к которому можно применить процедуры членения, представляя эту систему в виде некоторой структуры составляющих частей [там же].

Внешняя система (наш термин: ***система-класс***) – это класс объектов общей природы, объединенных некоторой целостной сущностью. Элементы такой системы «могут не обладать ни пространственной, ни временной общностью, ни даже генетической связью… Важна лишь общность природы образующих систему объектов» [там же, с. 69].

Таким образом, традиционный системный подход в принципе не использует концептуальных классификационных моделей, применение которых обязательно при осуществлении объектно-ориентированного анализа при разработке ИС [например: Буч, 2010; Бондаренко, 2004]. Поэтому при использовании традиционных системных методов для проведения объектно-ориентированного проектирования ИС все равно приходится выходить за их рамки и использовать дополнительные средства [Йордан, 1999].

Следовательно, основными проблемами традиционного системного подхода и, основанных на нем методов системного анализа и теории систем, а также их несоответствия объектной парадигме разработки ИС является теоретико-множественный и вообще чисто формальный подход к понятию системы, а также неопределенность процедур анализа и синтеза объектов как систем и отсутствие в арсенале системного подхода и анализа инструментальных средств классификационного моделирования.

Данные проблемы являются причиной, в частности, того, что средства системно-структурного подхода не могут быть эффективно использованы при разработке ПО средствами технологии объектно-ориентированного проектирования и программирования, которая в настоящее время является основной и самой перспективной [там же].

Все методы системно-структурного анализа полностью ортогональны принципам объектно-ориентированного проектирования [Буч, 2010].

Очень показательны, в данном случае, попытки применения при выполнении объектно-ориентированного проектирования методов системного структурного анализа, основанных на системно-структурном подходе. Относительно конкретных диаграмм, например диаграмм ERD, в [Йордан, 1999, с. 24–25] сделаны следующие выводы: «Применение ERD очень выгодно, однако при работе с этими средствами возникают определенные трудности. Во-первых, идентифицированные сущности не всегда соответствуют понятиям области приложения, особенно когда аналитик пытается создать сущности в третьей нормальной форме. Во-вторых, ERD не подходят для идентификации объектов, не хранящих данные; в эту категорию часто попадают, например, объекты, распознающие происхождение событий или осуществляющие функции контроля». В общем, авторы приходят к следующему выводу: «Методы ЗVM, весьма полезные для идентификации компонентов или объектов, не позволяют, тем не менее, идентифицировать ***подходящее*** множество объектов для разрабатываемой системы. Процесс обнаружения объектов все ещё управляется мнениями, интуицией и инсайтом» [там же, с. 26].

Следствием ортогональности системно-структурного анализа и объектно-ориентированного проектирования является, таким образом, проблема использования результатов анализа при проектировании объектно-ориентированного ПО. Специалисты отмечают, что на этапе анализа лучше использовать методы системно-структурного анализа

(ЗVM, IDEF и т.д.), так как «аналитик имеет дело с бизнес-процессами, по сути, являющимися функциями». Однако, с точки зрения проектирования ПО, конечно, необходимо использовать объектные методы (на основе UML) особенно в случае разработки сложных программных приложений. При этом, естественно, «описание бизнес-процессов должно делаться в том же средстве, в котором на последующих этапах работ будет проектироваться ИС» [Сахаров, 2000]. Это и создает противоречие.

2. Основные варианты системного подхода

2.1. Системно-структурный подход

У системно-структурного подхода к настоящему времени сформировалось два варианта его реализации: структурно-функциональный подход и процессный подход.

2.1.1. Структурно-функциональный подход

Структурно-функциональный подход – один из видов системного исследования явлений и процессов как структурно расчлененной целостности, в которой каждый элемент структуры имеет определенное функциональное назначение.

Основанный на данном подходе структурно-функциональный анализ при тщательном подходе к рассмотрению той или иной проблемы позволяет определить относительную роль каждого из взаимодействующих элементов и осуществить управление процессами, как социальными, так и экономическими. Структурно-функциональный анализ направлен главным образом на исследование механизмов, обеспечивающих стабильность системы, при этом, не выступая в качестве самостоятельной теории или ее методологической основы.

Таким образом, главными (ключевыми) понятиями структурно-функционального подхода являются понятия: *структура* и *функция*.

Структура – основная характеристика системы, совокупность устойчивых связей системы, обеспечивающих ее целостность и тождественность самой себе, сохранение ее основных свойств при различных внешних и внутренних изменениях.

Функция – исполнение обязанности, круг деятельности, способ поведения, присущий какому-либо объекту и способствующий сохранению существования этого объекта или той системы, в которую он входит в качестве элемента.

Практическая реализация структурно-функционального подхода к системному анализу позволяет выстроить ряд аксиоматических положений [Родионов, 2013].

1. *Независимость цели*. Цель не зависит от системы, поскольку определяется не данной системой, не ее потребностью, а потребностью

другой системы. Но понятие «система» по отношению к данной системе зависит от цели, т.е., от соответствия возможностей данной системы выполнить заданную извне цель, и потому система строится под цель, а не наоборот.

2. *Специализация функций системы*. В ответ на определенное внешнее воздействие система всегда дает определенный результат действия.

3. *Цельность системы*. Система проявляет себя как единичный и целостный объект. Это вытекает из единства цели, которое присуще только системе в целом, но не ее отдельным элементам в частностях. Цель объединяет элементы системы в единое целое.

4. *Ограниченная дискретность системы.* Нет ничего неделимого и любую систему можно разделить на элементы. При этом любая система состоит из конечного числа элементов.

5. *Иерархичность системы*. Элементы системы находятся в различных отношениях между собой и место каждого из них является местом на иерархической лестнице системы. Система хотя и проявляет себя как единичный и целостный объект, но состоит из элементов, т.е., систем более низкого порядка. В то же время она сама может быть системой (подсистемой, элементом), входящей в состав системы более высокого порядка. Иерархичность систем обусловлена иерархичностью целей.

6. *Функция системы*. Результат действия системы является ее функцией. Для достижения цели система должна целенаправленно выполнять определенные действия, в результате которых появляется функция системы. Цель является аргументом для системы, а результат действия системы – функцией.

7. *Результативность систем*. Соответствие результата действия поставленной цели характеризует результативность систем, которая напрямую связана с их функцией. Функция системы в плане результативности может быть достаточной, гиперфункцией, отстающей и полностью недостаточной. Система выполняет какие-либо действия, и это приводит к появлению ее результата действия, который должен соответствовать той цели, для которой данная система создана.

8. *Ограниченность действий системы*. Любая система характеризуется качественными и количественными ресурсами. В понятие ресурсы входит понятие функционального резерва – какие действия и сколько таких действий система может выполнить. Качественные ресурсы определяются типом элементов исполнения, а количественные ресурсы – их количеством.

9. *Дискретность функций системы*. Действия системы всегда дискретны, поскольку любые ее функциональные структурные элементы действуют по принципу «да или нет».

10. *Коммуникативность систем*. Сопряженные системы взаимодействуют между собой. В этом взаимодействии заключается связь между системами, их коммуникативность.

11. *Управляемость систем.* Любая система содержит элементы (системы) управления, которые контролируют соответствие между результатом действия системы и поставленной целью. Эти элементы управления образуют блок управления. Задание цели для данной системы всегда задает верхняя (старшая) в цепи иерархии система. Самоорганизующиеся системы могут поменять некоторые параметры цели, в зависимости от внешней ситуации, но, ни одна система не может поменять собственную цель, поскольку цель всегда задается извне.

2.1.2. Процессный подход.

Процессный подход – концепция управления, в соответствии с которой вся деятельность организации рассматривается как набор процессов.

Процессный подход был разработан и применяется с целью создания горизонтальных связей в организациях. Подразделения и сотрудники, задействованные в одном процессе, могут самостоятельно координировать работу в рамках процесса и решать возникающие проблемы без участия вышестоящего руководства. Процессный подход к управлению позволяет более оперативно решать возникающие вопросы и воздействовать на результат.

В отличие от функционального подхода, управление процессами позволяет концентрироваться не на работе каждого из подразделений, а на результатах работы организации в целом. Процессный подход меняет понятие структуры организации. Основным элементом становится процесс. В соответствии с одним из принципов процессного подхода организация состоит не из подразделений, а из процессов.

Процессный подход основывается на нескольких принципах. Внедрение этих принципов позволяет значительно повысить эффективность работы, однако вместе с тем, требует и высокой корпоративной культуры. Переход от функционального управления к процессному требует от сотрудников постоянной совместной работы, несмотря на то, что они могут относиться к различным подразделениям. От того, насколько удастся обеспечить эту совместную работу,

будет зависеть «работоспособность» принципов, заложенных в процессный подход. При внедрении управления по процессам важно придерживаться следующих принципов [Процессный подход]:

- *Принцип взаимосвязи процессов*. Организация представляет собой сеть процессов. Процессом является любая деятельность, где имеет место выполнение работ. Все процессы организации взаимосвязаны между собой;
- *Принцип востребованности процесса*. Каждый процесс должен иметь цель, а его результаты должны быть востребованы. У результатов процесса должен быть свой потребитель внутренний или внешний.
- *Принцип документирования процессов*. Деятельность по процессу необходимо документировать. Это позволяет стандартизовать процесс и получить базу для изменения и дальнейшего совершенствования процесса;
- *Принцип контроля процесса*. Каждый процесс имеет начало и конец, которые определяют границы процесса. Для каждого процесса в рамках заданных границ должны быть определены показатели, характеризующие процесс и его результаты;
- *Принцип ответственности за процесс*. В выполнении процесса могут быть задействованы различные специалисты и сотрудники, но отвечать за процесс и его результаты должен один человек.

Главное понятие, которое использует процессный подход – это понятие процесса.

Процесс – это совокупность взаимосвязанных и взаимодействующих видов деятельности, которые преобразуют входы в выходы (ISO 9001).

Важной составляющей процесса, которая не отражена в этом определении, является систематичность действий. Действия процесса должны быть повторяющимися, а не случайными.

Процессный подход предполагает наличие ключевых элементов, без которых он не может быть внедрен в организации. К таким ключевым элементам относятся: Вход процесса; Выход процесса; Ресурсы; Владелец процесса; Потребители и Поставщики процесса; Показатели процесса.

Входами процесса являются элементы, претерпевающие изменения в ходе выполнения действий. В качестве входов процессный подход рассматривает материалы, оборудование, документацию, различную информацию, персонал, финансы и пр.

Выходами процесса являются ожидаемые результаты, ради которых предпринимаются действия. Выходом может быть, как материальный продукт, так и различного рода услуги или информация.

Ресурсами являются элементы, необходимые для процесса. В отличие от входов, ресурсы не изменяются в процессе. Такими ресурсами процессный подход определяет оборудование, документацию, финансы, персонал, инфраструктуру, среду и пр.

Владелец процесса для процессного подхода является одним из самых главных. У каждого процесса должен быть свой владелец. Владельцем является человек, имеющий в своем распоряжении необходимое количество ресурсов и отвечающий за конечный результат (выход) процесса.

У каждого процесса должны быть поставщики и потребители. *Поставщики* обеспечивают входные элементы процесса, а *потребители* заинтересованы в получении выходных элементов. У процесса могут быть как внешние, так и внутренние поставщики, и потребители. Если у процесса нет поставщиков, то процесс не будет выполнен. Если у процесса нет потребителей, то процесс не востребован.

Показатели процесса необходимы для получения информации о его работе и принятии соответствующих управленческих решений. Показатели процесса – это набор количественных или качественных параметров, характеризующих сам процесс и его результат (выход).

За счет того, что процессный подход создает горизонтальные связи в работе организации, он позволяет получить ряд преимуществ, в сравнении с функциональным подходом. Основными преимуществами процессного подхода являются:

- Координация действий различных подразделений в рамках процесса;
- Ориентация на результат процесса;
- Повышение результативности и эффективности работы организации;
- Прозрачность действий по достижению результата;
- Повышение предсказуемости результатов;
- Выявление возможностей для целенаправленного улучшения процессов;
- Устранение барьеров между функциональными подразделениями;
- Сокращение лишних вертикальных взаимодействий;
- Исключение невостребованных процессов;

- Сокращение временных и материальных затрат.

Сравнение описанный выше подходов показывает, что это один и тот же системно-структурный подход. Дело в том, что из определений понятий «функция» и «процесс» следует, что и то и другое понятие описывают деятельность. При этом совершенно очевидно, что любая деятельность всегда функциональна и всегда соответствует некоторому процессу. Кроме того, любая функциональность реализуется в виде процесса, а любой процесс реализует некоторое соответствие выхода процесса его входу, т.е. функцию. Отличие их состоит только в том, с какой целью мы анализируем структуру некоторой деятельности. Если нас интересует структура деятельности составных частей (подсистем) некоторой организации и как они (части) поддерживают эту организацию своим функционированием, то мы акцентируем внимание на функциях этих частей (т.е. процессах в них происходящих) и говорим о *структурно-функциональном анализе*. Такой подход больше соответствует задачам организационного проектирования. Если же мы хотим разобраться в структуре деятельности некоторых элементов организации, обеспечивающих получение конкретного результата, то мы акцентируем внимание на процессах, которые эти элементы выполняют, выделяя упомянутые элементы в отдельную подсистему, и говорим о *процессном анализе*. Такой подход в большей степени соответствует задачам управления бизнес-процессами.

В любом случае анализ функциональности не возможен без анализа процедур и процессов, которые выполняются для обеспечения этой функциональности; анализ процессов, по сути дела, выявляет соответствующие функции элементов, выполняющих эти процессы.

2.2. Объектно-ориентированный подход

2.2.1. Основные понятия и особенности объектно-ориентированного подхода

Объектно-ориентированный подход, будучи ортогональным системно-структурному подходу, по сути дела, сам по себе является, в определенной степени, системным, так как использует основной принцип системного подхода: принцип целостности, а также классы (системы-классы) при построении объектно-ориентированных моделей.

Рассмотрим в соответствии с [Буч, 2010 и Вендров] основные особенности и понятия данного подхода.

Особенностью данного подхода является описание не процедур, необходимых для решения задачи, а тех сущностей, которые участвуют в этих процедурах и их обеспечивают. Дело в том, что процедуры осуществляются не сами по себе, не каким-то абстрактным образом, а путем взаимодействия некоторых вполне конкретных объектов, в соответствии с имеющимися у них свойствами. И если эти объекты и их свойства описаны в программе, то остаётся только дать этим объектам возможность взаимодействовать. Процедуры, которые необходимо выполнить для решения задачи, будут осуществляться как результат этого взаимодействия.

Объектно-ориентированный подход реализуется путем последовательного выполнения:

- ***Объектно-ориентированного анализа* (object-oriented analysis – OOA),** при котором моделируемая и разрабатываемая системы анализируются с точки зрения ***классов*** и ***объектов*****,** выявленных в предметной области.
- ***Объектно-ориентированного проектирования* (object-oriented design – OOD)**, при котором путем ***объектной декомпозиции*** создается ***объектная модель*** разрабатываемой системы.
- ***Объектно-ориентированного программирования* (object-oriented programming – OOP),** при котором программа представляется в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют ***иерархию наследования*.**

В соответствии с требованиями OOA и OOD для построения объектной модели сложной системы её необходимо представить в ***канонической форме***. Эта форма представления системы включает в себя две ортогональных иерархии: ***иерархию классов*** (таксономию или родовидовую классификацию) и ***иерархию объектов*** (мерономию или классификацию часть-целое). Предполагается, что такая объектно-ориентированная декомпозиция системы, позволяет вскрыть ее полную архитектуру, т.е. структуру классов и структуру объектов.

При этом если объект, как экземпляр класса, соответствует конкретному предмету или явлению, определенному во времени и в пространстве, то класс – абстракции существенного в объекте. Таким образом, в рамках объектно-ориентированного подхода ***класс*** рассматривается как множество объектов (экземпляров), имеющих общую структуру и общее поведение (внешняя система или система-класс). ***Объект*** же представляет собой конкретный опознаваемый

предмет (внутреннюю систему или систему-явление), имеющий четко определенное функциональное назначение в данной предметной области.

При этом свойства (***состояние*** и ***поведение***) объектов, как экземпляров классов, в объектной модели, определяет соответствующий класс в иерархии классов, описывающей моделируемую предметную область. Главной же задачей OOA и OOD считается выбор правильного набора абстракций для описания заданной предметной области, что подчеркивает важность концептуального классификационного моделирования в процессе объектно-ориентированной разработки.

Концептуальная база объектно-ориентированного подхода включает в себя четыре главных взаимосвязанных понятия: ***абстрагирование, иерархия, инкапсуляция и модульность.***

Абстрагирование – способ выделения существенных характеристик некоторого объекта (абстракций), отличающих его от всех других видов объектов и, таким образом, четко определяющих его концептуальные границы. Это понятие реализует возможность объектно-ориентированного подхода оперировать с системами-классами (внешними системами).

Иерархия – способ упорядочения абстракций (классов, систем-классов) по уровням.

Инкапсуляция – способ отделения элементов объекта (класса), определяющих его устройство, от элементов, определяющих его поведение (т.е. отделение ***реализации*** от ***интерфейса***). Это понятие реализует принцип целостности объектного подхода как системного.

Модульность – способ разложения системы на связанные, но относительно самостоятельные части (модули).

Объектно-ориентированное моделирование (анализ), проектирование и программирование в обязательном порядке основано на использовании названных понятий при описании классов и объектов.

Описание класса включает ***атрибуты***, с помощью которых характеризуется состояние объектов данного класса, и ***операции***, с помощью которых характеризуется функционирование (поведение) объектов данного класса. При этом разграничивается внешний облик класса (спецификация или интерфейс) и его внутреннее устройство (реализация). Главное в интерфейсе – объявление операций, поддерживаемых экземплярами класса. В него также могут входить объявления других классов, переменных, исключительных ситуаций, уточняющих абстракцию, которую класс должен выражать. Реализация, как правило, состоит в определении операций, объявленных в интерфейсе класса.

Между классами могут существовать следующие отношения:

- ***Наследование*** – такое отношение между классами (родовидовое отношение), при котором класс повторяет описание состояния и поведения суперкласса (одного – одиночное наследование; нескольких – множественное наследование). Данное отношение является основным при выявлении иерархии классов предметной области. Узкие, специализированные классы в иерархии, от которых создаются экземпляры (объекты), называются ***конкретными классами***. Общие классы, от которых экземпляры не производятся, называются ***абстрактными классами***. Самый общий класс в иерархии классов называется ***базовым*** или корневым.
- ***Ассоциация*** – отношение семантической зависимости, показывающее какие роли классы играют друг для друга. Ассоциации различаются по мощности: «один-к-одному», «один-ко-многим», «многое-ко-многим».
- ***Агрегация*** – отношение, соответствующее отношению «часть-целое» между объектами данных классов.
- ***Использование*** – может рассматриваться как разновидность отношения ассоциации, при котором одна из сторон (***клиент***) пользуется услугами или ресурсами другой стороны (***сервера***). Кроме того, использование может рассматриваться как один из аспектов отношения наследования, так как подкласс, наследуя состояние и поведение класса, выступает в роли его клиента. Так же клиентом класса является его экземпляр (объект), который использует атрибуты и операции данного класса.

При этом операция рассматривается как услуга, которую класс может предоставить своим клиентам. Как правило, эти операции бывают следующих видов:

- ***Модификатор*** – операция изменения состояния объекта.
- ***Селектор*** – операция считывания состояния объекта (без изменения состояния).
- ***Итератор*** – операция, организовывающая доступ ко всем частям объекта в строго определенной последовательности.
- ***Конструктор*** – операция создания и/или инициализации объекта.
- ***Деструктор*** – операция, освобождающая состояние объекта и/или разрушающая сам объект.

Описание объекта включает в себя описание его состояния, которое характеризуется перечнем атрибутов, соответствующих классу,

экземпляром которого является данный объект, и их текущими значениями. Кроме того, описание объекта включает также описание его поведения (функции), которое характеризуется методами, реализующими операции класса, экземпляром которого является данный объект.

Объединяя понятия состояния и поведения объекта, можно ввести понятие ***роли*** или ***ответственности*** объекта. Ответственность объекта имеет две стороны – знания, которые объект поддерживает, и действия, которые объект может исполнить. Она выражает смысл его предназначения и место в системе. Ответственность понимается как совокупность всех услуг и всех контрактных обязательств объекта. Следовательно, состояние и поведение объекта определяют его роль в системе, а она, в свою очередь, необходима соответствующему классу для выполнения его ответственности.

При выявлении структуры объектов моделируемой системы между объектами устанавливаются различные ***связи*** (посредством передачи сообщений), представляющие собой физическое или концептуальное соединение между объектами. Связи обозначают равноправные или «клиент-серверные» отношения между объектами. Наиболее желательной является функциональная связность, при которой все элементы тесно взаимодействуют в достижении определенной цели. Участвуя в связях, объект может выполнять одну из трех ролей:

- ***актора*** – при которой объект может воздействовать на другие объекты, но сам никогда не подвергается их воздействию (активный объект);
- ***сервера*** – при которой объект может только подвергаться воздействию со стороны других объектов, но никогда не выступает в роли воздействующего объекта (пассивный объект);
- ***агента*** – при которой объект может быть и активным, и пассивным.

Кроме того, между объектами могут быть выявлены иерархические отношения, а именно отношение «часть-целое», т.е. агрегация. При этом, идя от целого (агрегата), можно прийти к его частям (атрибутам). Агрегация означает физическое или концептуальное вхождение одного объекта в другой.

Классы и объекты выявляются (обнаруживаются, открываются) на этапе OOA и в совокупности составляют ***словарь предметной области***. На этапе OOD могут понадобиться новые классы, объекты и механизмы их взаимодействия для обеспечения поведения, требуемого моделью, которые приходится уже изобретать.

При проведении OOA и OOD необходимо осуществлять декомпозицию области приложения и разрабатываемой информационной системы. Для обеспечения такой декомпозиции в соответствии с объектно-ориентированным подходом используются абстрагирование и иерархия. Однако в настоящее время задача объектно-ориентированной декомпозиции и выделения классов и объектов не имеет законченного формального решения. «Объектно-ориентированная методика позволяет значительно повысить качество и продуктивность разработки программного обеспечения. Однако извлечь из нее выгоду можно только тогда, когда правильно определено множество объектов. Подходящее множество объектов для конкретной области приложения обеспечивает повторное применение системы и возможности ее расширения, а также гарантирует качество и продуктивность потенциальных улучшений, присущих объектно-ориентированной парадигме. Без формальных методов определения объектов разработчики программного обеспечения рискуют остаться просто хакерами на объектном уровне» [Йордан, 1999, с. 23].

При этом в литературе по объектно-ориентированному подходу отмечается, что «к сожалению, пока не разработаны строгие методы классификации и нет правила, позволяющего выделять классы и объекты. … Как и во многих технических дисциплинах, выбор классов является компромиссным решением» [Буч, 2010]. Существующие методики OOA на сегодняшний день предлагают эвристические правила идентификации классов и объектов, основанные на опыте классификации в других науках. При этом разные авторы предлагают различные наборы базовых классов для объектного моделирования и проектирования. Например, в работе [Шлеер, 1993] предлагается набор: осязаемые предметы, роли, события, взаимодействие; в работе [Ross, 1987] предлагается следующий набор: люди, места, предметы, организации, концепции, события; в работе [Coad, 1990] предлагается такой набор: структуры, другие системы, устройства, события, разыгрываемые роли, места, организационные единицы.

Наличие разных вариантов и отсутствие каких-либо обоснований этих вариантов свидетельствует о том, что, на сегодняшний день, системные методы классифицирования, позволяющие обоснованно выделять классы и объекты, в объектно-ориентированной технологии пока еще не применяются.

2.2.2. Соотношение объектно-ориентированного и системно-структурного подходов

Завершая описание особенностей и понятий системно-структурного и объектно-ориентированного подходов рассмотрим подробнее упомянутую выше проблему ортогональности этих подходов.

Отмеченная выше ортогональность обусловлена тем, что традиционные методы системного (или системно-структурного) подхода и анализа не связаны с выявлением иерархии классов предметной области, т.е. не решают задачи концептуального классификационного моделирования. Таким образом, ***традиционный системный анализ не поддерживает такие обязательные принципы объектного подхода, как абстрагирование и иерархию***.

К сожалению, в литературе по OOAD также, кроме сетований на то, что «рецептов для поиска классов не существует», не предлагается путей решения этой задачи, хотя и утверждается, что это основная задача OOAD [например, Буч, 2010 и Йордан, 1999]. Например, вся литература от Rational Software (под редакцией «трех друзей») ограничивается только рекомендациями общего характера. При этом подчеркивается, что они не знают ни методик построения классификаций, ни даже самого понятия «хорошая классификация». Попытки строить классификации имеются, но они осуществляются на уровне интуиции и экспертных знаний специалистов конкретной предметной области без привлечения какого-либо методического обеспечения.

Кроме того, ***традиционные методы системного (или системно-структурного) анализа не поддерживают объектно-ориентированную концепцию инкапсуляции*** (отделение интерфейса объекта от его реализации). Последнее обстоятельство, в свою очередь, обусловлено тем, что система в традиционном системном анализе и системном подходе (как уже было отмечено выше) всегда рассматривается как множество или элементов, или свойств, или состояний и т.д. В концепции же множества изначально заложена первичность элемента (части) по отношению к множеству (целому). Множество существует тогда и только тогда, когда тем или другим образом заданы его элементы. Теоретико-множественное понимание системы, собственно, и не позволяет обеспечить в ходе традиционного системного анализа инкапсуляцию выявляемых объектов (необходимую при OOAD), так как множество есть явление, внутреннее содержание которого не может быть скрыто. Сущностью же действительно системной концепции («pure system»), на самом деле, является выделение в первую очередь

целого, которое уже потом будет (а может и не будет) представлено в виде совокупности взаимодействующих частей (подсистем). На это давно уже обращали внимание ведущие методологи системных исследований [например, Шрейдер, 1982], однако до сих пор это не нашло практического воплощения в технологиях и инструментах системного анализа.

Необходимо подчеркнуть также, что методы традиционного системного анализа и объектно-ориентированный анализ направлены на выявление различных структур исследуемой системы. В системном (системно-структурном) анализе речь идет о функциональной структуре (вход, процесс, выход, связующие потоки) без внимания к реализующим эти функции объектам (субстанции). В OOAD речь идет о структуре объектов (и классов) системы. Функциональность (поведение) этих объектов рассматривается с точки зрения ответственности компонент ПО, так как объектный подход есть результат развития технологии программирования, а не анализа и моделирования действительности. Потеря при этом способности адекватно отображать соотношения вход-выход и потоковые процессы является его (объектного подхода) большим недостатком, резко снижающим эффективность моделирования реальных (а не программных) явлений [Сахаров, 2000]. Дело в том, что реальная (не виртуальная) действительность целиком и полностью состоит из проточных объектов, постоянно преобразующих входные потоки в выходные. Например, реальный сервер для своего существования должен поддерживаться не только потоками входящей энергии и запасных частей, но и потоками данных, которые кто-то как-то должен периодически туда добавлять, и потоками исполняемого кода, необходимого для обеспечения его функционирования в постоянно меняющихся условиях. Только тогда в ответ на запрос он сможет удовлетворить клиента. Иначе, его просто выключат и выбросят.

Таким образом, сложившаяся в настоящее время (и, безусловно, перспективная) практика разработки сложного ПО средствами объектно-ориентированного подхода, а также отсутствие методов анализа и моделирования предметной области в рамках OOAD привели к возникновению узкого места, требующего создания таких методов. При этом недостатки объектно-ориентированной методологии (с точки зрения моделирования предметной области) и ее ортогональность с существующими методами системно-структурного анализа (способными в принципе компенсировать эти недостатки) позволяют говорить о необходимости синтеза системно-структурного и объектно-ориентированного подходов.

2.3. Системно-объектный подход

2.3.1. Основные понятия и особенности системно-объектного подхода

Одним из вариантов упомянутого выше синтеза подходов является так называемый системно-объектный подход, предложенные в работе [Маторин, 2002, 1]. Понятийный аппарат системно-объектного подхода представляет собой совокупность системологических понятий, заимствованных из работ [Мельников, 1978; 1989; Шрейдер, 1982; Бреховских, 1986; Бондаренко, 1998] переработанных и дополненных авторами. Рассмотрим эти понятия.

Система – функциональный объект, функция которого обусловлена функцией объекта более высокого яруса (функцией ***надсистемы***).

Функция системы – роль, предназначение системы в надсистеме, которая проявляется в наличии ***функциональных связей*** рассматриваемой системы.

Функциональные связи системы – связи с другими системами в конкретной надсистеме.

Поддерживающие связи системы – функциональные связи ее подсистем. Они создают структуру системы. При этом подсистемы находятся в узлах этой структуры и поддерживают функционирование (***функциональные связи***) системы.

Подсистема – функциональный объект нижнего яруса иерархии системы, составляющий вместе с другими объектами ее ***субстанцию***.

Таким образом, функции подсистем обусловлены функцией системы, функция системы – функцией надсистемы и т.д.

Внешняя детерминанта системы (функциональный запрос надсистемы) – явление обуславливания функции системы функцией надсистемы.

Внутренняя детерминанта системы – в действительности проявляемая общая функция системы (ее функционирование). Эта детерминанта определяет функции подсистем (частные функции системы) и их взаимосвязи, т.е. субстанциальные и структурные характеристики системы.

Адаптация системы к запросу надсистемы – приближение внутренней детерминанты системы к ее внешней детерминанте. Критерием адаптированности является отношение между возможностями системы и функциональными требованиями надсистемы. Совершенной или оптимально адаптированной является система, у которой внутренняя детерминанта полностью соответствует внешней.

Свойство системы (или ***валентность***) определяется как способность поддерживать (при определенных условиях) связи одних видов и препятствовать осуществлению связей других видов. Любая же ***связь*** рассматривается как проявление между системами процесса обмена (т.е. потока) элементами, представляющими собой субстанции определенных глубинных ярусов связанных систем. При этом принято говорить об ***экстенциальных*** валентностях, если связи в действительности существуют, и об ***интенциальных*** (или ***потенциальных***) валентностях, если связей в действительности нет. В первом случае способности системы проявлены как реально существующие свойства-связи, во втором случае свойства системы не проявлены, но они существуют как способности (возможности).

Таким образом, ***функциональное свойство*** системы – есть свойство, которым обязательно должна обладать система для выполнения своих функций, т.е. способность поддерживать связи (потоки), на основе которых протекают важные для надсистемы взаимодействия системы с окрестностными системами. ***Поддерживающее свойство*** системы – свойство, необходимое для поддержания и обеспечения устойчивости функциональных свойств, т.е. способность поддерживать связи (потоки), служащие средством внутреннего поддержания, стабилизации функциональных свойств (связей).

В зависимости от пути проявления целостности, как основного признака системности, рассматривается два вида систем: ***внутренние*** и ***внешние***. См. выше параграф 1.3.

Соотношение внутренних и внешних систем соответствует соотношению категорий *явление* и *сущность*. Внутренняя система представляет собой ***систему-явление***, в которой обнаруживается, проявляется (экстенциально выражается) некоторая внешняя надсистема. Эта внешняя надсистема есть ***система-класс*** (***система-сущность***), в которой свойство данной системы-явления представлено интенциально или потенциально, то есть как способность. Таким образом, внутренние системы могут рассматриваться как проявления внешних систем, которые сами по себе остаются непроявленными (так сущность не может быть проявлена иначе, чем через явление).

Следовательно, понятие внешней системы (системы-класса) оказывается более общим по сравнению с понятием внутренней системы (системы-явления) и внутренняя система, будучи конкретным представителем некоторого класса, может рассматриваться как внешняя система единичного объема.

Как видно из представленного описания в концепции системно-объектного подхода четко и однозначно определяются отношение «часть-целое» или «целое-часть» (для систем-явлений) и отношение «род-вид» (для систем-классов). Оба эти отношения сводятся к специфическому системному отношению, не сводимому к отношению между множествами и не описываемому теоретико-множественными средствами. Оно называется ***отношением поддержания функциональной способности целого***. Совершенно очевидно, таким образом, что рассмотренная системная концепция задаёт вполне определённые, конкретные возможности для проведения операций анализа или синтеза ***систем как функциональных объектов*** в упомянутом выше смысле, т.е. является конструктивной.

Рассмотрим подробно процесс формирования (становления) системы, так как при его анализе понятийный аппарат данной системно-объектной концепции проявляется достаточно полно.

Предпосылкой начала процесса формирования системы, согласно данной концепции, является возникновение противоречия между функционированием надсистемы и поддерживающими надсистему функциями ее систем, т.е. противоречия между функциональными и поддерживающими потоками (свойствами) надсистемы. Это противоречие представляет собой нарушение баланса потоков связей (экстенциальных валентностей) в соответствующем узле надсистемы, когда возникают свободные интенциальные валентности окрестностных систем, а формирующийся новый ***узел*** оказывается ***вакантным***. С точки зрения системы, которая начнет формироваться из-за возникновения данного противоречия, оно (возникшее противоречие) представляет собой функциональный запрос надсистемы (т.е. внешнюю детерминанту системы). Этот запрос, представленный в виде вакантного узла надсистемы, определяет потребность надсистемы, т.е. родившуюся в ней необходимость в системе с данной функцией. Таким образом, запрос (внешняя детерминанта) задает ***область требуемых функциональных состояний*** для формирующейся системы через интенциальные валентности (связи) окрестностных систем.

В соответствии с внешней детерминантой системы, задающей область требуемых функциональных состояний, из резерва (набора систем) выбирается некоторая система как ***исходный материал***. Эта система обладает ***областью возможных состояний***, характеризующей ее предрасположенность (интенцию/потенцию) к выполнению определенных (в данном случае требуемых надсистемой) функций. В результате фактического попадания исходного материала в вакантный узел

надсистемы, необходимость превращается в возможность, потоки замыкаются, интенции превращаются в экстенции. Таким образом, система начинает функционировать в соответствии с запросом. При этом исходный материал превращается в субстанцию надсистемы, что и определяет процесс формирования системы с данными функциональными свойствами для поддержания функционирования надсистемы.

Фактическое функционирование системы в ранее вакантном узле надсистемы (новая внутренняя детерминанта системы) становится причиной возникновения противоречия между функциональными и поддерживающими потоками уже внутри системы. Это противоречие становится причиной формирования подсистем с определенными поддерживающими систему функциями и т.д.

В результате описанного процесса система адаптируется к функциональному узлу надсистемы. Процесс ***адаптации*** системы к запросу надсистемы, таким образом, как заключительная фаза становления системы, начинается с того момента, когда данная система в качестве исходного материала помещается в соответствующий вакантный функциональный узел надсистемы. До начала адаптации, когда данная система еще является исходным материалом, внутренние поддерживающие свойства (потоки) данной системы имеют потенции (или интенции) к поддержанию требуемых функциональных свойств (потоков), что и способствует выбору именно данной системы в качестве исходного материала. Но, в то же самое время, внутренние свойства (потоки) данной системы, как явления, потенциально и даже экстенциально могут и поддерживают множество других, в данном случае не требуемых, функциональных свойств. Это и обеспечивает ширину области возможных состояний системы (исходного материала), достаточную для включения (охватывания) области требуемых функциональных состояний вакантного узла, т.е. – определенную избыточность свойств до начала адаптации.

В ходе адаптации данной системы к конкретному функциональному запросу под воздействием ее внутренней детерминанты внутренние свойства (потоки) системы, поддерживающие требуемую функцию, будут превращаться из интенций/потенций в экстенции, а поддерживающие, в данном случае не нужные, функциональные свойства, наоборот, – из экстенций в потенции и далее в интенции. Таким образом, в результате адаптации уменьшается избыточность свойств системы, все сильнее проявляются ее существенные для данной надсистемы свойства. Система из исходного материала, потенциально пригодного для выполнения заданной функции, превращается во все более

совершенную субстанцию данной надсистемы, все более соответствующую запросу.

Чем глубже адаптирована система к функциональному запросу надсистемы, тем ярче проявляются ее существенные для надсистемы свойства, тем выше степень сформированности её сущности, тем меньше избыточность её свойств. Система, у которой область возможных состояний в результате адаптации к запросу надсистемы, не просто покрывает, а максимально близка к области требуемых функциональных состояний, называется ***оптимально адаптированной*** или ***совершенной***. Такие системы представляют собой четко сформированные, ярко проявляющиеся, вполне устойчивые явления определенной сущности. При этом процесс адаптации системы к функциональному запросу, который периодически сам изменяется, рассматривается в системологии как ***эволюция*** системы.

Одним их примеров системы – функционального адаптивного объекта – может служить сотрудник (специалист) как конкретное «должностное лицо» исполняющий свои функциональные обязанности по занимаемой должности, в конкретном отделе (надсистеме) научно-исследовательской организации («наднадсистеме»). В этом случае можно проследить все, отмеченные выше, этапы формирования системы и ее взаимодействия с надсистемой.

В частности, постановка новой, не предусматривавшийся ранее, задачи отделу со стороны организации приводит к рассогласованию между объемом и видом выполняемых отделом работ, с одной стороны, а также штатным расписанием (структурой) отдела и функциональными обязанностями сотрудников, рассчитанными на прежний круг и объём задач, с другой. Следствием этого рассогласования может быть появление вакантного узла («вакансии») в структуре отдела, прежде всего, содержательно, а затем и формально путем введения в штатное расписание новой должности с требуемыми функциональными обязанностями.

Совершенно очевидно, что на эту должность будут подбираться сотрудники (из некоторого множества кандидатов как резерва исходного материала, например – выпускников вузов) потенциально пригодные для выполнения требуемых функциональных обязанностей, т.е. такие, у которых область их возможных состояний покрывает область требуемых функциональных. Совершенно очевидно также, что после того как кандидат приступает к исполнению служебных обязанностей по данной должности, т.е. становится сотрудником отдела (субстанцией), его потенциальные возможности с течением времени становятся всё более и более

соответствующими предъявляемым требованиям. Таким образом, происходит формирование вполне устойчивого в данных условиях явления определенной профессиональной сущности, т.е. специалиста.

2.3.2. Детерминантный анализ как составная часть системно-объектного подхода

Успешное усвоение базовых понятий системно-объектного подхода (системологии) обеспечивает овладение более сложными понятиями этой концепции, непосредственно использующимися при анализе сложных систем. Рассмотрим эти понятия, обеспечивающие специфический метод изучения объектов и процессов, так называемый, *детерминантный анализ* [Мельников, 1978].

Необходимым этапом системного анализа слабо формализованных объектов является анализ причин наличия у объекта (системы) определенных свойств. Детерминантный анализ как раз и позволяет установить причину существования и источник поддержания имеющихся внутренних свойств системы.

> Согласно рассматриваемой системной концепции, главное, всеми поддерживаемое собственно функциональное свойство целого, по отношению к которому все остальные свойства целого и его компонентов оказываются лишь косвенно функциональными, поддерживающими это главное свойство изнутри, становится определяющим, детерминирующим свойством целого. Оно (это свойство) в данной системной концепции названо *внутренней детерминантой* функционального объекта [там же].

Смена этого детерминирующего свойства, что может потребоваться, например, при смене функции уже существующего целого или при смене условий, в которых должно функционировать целое, неизбежно скажется на всех других (поддерживающих) свойствах системы, ибо они должны стать такими, чтобы в своей совокупности и во взаимодействии поддерживать изменившуюся детерминанту, то есть новое детерминирующее свойство целого.

Следовательно, если мы от констатации лежащих на поверхности свойств целого хотим перейти к пониманию их обязательности или факультативности, к выявлению их взаимной подчиненности и их роли при обеспечении функционирования этого целого, то знание детерминирующего (и поэтому всегда собственно функционального) свойства функционального объекта оказывается для нас чрезвычайно важным.

Очевидно также, что поскольку в любой части, в любом элементе функционального объекта мы можем установить поддерживающие и поддерживаемые свойства, то и у любого компонента мы сможем обнаружить свойство, старшее по рангу, поддерживаемое всеми другими свойствами этого компонента. Следовательно, понятие внутренней детерминанты не теряет своего значения, если мы изучаем не только функциональный объект как целое, но и любую его часть. Такие детерминанты частей, вплоть до элементарных компонентов, функционального объекта удобно называть частными по отношению к внутренней детерминанте целого как общей детерминанте.

Использование понятия «внутренняя детерминанта» и рассмотрение ее основных черт позволяет сделать некоторые выводы полезные для теоретического и практического использования системно-объектного подхода.

Если косвенно функциональные, поддерживающие свойства оказывают друг на друга такое взаимовлияние, что укрепляют друг друга и, тем самым, гарантируют высокую устойчивость и стабильность собственно функциональных свойств объекта (системы), то есть стабильность его внутренней детерминанты, то этот функциональный объект, даже будучи вырванным из тех условий, в которых он функционировал, долго остается носителем все той же детерминанты и, следовательно, потенциально способным более, чем другие объекты, выполнять данную функцию.

Кроме того, если система глубоко адаптирована и, следовательно, иерархически самое старшее из поддерживаемых свойств системы, то есть ее общая внутренняя детерминанта, весьма устойчива благодаря тому, что частные детерминанты компонентов системы на всех ярусах являются средством поддержания внутренней детерминанты системы как целого, то для объяснения многих особенностей ее структуры и субстанции, а также состава ее компонентов, их свойств и детерминант, нередко достаточно знания внутренней детерминанты системы как целого. Таким образом ясно, что система, будучи вырванной из надсистемы, может рассматриваться как имманентный объект, т.е. объект имеющий внутренне присущие ему свойства, в отвлечении от того факта, что сформирован он в недрах определенной надсистемы в связи с возникновением в ней потребности наличия такого объекта, который выполнял бы в надсистеме определенную функцию.

Анализ внутренней детерминанты объекта (системы) позволяет раскрыть причины наличия у объекта его внутренних свойств. Однако, знания только внутренней детерминанты недостаточно для анализа и

понимания свойств объекта в целом. Для этого необходимо знание причины наличия у объекта самой внутренней детерминанты.

Если исследователь системы поставит перед собой вопрос, почему у этой системы сформировалась именно такая, а не какая-либо иная внутренняя детерминанта, то в этом случае уже нельзя будет обойтись без осмысления того, по отношению к какой надсистеме изучаемая система является функциональным объектом. Ведь как следует из принятого определения системы, ее главное, иерархически старшее, поддерживаемое всеми остальными свойствами, детерминирующее свойство закреплялось в процессе формирования системы потому, что именно оно способно лучше, чем какие-либо другие, обеспечить выполнение системой той ее функции, которая необходима, чтобы надсистема усилила свою устойчивость.

Следовательно, если будет найден ответ на вопрос, какова функция системы, почему при этой функции именно данное свойство системы как целого необходимо в первую очередь, то это будет одновременно ответ на вопрос, чем определяется, детерминируется извне определяющее, детерминирующее внутреннее свойство системы. Так будет установлена детерминанта внутренней детерминанты системы. И если определяемую детерминанту как главную, общую характеристику системы мы назвали внутренней, то детерминанту, определяющую выбор внутренней детерминанты, естественно назвать *внешней детерминантой* [Мельников, 1978].

Детерминанта системы, понимаемая не как внутренняя, а как внешняя, чаще всего соотносится с представлениями о *функциональном запросе* надсистемы на определенные виды взаимодействия системы с другими системами этой надсистемы [там же].

Если при изучении системы удалось сначала установить внешнюю детерминанту, то внутренняя выводится из нее на основе содержательных рассуждений об этапах формирования системы. Однако, если внешняя детерминанта еще не вскрыта, но внутренняя, после анализа иерархических отношений между свойствами исследуемой системы, уже установлена, то и на этом уровне изученности удается осуществить глубокий системный анализ интересующего нас объекта, ибо состав компонентов, их свойств и их детерминант, состав функционально важных видов связей между компонентами и, наконец, структуры этих связей – все это в свете детерминантного рассмотрения предстает как почти непрерывная цепь очевидных причин и следствий.

В качестве примеров игнорирования понятий «внешняя детерминанта» и «внутренняя детерминанта» и, следовательно, детерминантного анализа при принятии решений, вызвавших серьезные проблемы, можно привести следующие исторические факты.

1). 12 февраля 1958 года китайский лидер Мао Цзэдун подписал исторический указ об уничтожении в стране всех крыс, мух, комаров и воробьев.

Идея запуска масштабной кампании, ставшей частью политической программы «Большой скачок», родилась 18 февраля 1957 года на очередном съезде Коммунистической партии Китая. Ее инициатором выступил, как ни странно, биолог Чжоу Цзянь, являвшийся в то время заместителем министра образования страны. Он был убежден, что массовое уничтожение воробьев и крыс приведет к невиданному расцвету сельского хозяйства. Мол, китайцы никак не могут побороть голод потому, что их «объедают прямо на полях прожорливые воробьи». Чжоу Цзянь убедил членов партии в том, что в свое время Фридрих Великий якобы проводил подобную кампанию, и ее результаты оказались весьма вдохновляющими.

Мао Цзэдуна особо убеждать не пришлось. Свое детство он провел в деревне и не понаслышке знал об извечном противостоянии крестьян и вредителей. Указ был им радостно подписан, и вскоре по всей стране китайцы с лозунгами «Да здравствует великий Мао» ринулись уничтожать обозначенных в указе своего лидера мелких представителей фауны. С мухами, комарами и крысами как-то сразу не заладилось. Крысы, приспособленные для выживания в любых условиях вплоть до ядерной зимы, никак не хотели истребляться полностью. Мухи и комары и вовсе вроде бы не заметили объявленной им войны. «Козлами отпущения» стали воробьи.

Поначалу птиц пробовали травить и отлавливать силками. Но такие методы оказались малоэффективными. Тогда воробьев решили «брать измором». Завидев птиц, любой китаец их старался пугать, заставляя как можно дольше находиться в воздухе. Старики, школьники, дети, мужчины, женщины с утра до ночи размахивали тряпками, стучали в кастрюли, орали, свистели, вынуждая обезумевших птиц порхать от одного китайца к другому. Метод оказался действенным. Воробьи просто не могли находиться в воздухе дольше 15 минут. Изможденные, они падали на землю, после чего их добивали и складировали в огромные кучи. Понятно, что под удар попали не только воробьи, но все мелкие птицы в принципе. Чтобы вдохновить и без того полных энтузиазма китайцев, в прессе регулярно публиковались

фотографии многометровых гор из трупов птиц. Обычной практикой было снять с уроков учеников школ, выдать им рогатки и отправить отстреливать любых мелких птах, разорять их гнезда. Особо отличившимся школьникам выдавали грамоты.

Только за первые три дня кампании в Пекине и Шанхае уничтожили почти миллион птиц. А почти за год таких активных действий лишились двух миллиардов воробьев и прочих мелких пернатых. Китайцы ликовали, праздновали победу. Про крыс, мух и комаров к тому моменту уже никто и не вспоминал. На них махнули рукой, поскольку бороться с ними чрезвычайно сложно. Уничтожать воробьев оказалось гораздо веселее.

Особых противников этой кампании ни среди ученых, ни среди экологов не наблюдалось. Оно и понятно: протест и возражения, даже самые робкие, были бы восприняты как антипартийщина.

К концу 1958 года птиц в Китае практически не осталось. Дикторы с телеэкранов рассказывали об этом как о невероятном достижении страны. Китайцы задыхались от гордости. Никто даже не сомневался в правильности действий партии и своих собственных.

В 1959 в «бескрылом» Китае уродился небывалый урожай. Даже скептики, если таковые имелись, вынуждены были признать, что антиворобьиные меры принесли положительные плоды. Конечно, все заметили, что всевозможных гусениц, саранчи, тли и прочих вредителей заметно прибавилось, но учитывая объемы урожая, все это казалось незначительными издержками. Оценить эти издержки в полной мере китайцы смогли спустя еще один год.

В 1960 году сельскохозяйственные вредители расплодились в таком объеме, что за ними сложно было разглядеть и понять, какую именно сельхозкультуру они пожирают в данный момент. Китайцы были растеряны. Теперь целые школы и производства опять снимали с работы и учебы – уже для того, чтобы собирать гусениц. Но все эти меры были абсолютно бесполезны. Никак численно не регулируемые естественным путем (чем как раз раньше занимались мелкие птицы), насекомые размножались ужасающими темпами. Они быстро сожрали весь урожай и принялись за уничтожение лесов. Саранча и гусеницы пировали, а в стране начался голод. С экранов телевизоров китайцев пытались «кормить рассказами» о том, что все это временные трудности и скоро все наладится. Но обещаниями сыт не будешь. Голод был нешуточным – люди массово умирали. Ели кожаные вещи, ту же саранчу, а кто-то и вовсе питался согражданами. В стране началась паника. Запаниковали и члены партии. По самым скромным подсчетам,

от навалившегося на страну голода в Китае погибло около 30 миллионов человек. Тогда руководство, наконец, вспомнило, что все беды начались с истребления воробьев. За помощью Китай обратился к Советскому Союзу и Канаде – просили срочно выслать им птиц. Советские и канадские руководители, конечно, удивились, но на призыв откликнулись. Воробьев доставляли в Китай целыми вагонами. Теперь уже начали пировать птицы – нигде больше в мире не было такой кормовой базы, как невероятные популяции насекомых, буквально покрывших Китай. С тех пор в Китае особенно трепетное отношение к воробьям.

(Подробно см. https://news.mail.ru/society/32525057/?frommail=10)

2). «Кроличья проблема» в Австралии является классическим примером необдуманного вмешательства человека в уникальную экосистему и его грандиозных последствий. Обыкновенный европейский кролик стал настоящим бичом целого континента.

Принято считать, что эта история началась в 1859 году, когда австралийский фермер Томас Остин выпустил в свой парк несколько кроликов. Это произошло в штате Виктория, район Джилонга. До этого кролики были завезены в Австралию первыми колонистами в качестве источника мяса и обычно содержались в садках. Томас Остин был заядлым охотником и решил, что особого ущерба кролики не принесут, станут отличным источником мяса и на них можно будет с удовольствием охотиться в дикой природе. По другим источникам, выпуски или побеги кроликов в дикую природу отмечались неоднократно в середине 19 века на юге и севере континента, так что винить одного Томаса Остина в распространении кроликов не стоит.

Идея была хорошая. Кролики очень быстро воспроизводятся, имеют вкусное диетическое мясо и достаточно ценные шкурки (кроличий пух), что для первых поселенцев было немаловажно. До этого кролики были достаточно успешно завезены в США и Южную Америку, где никаких проблем с ними не возникло – они включились в экосистемы и их численность контролировали естественные хищники этих мест. Но Австралия – континент особенный, поэтому все пошло не так.

Проблемы начались уже через несколько лет. Численность кроликов сильно выросла и их стали встречать уже за 100 км от места первоначального выпуска. Никто не учел тот факт, что кролики размножаются в геометрической прогрессии: одна крольчиха может произвести в год 20–40 крольчат, и через год общее семейство увеличивается уже до 350 особей. Поскольку в Австралии нет холодных зим, кролики стали плодиться почти круглый год. Хороший климат, обилие корма и

отсутствие естественных хищников явились прекрасными условиями для взрывного роста популяции. К началу 20 века численность кроликов составляла приблизительно 20 млн, а к середине столетия – уже 50 млн. На одного жителя Австралии приходилось по 75–80 кроликов.

С кроликами стали бороться как с врагами овец. Зверьки выедали все пастбища, и овцам не хватало корма. Приводятся такие цифры: 10 кроликов съедает столько же травы, сколько 1 овца, но мяса овца дает в 3 раза больше. Думается, что местных жителей мало заботили проблемы сохранения флоры и фауны, а ведь кролики наносили урон не только овцам и фермерам. Там, где обитали кролики, до 1900 года погибли несколько видов кенгуру (им не хватало корма), серьезно пострадали другие мелкие сумчатые животные, а также некоторые виды аборигенной фауны – кролики выедали растения с корнем и обгладывали молодые деревца, уничтожая их полностью. В результате обыкновенный европейский кролик стал типичным представителем инвазивного вида животных – так называют живые организмы, которые в результате интродукции в новые экосистемы начинают активно захватывать их и вытеснять коренных обитателей.

Сама борьба с кроликами принесла немало бед для Австралийской флоры и фауны. Первоначально решили завезти естественных врагов кроликов – лисиц, хорьков, кошек, горностаев, ласок. Но попытка не увенчалась успехом. Привезенные виды также стали инвазивными, переключившись на местных сумчатых животных и птиц, которые были не так быстры, как кролики, и не могли сопротивляться новым хищникам.

Тогда обратились к традиционным методам – ядохимикаты, отстрел, взрывание нор. Это было не эффективно, учитывая огромную численность животных. В штате Западная Австралия в период с 1901 по 1907 гг. построили огромный проволочный забор. Он так и называется – «Забор от кроликов №1». Забор постоянно патрулируется на машинах, кроличьи подкопы засыпаются, кролики отстреливаются. Сначала забор патрулировали на верблюдах. После появления автомашин, верблюдов за ненадобностью выпустили на волю, они расплодились, стали уничтожать пастбища, и в Австралии появилась новая проблема.

В середине 50-х гг. 20 века для борьбы с кроликами стали использовать достижения медицины. В Австралию привезли кроличьих блох и комаров, зараженных вирусом миксоматоза. Это заболевание вызывает опухоли и смерть кроликов. Таким образом было уничтожено около 90% заболевших животных. Но оставшиеся кролики выработали иммунитет, с течением времени стали реже заболевать и ещё реже

умирать. Так что на данный момент проблема кроликов в Австралии до сих пор не решена.

(Подробно см. https://www.kakprosto.ru/kak-858429-pochemu-v-avstralii-byla-problema-s-krolikami)

Приведенные факты свидетельствуют о важности проведения детерминантного анализа до принятия ответственных решений.

2.3.3. Соотношение системно-объектного и объектно-ориентированного подходов

Традиционные методологии разработки программных систем ориентированы на описание данных и процедур, но не на описание сущностей реального мира и их поведение. Поскольку инжиниринг бизнеса ориентирован на реальные бизнес-структуры и бизнес-процессы, а не на формальные данные и процедуры, традиционные подходы информационного обеспечения бизнеса оказались неадекватными. Объектно-ориентированный подход является подходом, позволяющим описывать как сущности, так и их реальное поведение. Кроме того, он обеспечивает создание прозрачных, легко модифицируемых моделей бизнеса и ИС, допускающих повторное использование отдельных компонент. Именно поэтому, в настоящее время, объектно-ориентированная методология анализа, моделирования и разработки ИС является базовой методологией создания программного инструментария для реинжиниринга бизнес-процессов [Реинжиниринг …].

Так как объектно-ориентированный подход оказывается ортогональным системно-структурному (что было показано выше), актуальным становится сравнение основных понятий объектно-ориентированной методологи и системно-объектного подхода.

Главным предметом исследования системно-объектного подхода является «***система***», объектно-ориентированная методология опирается на понятие «***объект***». Концептуальное сходство системно-объектного и объектно-ориентированного подходов (ООП) основывается, в первую очередь, на сходном понимании природы ***системы*** и ***объекта***, соответственно.

С точки зрения системно-объектного подхода, как уже было отмечено, ***система*** – есть функциональный объект, функция которого обусловлена функцией объекта более высокого яруса, т.е. надсистемой; а ***объект***, с точки зрения ООП, представляет собой предмет или сущность, имеющую определенное функциональное назначение в данной предметной области [Буч, 2010].

При этом, вроде бы за небольшим внешним текстуальным сходством, определений системы и объекта скрывается их глубокое концептуальное единство, состоящее в том, что оба понятия не определяются в терминах теории множеств и не рассматриваются как разновидности понятия «множество». Рассмотрение же системы как множества, свойственное традиционному системному подходу, приводит к невозможности поддержать средствами соответствующего системного анализа инкапсуляцию, являющуюся неотъемлемой составной частью ООП и обеспечивающую разделение внешней и внутренней стороны объекта.

Объектно-ориентированная методология в значительной степени опирается также на следующие основные понятия [Буч, 2010; Вендров]:

- Дихотомию «***класс/объект***», обеспечивающую представление разрабатываемой системы в «канонической форме», т.е. в виде двух ортогональных иерархий – иерархии классов и иерархии объектов (экземпляров классов).
- Дихотомию «***интерфейс/реализация***», обеспечивающую раздельное рассмотрение поведения и реализующего (поддерживающего) это поведение устройства экземпляра или абстракции.
- Отношение «***клиент – сервер***», определяющее функциональные роли элементов и компонент предметной области и разрабатываемой системы.
- Понятие «***ответственности***», связанное с контрактным проектированием и выражающее предназначение и место объекта или класса в системе.

Анализ этих понятий и результатов их сравнения с понятиями системно-объектного подхода позволяет выявить глубокую аналогию основных положений ООП и подхода системно-объектного.

С точки зрения дихотомии «***класс/объект***» системно-объектный подход рассматривает два вида систем: ***системы-классы*** и ***системы-явления*** (называемые в [Шрейдер, 1982] внешними и внутренними системами соответственно).

При этом в настоящее время удалось обеспечить единство содержательного и формального рассмотрения обоих видов систем. Рассмотрение систем-явлений предметной области позволяет оценить её с точки зрения целостности, устойчивости функционирования, глубины и оптимальности адаптации. Рассмотрение систем-классов предметной области позволяет оценить её с точки зрения естественности (онтологичности) и функционального соответствия объективным запросам систем более высокого порядка [Бондаренко, 1996].

С точки зрения дихотомии «***интерфейс/реализация***» системно-объектный подход рассматривает два вида свойств: ***функциональные и поддерживающие***.

С точки зрения отношения «***клиент – сервер***» (в данном случае удобнее сказать «сервер – клиент»), при котором сервер своими ресурсами и услугами (своим функционированием) поддерживает функционирование клиента, в системно-объектном подходе рассматривается ***отношение поддержания функциональной способности целого***. Данное отношение представляет собой отношение система – надсистема, обеспечивающее приобретение надсистемой функциональной способности, поддерживаемой системами и несводимой к способностям систем. Противоположным ему (соответствующим именно отношению «клиент – сервер») является ***отношение детерминирования свойств частей свойствами целого***.

С точки зрения контрактного проектирования и, в частности, понятия «***ответственности***», которое выражает предназначение объекта в системе через совокупность его контрактных обязательств, в системно-объектном подходе рассматриваются понятия ***внутренней детерминанты*** и ***внешней детерминанты***, которая и определяет выбор внутренней.

Кроме того, требование исследования иерархии объектов и иерархии классов в ходе объектно-ориентированного анализа системы находится в полном согласии и соответствии с требованиями системо-объектного подхода, который предполагает необходимость ***детерминантного анализа*** внутренних систем предметной области (систем-явлений или экземпляров) и внешних систем (систем-классов). Следовательно, системно-объектный подход позволяет проанализировать полную и целостную архитектуру рассматриваемой системы в соответствии с требованиями методологии OOAD.

Связь между системно-объектным и объектно-ориентированным подходами представлена в таблице 1.3.

При этом рассмотрение внутреннего и внешнего аспекта системности характерно именно для системно-объектного подхода и ни в какой другой концепции системного подхода не используется.

Сходство представленных понятий имеет большое значение, как для системно-объектного подхода – с точки зрения выявления еще одной важной области его применения: объектно-ориентированного проектирования современного ПО; так и для фундаментального теоретического обоснования и развития самой объектно-ориентированной методологии.

Таблица 1.3

Объектно-ориентированный и системно-объектный подходы

Объектно-ориентированный подход	Системно-объектный подход
Объект	Система-явление (внутренняя система)
Класс	Система-класс (внешняя система)
Объектно-ориентированная декомпозиция	Системная декомпозиция, на основе отношения поддержания функциональной способности целого.
Структура (иерархия) объектов	Партитивная (цело-частная) классификация
Структура (иерархия) классов	Таксономическая (родовидовая) классификация
Контрактное программирование (клиент-сервер)	Внешняя детерминанта (запрос надсистемы) – Внутренняя детерминанта (функция системы, соответствующая запросу)
Варианты использования (прецеденты)	Внутренняя детерминанта
Главная задача объектно-ориентированного проектирования: выбор правильного набора абстракций (классов)	Метод построения классификаций, отражающих существенные свойства предметной области: системологический классификационный анализ

Объектно-ориентированное мировоззрение, при котором «требования к системе воспринимаются с точки зрения классов и объектов, выявленных в предметной области» [Буч, 2010], и системно-объектного мировоззрение, при котором предметная область рассматривается как иерархия функциональных объектов (систем-явлений), находящихся в отношении поддержания функциональной способности целого [Мельников, 1978], а также как иерархия систем-классов, находящихся в том же отношении [Бондаренко, 1996], таким образом, совершенно сходятся. Можно даже утверждать, что такие составные части объектно-ориентированной методологии как OOA и OOD есть, по сути своей, ни что иное, как изложение системно-объектного подхода в терминах программной инжинирии.

2.3.4. Синтез системного и объектно-ориентированного подходов

Представленная выше совокупность понятий системно-объектного подхода и показанное их сходство с понятиями объектно-ориентированного подхода позволяют устранить отмеченное выше противоречие между системным (системно-структурным) и объектно-ориентированным подходами. Рассмотрим это подробно.

В основе решения проблемы синтеза системного и объектно-ориентированного подходов лежит теоретическая концепция, которая обеспечивает интеграцию функциональной (процедурной) и объектной (субстанциальной) декомпозиции системы с учетом ее взаимодействия со средой. Для обеспечения единства системного и объектного подходов необходимо, в первую очередь, преодолеть различия этих подходов с точки зрения направленности процесса декомпозиции. Как было отмечено выше, системному (системно-структурному) подходу (анализу) свойственна так называемая ***процедурная*** (функциональная) декомпозиция системы, а объектному подходу – ***объектная***. При этом они рассматриваются как ортогональные всеми специалистами, как по системному анализу, так и по объектному подходу.

Рассмотрим вариант интеграции функциональной и объектной декомпозиции системы средствами представленного понятийного аппарата системно-объектного подхода.

Решение поставленного вопроса следует начать с обсуждения понятия «структурная декомпозиция», которое часто используется специалистами аналитиками, вероятно, ввиду того, что некоторые методы и подходы традиционного системного анализа именуются «структурными технологиями» (например, SADT). Дело в том, что любой метод анализа и проектирования всегда будет структурным. Всегда в процессе анализа и моделирования выявляется и моделируется некоторая структура. Однако, структура (сама по себе) есть чистая абстракция, что делает ее, конечно, удобным предметом формального исследования, особенно средствами классической математики, которая ни для чего более, собственно, и не предназначена. Но выявления структуры «самой по себе» недостаточно при проведении содержательного (читай системного) анализа сложных, например, организационных, систем. Вот почему формализация процедур системного анализа средствами классической математики резко снижает выразительные возможности системных методов. Например, до сих пор так никто и не знает, как могут быть применены для практических нужд системного анализа известные и очень математические рассуждения по поводу систем (в теоретико-множественном их понимании) в книге [Месарович, 1978], авторы которой и сами не делают даже малейшего намека на такую возможность.

В действительности, структура всегда является структурой чего-либо и вариантов существует не так уж много: это или функции (процессы), или объекты. Таким образом, вопрос состоит только в том, о какой структуре идет речь. В традиционном системном анализе речь

идет о функциональной структуре (структуре процессов) без внимания к реализующим эти функции объектам, т.е. субстанции системы. В объектно-ориентированном подходе речь идет о структуре объектов (субстанции) системы, функциональность которых рассматривается только с точки зрения ответственности компонент ПО.

Настоящие системные (системно-объектные) представления о реальной действительности позволяют объединить процессы выявления и функциональной, и объектной структуры. Согласно данным представлениям, как было отмечено выше, во-первых, система всегда есть ***функциональный объект***, функционирование которого, с одной стороны, поддерживает надсистему, а, с другой стороны, само поддерживается функционированием подсистем. Во-вторых, свойства системы есть внутренние способности этой системы поддерживать связи некоторого вида и/или препятствовать осуществлению связей какого-либо вида, т.е. ***характеризуются связями с другими системами***. Любая же связь между системами есть ***процесс обмена*** между ними элементами определенных глубинных ярусов связанных систем. Таким образом, свойства системы представляют собой проявления ее активности включаться в связи, в ***обменные потоки*** с другими системами в структуре надсистемы.

Следовательно, анализ средствами системно-объектного подхода свойств системы как целостного функционального объекта «основывается прежде всего на обнаружении тех потоков, в которые он включен как элемент надобъекта, т.е. как «проточный» элемент в сети замкнутых обменных потоков надобъекта. Естественно, что обнаружение этих качеств будет одновременно и достаточно полной характеристикой функций этого объекта, и выразителем его целостности, ибо в качественных характеристиках не может не проявиться в этом случае балансность втекающих и вытекающих потоков» [Мельников, 1978, с. 43].

Описанные представления позволяют говорить о том, что структура, в действительности (при системном, конечно, ее рассмотрении) не может быть или только функциональной, или только объектной (субстанциальной). Это на самом деле одна и та же структура, а их разделение есть результат произвола мышления аналитика, ограниченного рамками привычки.

Во-первых, каждая система характеризуется определенными видами связей с другими системами. Если связи отсутствуют, то данную систему рассматривать вообще не имеет никакого смысла. При этом, с точки зрения других систем, любая конкретная система представляется перекрестком, т.е. ***узлом***, связей, по которым что-либо поступает к ней

(«втекает») от других и что-либо поступает от нее («вытекает») к другим. Таким образом, необходимо учитывать, что любая система обязательно является и потребителем каких-то видов ресурсов (материальных и информационных) от других систем, и поставщиком каких-то видов ресурсов для других систем. Качественная узловая характеристика системы является основной и характеризует ее целостно как элемент (подсистему) системы более высокого яруса, т.е. надсистемы. Можно сказать, что узел входящих и выходящих связей характеризует миссию данной системы в структуре надсистемы, так как он определяет, что и от кого поступало к системе и что и кому поступало от нее, т.е. характеризует предназначение системы.

Во-вторых, с точки зрения втекающих и вытекающих потоков/связей, каждая система характеризуется функциональными способностями (процессами, ***функциями***), обеспечивающими преобразование «втекающих» по связям ресурсов в «вытекающие» ресурсы. Эти функциональные способности (процессы) обеспечивают баланс «притока» и «оттока» по функциональным связям узла, занимаемого данной системой. При этом баланс одного и того же узла может быть обеспечен, в принципе, разными наборами функциональных способностей (наборами процессов), т.е. разными функциональными зависимостями выхода от входа. Формальная функциональная характеристика системы характеризует потенциальную возможность системы сбалансировать определенный узел.

В-третьих, с точки зрения функциональных способностей балансировать определенный узел, каждая система характеризуется как ***объект***, реализующий эти функциональные способности (функциональные зависимости), т.е. физически осуществляющий эти процессы. При этом один и тот же набор функциональных способностей может быть реализован, в принципе, различными по своей природе и конструкции объектами. Необходимо только, чтобы производительности этих объектов по входу и выходу соответствовали количественным характеристикам втекающих и вытекающих потоков объектов, связанных с данной системой. Количественная объектная характеристика системы характеризует практическую действительную ее способность сбалансировать определенный узел.

Данные рассуждения приводят к необходимости единовременного представления любой системы с трех точек зрения [Маторин, 2002]:

- как ***структурного элемента*** надсистемы в виде перекрестка связей с другими системами – ***узла***;

- как ***функционального элемента,*** выполняющего определенную роль с точки зрения поддержания надсистемы путем балансирования данного узла – ***функции***;
- как ***субстанциального элемента***, реализующего данную функцию в виде некоторого образования, обладающего конструктивными, эксплуатационными и т.д. характеристиками – ***объекта***.

Следовательно, разбиение системы на подсистемы, представляющие собой трехэлементные конструкции (см. рис. 1.3) «***Узел – Функция – Объект***» (***УФО-элементы***), обеспечит единство функциональной и объектной декомпозиций, так как является наиболее адекватным реальной действительности способом представления структуры, состава и функциональности системы, с учетом её взаимодействия со средой. При этом данное представление системы является конкретизацией определения системы, представленного выше.

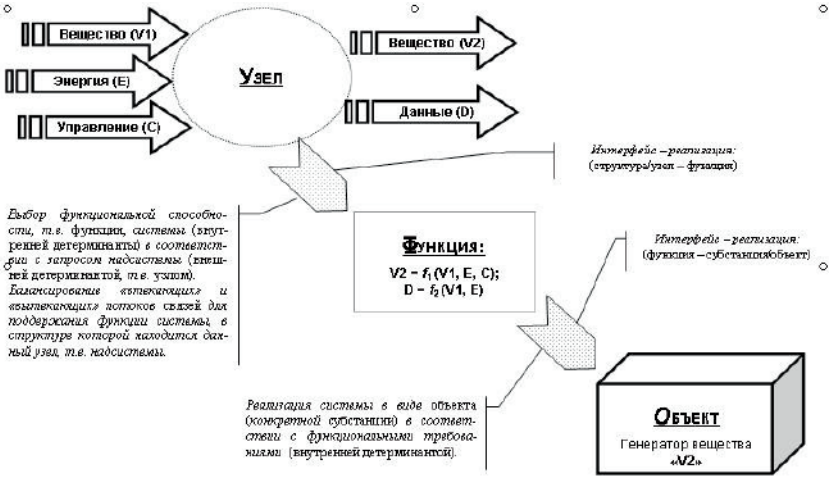


Рис. 1.3. Система как элемент «Узел-Функция-Объект»

Подход «Узел – Функция – Объект» (***УФО-подход***) позволяет рассматривать любую систему или предметную область как совокупность взаимодействующих УФО-элементов (как ***УФО-конфигурацию***), так как любое явление действительности (см., например, таблицу 1.4) представляет собой структурную часть еще более целого (взаимодействует с другими явлениями); функционирует определенным образом и при этом является каким-то объектом. Описанный подход позволяет обеспечить функциональное и объектное моделирование одновременно, т.е. в одной модели, в ходе системно-объектного ***УФО-анализа***.

Таблица 1.4

Аспекты подхода Узел-Функция-Объект

Узел	Интенция	Причина	Потреность	Мотив	Требования (Задание)
Функция	Потенция	Условие	Возможность	Алиби	Проектирование
Объект	Экстенция	Следствие	Деятельность	Веществ. док-ва	Реализация

В процессе моделирования систем в терминах УФО-подхода модельные УФО-элементы, собранные в различные конфигурации, образуют диаграммы взаимодействия элементов, которые позволяют визуализировать функциональность элементов системы более высоких уровней. Таким образом, моделируемая система представляется в виде иерархии УФО-элементов. Данное представление позволяет учесть различные аспекты рассмотрения этой системы (структурные, функциональные, субстанциальные) в одной системно-объектной модели – УФО-модели.

Для агрегации модели системы из готовых составных частей или для декомпозиции модели системы на составные части определены следующие правила комбинирования УФО-элементами (именуемые ***правилами системной композиции***), которые являются следствием упомянутого выше системологического отношения поддержания функциональной способности целого:

1. *Присоединения*: элементы должны присоединяться друг к другу в соответствии с качественными характеристиками присущих им связей (потоков).

2. *Баланса*: при присоединении элементов друг к другу (в соответствии с первым правилом) должен обеспечиваться баланс «притока» и «оттока» по входящим и выходящим функциональным связям (потокам).

3. *Реализации*: при присоединении элементов друг к другу (в соответствии с первым и вторым правилами) должно быть обеспечено соответствие интерфейсов и количественных объектных характеристик функциональным.

4. *Замкнутости*: внутренний поток, не связанный так или иначе с «проточными» потоками от входа системы к ее выходу, замкнут т.е. образует цикл.

В целях автоматизации применения УФО–подхода спроектирован и реализован CASE-инструментарий UFO-toolkit (*http://www.ufo-*

toolkit.ru). Иерархия УФО-элементов и их конфигураций, которую поддерживает UFO-toolkit, основана на классификации связей (потоков), пересечения которых и образуют узлы. Моделирование любой системы начинается со специализации базовой классификации связей под конкретную предметную область. Абстрактный класс «Связь (**L**)» в базовой классификации связей делится на подклассы «Материальная связь (**M**)» и «Информационная связь (**I**)»; класс материальных связей делится на подклассы «Вещественная связь (**V**)» и «Энергетическая связь (**E**)», класс информационных связей – на подклассы «Связь по данным (**D**)» и «Управляющая связь (**C**)». Здесь и далее для обозначения вещественных связей используется символ **V** в связи с тем, что символ **S** (substance) зарезервирован для обозначения систем.

Данные правила позволяют собирать из УФО-элементов модели различной сложности, которые называются УФО-конфигурациями. При этом инструментарий УФО-подхода (см. далее) позволяет определять более правильные конфигурации и рационализировать, например, процедуру организационного проектирования бизнес-системы.

2.3.5. Связь понятий системно-объектного подхода с понятиями теории организации, логистики и инжиниринга бизнеса

Существенной полезной особенностью системно-объектного подхода является его адекватность теории организации, логистике и инжинирингу бизнеса, о чем свидетельствуют результаты сравнительного анализа понятийного аппарата данного подхода и упомянутых методологий теоретического освоения и практического использования организационных систем. Для сравнения выберем конкретные понятия теории организации, логистики и инжиниринга бизнеса.

Результаты сопоставления понятий, законов и принципов теории организации с понятиями системно-объектного подхода представлены в таблице 1.5.

Таблица 1.5

Теория организации и системно-объектный подход

Теория организации	Системно-объектный подход
Организационная система	Функциональный объект, функция которого обусловлена функцией объекта более высокого порядка
Миссия организации	Запрос надсистемы (внешняя детерминанта системы) в виде вакантного узла надсистемы
Потенциал организации	Область возможных функциональных состояний системы (потенциальная валентность)

Продолжение Таблицы 1.5.

Теория организации	Системно-объектный подход
Общий потенциал организации	Общая функция системы (внутренняя детерминанта системы); функциональные связи (свойства)
Частный потенциал организации	Частная функция системы (функция подсистемы); поддерживающая связь (свойство)
Закон синергии	Принцип системности (системный эффект)
Закон приоритета целого над частью	Отношение поддержания функциональной способности целого
Принцип приоритета цели над функцией	Выбор внутренней детерминанты в соответствии с внешней детерминантой
Принцип приоритета функции над элементами и структурой	Формирование внутренних поддерживающих свойств (субстанции и структуры) в соответствии с функциональными свойствами (внутренней детерминантой)
Принцип соответствия между поставленными целями и выделенными ресурсами	Соответствие области возможных функциональных состояний исходного материала области требуемых состояний вакантного узла надсистемы

Результаты сопоставления понятий и принципов логистики с понятиями системно-объектного подхода представлены в таблице 1.6.

Таблица 1.6

Логистика и системно-объектный подход

Логистика	Системно-объектный подход
Логистическая система	Функциональный объект, функция которого обусловлена функцией объекта более высокого порядка
Поток	Связь как проявление процесса обмена элементами глубинных ярусов связанных систем
Запас	Исходный материал
Логистическая цепь	Структура связей системы (поддерживающих)
Логистический процесс	Функциональные (связи) свойства логистической системы
Принцип системности	Отношение поддержания функциональной способности целого
Принцип интегральной оптимальности	Выбор внутренней детерминанты в соответствии с внешней детерминантой
Способность логистической системы к целенаправленному приспосабливающемуся поведению	Формирование внутренних поддерживающих свойств (субстанции и структуры) в соответствии с областью требуемых состояний вакантного узла надсистемы, адаптация

Из таблиц хорошо видно сходство, например, понятий «общий потенциал организации» и «частный потенциал организации» с понятиями «функциональные свойства» и «поддерживающие свойства». При этом, очевидно, что явление или закон синергии представляет собой, по сути дела, ни что иное, как конкретное проявление принципа системности. Очевидно, можно даже утверждать, что организация, в которой не проявляется явление синергии, грубо говоря, вообще не является системой (либо является системой с очень маленькой ***мерой системности*** [Мельников, 1978]). Очень близким, например, к понятию теории организации «миссия организации» является системологическое понятие «внешняя детерминанта (запрос надсистемы)».

Из таблиц также видно сходство, например, логистического понятия «поток» с понятием функциональной системологии «связь», а также то, что логистический принцип системности представляет собой, по сути дела, проявление в логистической системе системологического отношения поддержания функциональной способности целого.

Результаты сопоставления понятий инжиниринга бизнеса с понятиями системно-объектного подхода представлены в таблице 1.7.

Таблица 1.7

Инжиниринг бизнеса и системно-объектный подход

Инжиниринг бизнеса	Системно-объектный подход
Внешняя модель (П-модель) бизнеса	Модель функциональных связей
Внутренняя модель (О-модель) бизнеса	Модель поддерживающих связей
Образ будущей компании (спецификация целей)	Внешняя детерминанта (предельная внутренняя детерминанта) системы
Бизнес-процесс	Формирование из потенциально пригодного исходного материала на глубинном ярусе системы определенной субстанции для обмена по функциональной связи с окрестностной системой
Почему компания делает то, что она делает?	Функциональный запрос надсистемы (детерминантный анализ).
Почему компания делает то, что она делает таким образом?	Текущая внутренняя детерминанта системы (партитивное классифицирование)
Усовершенствование бизнеса	Адаптация системы
Реинжиниринг бизнеса	Эволюция системы

Из таблицы видно сходство, например, мероприятий по определению «почему компания делает то, что она делает» с процедурой определения функционального запроса надсистемы.

Представленные таблицы свидетельствуют о концептуальном сходстве современной теории организации, логистики и инжиниринга бизнеса, а также о том, что эти дисциплины представляют собой, по сути дела, проекцию (конкретизацию) системно-объектного подхода на различные аспекты бизнеса. Это свидетельствует о возможности и целесообразности применения единого системного подхода для решения проблем этих дисциплин.

II. ТЕОРИЯ СИСТЕМ

1. Начала теории систем, основанной на системно-объектном подходе

1.1. Структура и функции научной теории

Несмотря на более чем полувековую историю системных исследований, наличие национальных и международных сообществ, работающих в данном направлении, наличие периодических изданий, конференций, множества публикаций и, наконец, учебных дисциплин по теории систем и системному анализу почти в каждом университете, теория систем (общая или абстрактная) находится, в настоящее время, на этапе формирования своего фундамента. Существующие теоретические построения, как бы они не назывались, не представляют собой полноценной законченной научной теории ни содержательной, ни формальной. Кроме того, как это ни странно, они не обосновывают и не включают в себя как составную часть известные общесистемные принципы и закономерности. Данная ситуация обсуждается уже давно. Результаты этих обсуждений и результаты формирования теории систем можно проследить в многочисленных повторяющих друг друга публикациях (например, [Волкова, 2006; 2015; Прангишвили, 2000]), в том числе, в Интернете (например, см. Википедию и [Безматерных; Дубровский; Мельник]).

Дело в том, что полноценная научная теория, по мнению специалистов науковедения, которые в большинстве своем ссылаются на К. Поппера, должна содержать следующие структурные элементы:

- *Основания теории* – (фундаментальные понятия, принципы, законы, уравнения, аксиомы и т.д.).
- *Идеализированный объект* – (абстрактная модель существенных признаков (свойств и связей) изучаемых объектов действительности).
- *Логика теории* – (совокупность правил и способов доказательства, нацеленных на прояснение структуры знания, на описание его формальных связей и элементов и направленных на исследование и развитие знаний).
- *Совокупность законов и утверждений* – (следствия, выведенные из основных положений теории).

Кроме того, научная теория имеет и должна быть в состоянии выполнять некоторые функции:

- *Синтетическая функция* – (объединение отдельных достоверных знаний в единую, целостную систему).
- *Объяснительная функция* – (выявление причинных и иных зависимостей, многообразия связей данного явления, его существенных характеристик, законов его происхождения и развития, и т.п.).
- *Методологическая функция* – (на базе теории формулируются многообразные методы, способы и приемы исследовательской деятельности).
- *Предсказательная функция, функция предвидения* – (на основании теоретических представлений о «наличном» состоянии известных явлений делаются выводы о существовании неизвестных ранее фактов, объектов или их свойств, связей между явлениями и т.д.).
- *Практическая функция* – (конечное предназначение любой теории – быть воплощенной в практику, быть «руководством к действию» по изменению реальной действительности).

Сложившееся же на сегодняшний день положение с теорией систем таково, что разработчики системных теорий, как правило, продолжают использовать теоретико-множественное определение понятия «система», которое (данное понятие) ни чего общего с теорией множеств, как было показано выше, не имеет. Применяемая, следовательно, при разработке системных теорий системная концепция, фактически не учитывающая все принципы системного подхода, и не позволяет до конца сформировать сами основания теории систем, хотя определенный набор системных понятий, принципов и закономерностей определенно существует.

Этот же теоретико-множественный подход не позволяет сформировать идеализированный объект теории систем, как абстрактную модель существенных признаков изучаемого объекта, т.е. именно системы, так как представление системы в виде множества лишает систему ее основного специфического свойства – целостности.

Искажение основ теории и идеализированного объекта теории нарушают логику теории и не позволяют сформировать совокупность следствий из основных исходных положений, которые еще и сами плохо определены.

Нарушения в структуре системной теории не позволяют ей полноценно выполнять требуемые функции. Немногочисленные объяснительные и методологические возможности, а также практическая

функция (системный анализ) по сути дела, являются возможностями системного подхода, представляющего собой методологическую часть системных исследований вообще.

При этом не теоретико-множественная концепция системы, использующаяся в рамках системно-объектного подхода, учитывает все общесистемные принципы и закономерности и позволяет, таким образом, предложить содержательные и формальные основы действительно системной теории. Рассмотрим это подробнее, уточняя результаты исследований, представленных в работах [Маторин, 2016 – 2018].

1.2. Структурные элементы и основные положения теории, основанной на системно объектном подходе

В *основании теории систем*, основанной на системно-объектном подходе, лежит понимание системы как функционального объекта, функция которого обусловлена функцией объекта более высокого яруса [Мельников, 1978] и совокупность представленных выше понятий этого подхода (связь как поток элементов глубинного яруса связанных систем, система-явление, система-класс, внешняя детерминанта (функциональный запрос надсистемы), внутренняя детерминанта, отношение поддержания функциональной способности целого и т.д.), а также набор известных общесистемных принципов и закономерностей.

Идеализированным объектом теории систем, основанной на системно-объектном подходе, как абстрактной моделью существенных признаков изучаемого объекта (т.е. системы) выступает элемент «Узел-Функция-Объект» (УФО-элемент), моделирующий структурные, функциональные и субстанциальные свойства системы. Формально система как УФО-элемент отдельными аспектами или консолидировано может быть представлена алгебраическими средствами теории паттернов Гренандера, средствами исчисления процессов Милнера или средствами исчисления объектов Абади-Кардели, что будет показано в дальнейшем.

К элементам *логики теории*, в настоящее время, можно отнести упомянутые выше правила системной композиции, классификацию потоков связей, на основании которой в дальнейшем будет предложена классификация систем как УФО-элементов, алгебраические операции с такими элементами по аналогии с операциями упомянутых выше исчислений.

К *совокупности законов и утверждений*, на сегодняшний день, относится обоснование взаимосвязей общесистемных закономерностей и доказательство расширенного понимания принципа моноцентризма Богданова (представлены далее). Кроме того, обоснование системного понимания картины мира, общественной и личной жизни, а также обоснование классификации системных элементов (введение алфавита).

Представленные выше структурные элементы теории систем, основанной на системно-объектном подходе, необходимо дополнить следующими концептуальными положениями данной теории [Маторин, 2017; Маторин, 2018, 1].

Во-первых, система рассматривается как *функциональный объект или класс, функция или роль которого обусловлена функцией или ролью объекта или класса более высокого яруса* (т.е. надсистемы). При этом рассматривается два принципиально различных вида систем: *внутренние системы* (системы-явления) и *внешние системы* (системы-классы).

Во-вторых, любая система обязательно связана с другими системами и эти связи представляют собой *потоки элементов глубинного яруса связанных систем*. При этом связи данной системы с другими системами – *функциональные*, связи между подсистемами данной системы – *поддерживающие*.

В-третьих, упомянутое в определении системы явление обуславливания функции системы функцией надсистемы рассматривается как функциональный запрос надсистемы на систему с определенной функцией (*внешняя детерминанта* системы).

В-четвертых, внешняя детерминанта системы есть причина ее возникновения, цель ее существования и главный определитель ее структурных, функциональных и субстанциальных свойств. Таким образом, она (внешняя детерминанта системы) рассматривается в качестве ***универсального системообразующего фактора***.

В-пятых, функционирование системы под влиянием внешней детерминанты является ее *внутренней детерминантой*, так как непосредственно определяет ее внутренние свойства (структурные, функциональные и субстанциальные свойства подсистем).

В-шестых, функционирование системы в соответствии с внешней детерминантой устанавливает между системой и надсистемой *отношение поддержания функциональной способности более целого*.

В-седьмых, процесс приближения внутренней детерминанты системы к ее внешней детерминанте представляет собой *адаптацию*

системы к запросу надсистемы. При этом отношение внешней и внутренней детерминант системы соответствует отношению области требуемых функциональных состояний (**ОТФС**) системы в соответствии с запросом надсистемы к области возможных функциональных состояний (**ОВС**) исходного материала для требуемой системы. Это отношение понимается как *мера системности* (***M*$_s$**). Максимально адаптированной считается система, у которой ***M*$_s$** близка к 1. При формировании системы с требуемой функцией в узле надсистемы по запросу последней выбирается исходный материал для нужной системы, для которого, естественно, справедливы неравенства: **ОВС > ОТФС** и **0 > *M*$_s$ < 1**. В действительности не существует систем с ***M*$_s$** = 0 или ***M*$_s$** = 1. Адаптация системы к изменяющемуся запросу надсистемы представляет собой *эволюцию* системы.

В-восьмых, следствием упомянутого выше определения системы и понимания связи между системами является представление системы в виде триединой конструкции «Узел-Функция-Объект» (УФО-элемент), где:

- **узел** – структурный элемент надсистемы в виде перекрестка связей данной системы с другими системами;
- **функция** – динамический (функциональный) элемент надсистемы, выполняющий определенную роль с точки зрения поддержания надсистемы путем балансирования связей данного узла;
- **объект** – субстанциальный элемент надсистемы, реализующего данную функцию в виде некоторого материального образования, обладающего конструктивными, эксплуатационными и т.д. характеристиками.

В-девятых, для агрегации системы из составных частей, представляемых в виде УФО-элементов, или для декомпозиции системы на такие составные части определены представленные выше правила комбинирования УФО-элементами, именуемые **правилами системной композиции** (см. пп. 2.3.4).

Данные положения конкретизируют, в частности, представление системы как средства решения проблем [Оптнер, 1969]. Дело в том, что определение узла, в котором должна находится система, есть ни что иное, как конкретизация проблемы, которую должна решать система. Определение же ее функциональности есть, по сути дела, определение средства решения проблемы, характеризующейся узлом.

Кроме того, данные положения полностью соответствуют системному подходу предложенному в работе [Гиг, 1981], уточняя и

конкретизируя его. Системный подход представляется в данной работе как такой «принцип исследования, при котором рассматривается система в целом, а не ее отдельные подсистемы. Его задачей является оптимизация системы в целом, а не улучшение эффективности отдельных подсистем» [там же, с. 28]. Одним из основных положений такого подхода «является отказ от изучения задач системы без рассмотрения ее взаимосвязей с более широкой системой, в которую она входит» [там же, с. 48].

Представленный концептуальный аппарат имеет также ряд общих понятий с *теорией живых систем* Миллера [Miller, 1975]. По крайней мере, он обладает такими же чертами, какие отмечены автором работы [Гиг, 1981] по отношению к теории Миллера. При этом данная теория рассматривается в качестве примера одной из немногих действительно системных теорий. Эти черты состоят в следующем [там же, с. 99–100]:

- Система в целом и ее отдельные части описываются на одном языке.
- Может быть получен исчерпывающий список компонентов системы, причем в списке отражаются и их взаимосвязи, а не только констатируется наличие данного компонента.
- Подход позволяет рассмотреть и оценить все процессы в целом, а не исследовать по частям.

1.3. Обоснование представления системы в виде элемента «узел-функция-объект»

Для обеспечения содержательного описания своей предметной области любая самостоятельная теория должна использовать собственную систему понятий и категорий, т.е. должна иметь особенный, специфический концептуальный аппарат в качестве инструмента своих исследований. Специфичность и взаимосвязанность понятийного аппарата являются одним из показателей эффективности и самостоятельности любой теории, что подтверждается мировым опытом научных исследований, а также данными и выводами теории познания, осуществляющей рефлексию над процессом научного исследования. Методами гносеологии установлено, что использование концептуального аппарата является обязательным и необходимым условием любого исследования. Всякое исследование, тем более связанное с формированием новой теории, должно поэтому начинаться с разработки и уточнения основных, базовых понятий данной теории.

Для обеспечения формального описания предметной области при решении своих задач разрабатываемая системная теория должна представлять собой формальную систему (или исчисление), т.е. использовать алфавит, включающий в себя знаки (символы), применяемые для записи по определенным правилам выражений (формул). При этом принято считать, что эти знаки рассматриваются совершенно формально, без какой бы то ни было содержательной интерпретации. Интерпретацию формальная теория или система может получить (а может и не получить) уже после своего создания [Петров, 1977]. Однако, если термин «знак» использовать в смысле Г. Фреге (а не в смысле Д. Гильберта), то можно утверждать, что любой знак всегда представляет собой единство означающего и означаемого. Это значит, что знаки алфавита формальной системы, на самом деле, не могут не иметь сами по себе смысла, т.е. содержательной интерпретации, иначе они вообще не будут знаками. Действительно, в любой формальной системе имеются знаки, либо относящиеся к абстрактным понятиям очень большого объема (переменные, кванторы, функциональные блоки и т.д.), либо обозначающие конкретные собственно математические или логические операции (дифференцирование, конъюнкция, объединение и т.д.). Последнее обстоятельство и является основной проблемой, затрудняющей описание традиционными формальными средствами специфических системных свойств и отношений ввиду их глубокого и разностороннего содержательного характера. Эта проблема, однако, может быть преодолена за счет использования алфавита, знакам которого заведомо приписывается определенный понятийный смысл по некоторому заранее оговоренному правилу в рамках определенной предметной области.

Дело в том, что, например, на естественном языке удается описывать весьма содержательные объекты (строить содержательные высказывания) в значительной степени благодаря использованию существующей в этом языке исходной совокупности слов, которые имеют смысл (содержание) до построения из них какого-либо высказывания. Таким образом, можно ожидать повышения выразительных возможностей формальной системы (или исчисления), если алфавитные символы этой системы будут заранее иметь определенное содержание (смысл). При этом в некотором подмножестве естественного языка, в так называемом «языке делового общения», в котором слова представляют собой в основном термины, имеющие понятийное содержание, исходная совокупность слов составляет иерархическую систему понятий (терминов), т.е. классификационную структуру [Маторин, 1997, 1 и 2; Matorin, 1998]. При этом данная структура не жесткая, так как слова могут и добавляться, и удаляться из нее.

Таким образом, для того, чтобы некоторая формальная система была не только формальной, но и семантической необходимо задавать (определять) смысл алфавитных символов формальной системы с помощью классификационной схемы. Использование классификации для придания значкам формальной системы уникального предметно-ориентированного содержания превращает эту формальную систему в систему формально-семантическую [Matorin, 2001; Зимовец, 2012, 4]. Для разрабатываемой теории это тем более актуально в связи с тем, что она охватывает оба известных пути проявления системности, т.е. внутренние системы (системы-явления, экземпляры) и внешние системы (системы-классы).

Следовательно, для решения и содержательных, и формальных задач разрабатываемая системная теория должна включать в себя, в первую очередь, классификацию (или концептуальную модель) основных понятий.

Сказанное выше обязывает определять основные понятия разрабатываемой системной теории, с помощью которых предлагается осуществлять построение данной теории, путем развертывания концептуальной классификационной схемы системных компонент и их свойств. В качестве основы (корня) такой классификации предлагается рассматривать категориальную иерархию классов, представленную на рисунке 2.1.

Данная категориальная концептуальная схема иерархии классов основана на предложенной в работе [Маторин, 1997, 1] *семантической модели системы парных категорий* в виде иерархической структуры системы категориальных понятий с одной вершиной, а также опыте построения концептуальных классификационных моделей для различных предметных областей. В результате ее развития и уточнения получена классификационная схема, учитывающая закономерности естественной классификации, обоснованные, например, в работе [Bondarenko, 2001]. Учет данных закономерностей означает представление обычной таксономии в виде *параметрической классификации*. Параметрической же считается классификация, в которой классифицируемые элементы систематизированы в соответствии с классификацией их свойств, являющейся частью общей иерархии классов [Забродин, 1981]. Практически это означает, что в данной иерархии для каждого классифицируемого компонента (как класса или понятия) присутствует не только его родовой признак (вышестоящий класс), но и понятие (класс), являющееся видовым отличием в содержании классифицируемого понятия, т.е. свойство компонента. На рисунке связь понятия с его видовым отличием (класса с его свойством) изображена пунктирной линией.

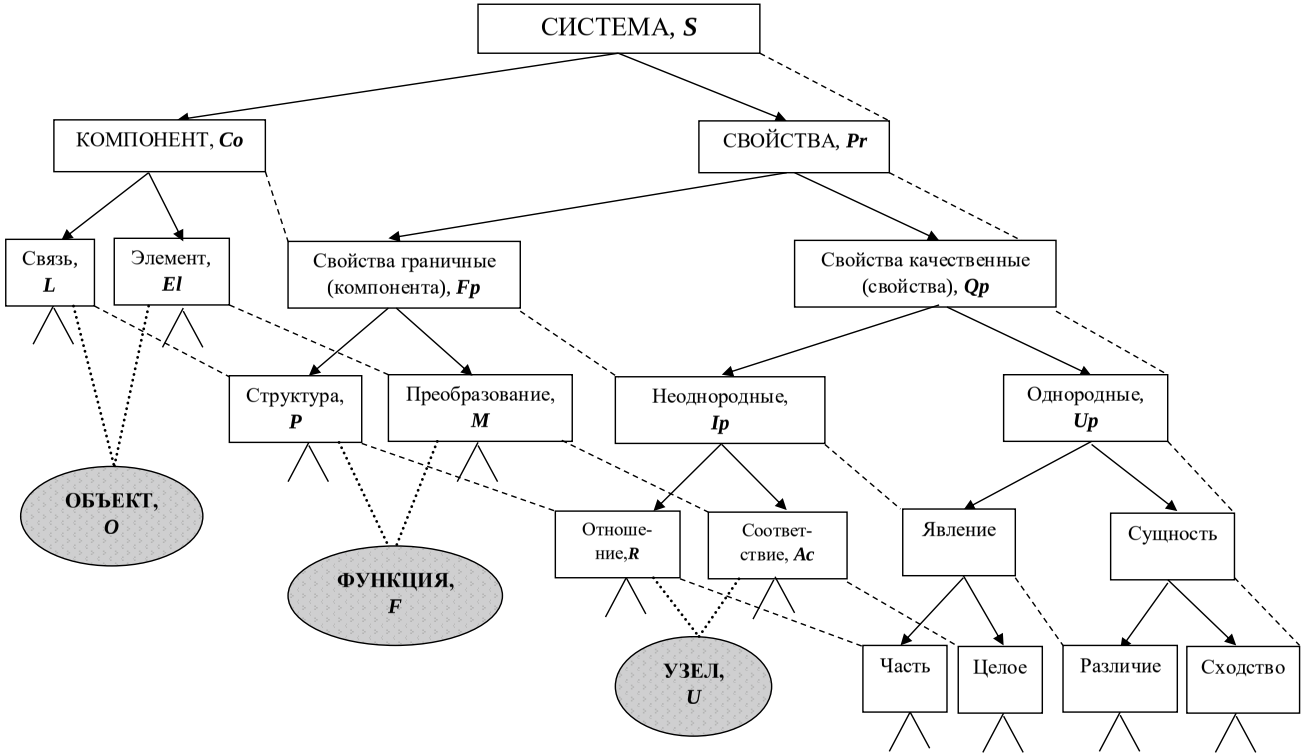


Рис. 2.1. Категориальная иерархия классов

Рассмотрим предлагаемую иерархию подробнее.

В данной иерархии, корневой класс Система (единственная категория) разделен на подклассы Компонент и Свойство. Правомерность подобного рассмотрения обусловлена следующими обстоятельствами. Во-первых, в соответствии с работой [Мельников, 1978], наиболее сильно различающимися разновидностями свойств являются Свойства граничные (характеризующие пространственные и временные ограничения на определенные качественные свойства) и Свойства качественные (выявляющиеся в отвлечении от показателей, которые позволяют обнаруживать пространственно-временные характеристики внешнего мира). Во-вторых, современная логическая наука явления, обладающие границами, определяет, как «вещи» (т.е. носители свойств, у нас: компоненты), а качествами – как «свойства». «Каждая вещь имеет пространственные характеристики, пространственные границы, чего нельзя сказать об отдельно взятом свойстве ... выражение «имеет пространственные границы» неприменимо к особенностям того или иного свойства вещи, благодаря которому одна вещь качественно отличается от другой» [Чупахин, 1973, с. 43]. Кроме того, по мнению ряда философов, в мышлении первоначально образовались категории, отражающие именно свойства и их носители [Диалектика, 1985]. Дополнительно следует отметить, что иерархия классов в популярной программной среде G2, предназначенной для разработки экспертных систем, обладает единственным корневым классом «item-or-value» («сущность и значение») с производными классами «item» и «value» [Попов, 1996]. Внутри класса Компонент выделены подклассы Связь и Элемент на том основании, что, с одной стороны, в качестве главных граничных свойств рассматриваются Структура (как более граничное свойство) и Преобразование (или «отображение», как менее граничное свойство и обобщение понятия «функция»), а, с другой стороны, связи – это как раз то, что определяет структуру системы, а элементы системы – это то, что обеспечивает преобразование связей как потоков в структуре системы.

Применение метода системологического классификационного анализа для построения рассматриваемой иерархии классов приводит, в частности, к тому, что данная классификация, как категориальная концептуальная модель предметной области, позволяет однозначно определять (т.е. знать и понимать, в том числе компьютерной программе) термины, включенные в нее. Дело в том, что в соответствии со структурными свойствами естественной классификации содержание представленных на рисунке терминов (классов) может быть формально логически определено. Таким образом, учет в данной классификационной

схеме закономерностей естественной классификации позволяет получить в ней для каждого компонента однозначное формально логическое определение через род и видовое отличие (см. таблицу).

Таблица 2.1

Определения (дефиниции) понятий категориальной классификации

Понятие:	Родовой признак:	Видовое отличие:
Компонент	Система	Определяющаяся **граничными свойствами**
Свойство	Система	Определяющаяся **качественными свойствами**
Связь	Компонент системы	Определяющий **структуру** системы
Элемент	Компонент системы	Определяющий **преобразование** в системе

Категориальная классификационная структура может быть развернута (специализирована) до более конкретных классов путем разделения на каждом уровне свойств на более граничные (располагающиеся слева) и на менее граничные (располагающиеся справа) в соответствии с принципиальным делением класса **Свойство** на **Св. граничные** и **Св. качественные**. При этом следует иметь ввиду, что теоретически возможно иное деление свойств. Однако, по мнению авторов, по крайне мере на сегодняшний день, не существует других, хоть в какой-то степени, обоснованных вариантов, т.е. вариантов деления класса **Свойство**, обеспечивающих параметричность таксономической классификационной схемы. Предложенная иерархия классов позволяет использовать при решении каждой конкретной задачи свой конкретный набор средств моделирования (элементов и связей, понятных как пользователю, так и компьютеру !!!). Например, для моделирования информационного бизнеса (организационных систем – средств массовой информации) можно разнообразить и уточнить до соответствующих конкретных классов классы информационных связей и элементов, а компоненты, связанные с веществом и энергией оставить в виде абстрактных классов; для моделирования энергетических предприятий необходимо конкретизировать виды энергетических компонент; для моделирования транспортных компаний – классы вещественных связей; для моделирования производства – классы элементов, моделирующие получение вещества соответствующего вида. Незыблемым остается лишь принцип, в соответствии с которым свойства элементов и связей, используемых для создания объектной модели организационной системы, определяются их параметрической таксономической классификацией, т.е. категориальной иерархией классов,

учитывающей эти свойства. Это обеспечивает возможность обоснованной декомпозиции организационной системы (бизнес-системы), и даже возможность передачи, например, компьютерной информационной системе поддержки бизнеса (реинжиниринга бизнеса) формально-логических редакторских и контрольных функций при создании моделей и управлении ими.

Категориальная иерархия классов вместе с заложенными в нее принципами построения представляет собой понятийную (категориальную) «сетку» или «призму», через которую аналитик смотрит на моделируемую предметную область, что и обеспечивает основания для выявления определенных абстракций. Специализированная к заданной предметной области иерархия, кроме того, выполняет роль *библиотеки базовых классов* для решения основной задачи объектно-ориентированного анализа и проектирования, т.е. выявления необходимого набора абстракций предметной области. Кроме того, предложенная иерархия классов, как модель системно-функционального аспекта универсума, представляет собой модель онтологии высокого уровня, в которой представлены и систематизированы знания о реальной действительности. Можно утверждать, что представление и использование таких знаний в полной мере соответствует парадигме интеллектуального знание ориентированного развития компьютерных информационных технологий.

Родовидовые определения, получаемые из категориальной иерархии классов (см. таблицу) можно формализовать в виде кортежа упорядоченной пары классов следующим образом:

$$Co = \langle S, Fp \rangle,$$

где ***S*** – класс **Система** и родовой признак класса ***Co*** **Компонент**; ***Fp*** – класс **Св. граничные** и видовое отличие класса ***Co***. (Здесь и далее буквенные обозначения классов (систем-классов) будем писать курсивом прописными буквами, а экземпляров (систем-явлений) – обычным прямым шрифтом и строчными буквами.)

Таким образом, не имея, при данном подходе, возможности формально определить понятие «система» (как в теории множеств нет формального определения множества), будем формально говорить о компонентах, обладающих системными свойствами. Отмеченное обстоятельство подчеркивает тот факт, что любая система всегда является подсистемой какой-либо суперсистемы или надсистемы. Аналогично можно определить все остальные понятия системных компонент, учтенные в категориальной иерархии классов.

Любое единичное понятие о конкретном объекте (внутренней системе или системе-явлении) как экземпляре класса **Компонент** также может быть определено через родовой класс (конкретный листьевой класс, от которого объект наследует свои свойства) с учетом соответствующей конкретизации свойств. Это объясняется тем, что внутренняя система (система-явление) есть внешняя система или система-класс единичного объема. При этом обязательно будет наблюдаться явление множественного наследования, т.е. конкретный объект будет иметь не одно родовидовое определение (наследовать не от одного класса). Во-первых, он будет наследовать от какого-либо видового (листьевого) класса (классов) **Связь**, так как обязательно будет связан с другими конкретными системами (объектами) в структуре надсистемы. Во-вторых, он будет наследовать от какого-либо видового (листьевого) класса (классов) **Элемент**, так как будет обладать определенной функцией (выполнять определенное преобразование) в надсистеме.

Таким образом, в общем случае, можно формализовать представление о конкретной системе **s** (системе-явлении) как функциональном объекте с помощью упорядоченного набора классов следующим образом:

$$\boldsymbol{s = <L, p> \wedge <El, m> = <L, p, El, m>,}$$

где класс ***L*** представляет конкретный вид или виды связей (листьевой класс, конкретизированный от класса **Связь**), в которых участвует система-явление **s** (т.е. соответствующее множество элементарных компонент, образующих связующие потоки), а **p** – конкретная структура (как явление), заданная на множестве элементарных компонентах, образующих указанный вид связей (связующих потоков); ***El*** – конкретный вид или виды элементов, т.е. листьевой класс, конкретизированный от класса **Элемент**, к которым относится система-явление **s**, а **m** – конкретная операция преобразования (как явление), заданная на множестве элементарных компонент, протекающих через систему **s** по входящим и выходящим связям.

Элементы кортежа, приведенного выше, могут быть определены в соответствии с категориальной иерархией классов следующим образом:

$$\boldsymbol{L = <Co, P>; \; p = <P, r>;}$$

$$\boldsymbol{El = <Co, M>; \; m = <M, ac>,}$$

где класс ***L*** представляет конкретный вид или виды связей, в которых участвует система-явление **s**; ***Co*** – класс системных компонент;

P – класс **Структура**; **p** – конкретная структура (как явление, экземпляр), определенная на множестве элементарных компонентах, образующих указанные виды связей (связующие потоки); **r** – явление (экземпляр) класса **Отношение**; ***El*** – конкретный вид или виды элементов, к которым относится система-явление **s**; ***M*** – класс **Преобразование**; **ac** – конкретный экземпляр класса **Соответствие** как явление, заданное на множестве элементарных компонент, протекающих через систему **s** по входящим и выходящим связям. Таким образом, исходный кортеж может быть приведен к следующему виду:

$$s = <Co, P, r, M, ac>.$$

Учтем теперь, во-первых, что **r** и **ac** являются частями любого конкретного отдельно взятого перекрестка связей (входных и выходных потоков некоторой системы), так как такой перекресток определяет, с одной стороны, некоторое конкретное *соответствие* между входами и выходами системы, но, с другой стороны, как отдельно взятый перекресток, не определяет реальных связей (потоков), а только возможность (причем только интенцию) их установления, что и рассматривается как *отношение*, например в работе [Мельников, 1978]. Таким образом, можно записать, что $r \cup ac = u$. (Корректнее, конечно, $\{r\} \cup \{ac\} = \{u\}$, но в данном случае это не существенно). Во-вторых, ***P*** и ***M*** являются видами частей, обеспечивающих процесс функционирования системы, так как, с одной стороны, ***M*** определяет процесс преобразования, а, с другой стороны, ***P*** задает его структуру, что необходимо и достаточно для задания функции системы. Таким образом, можно записать, что $P \cup M = F$. В-третьих, класс ***Co*** состоит из подклассов ***L*** и ***El***, представляющих собой и его виды. При этом элементы и связи являются составными частями любой конкретной системы, как реально существующего объекта. Примечательно, что такое понимание хорошо согласуется, например, с понятием «образующей» в теории паттернов Гренандера [Гренандер, 1979]. Таким образом, можно записать, что $Co = L \cup El = O$.

Следовательно:

$$s = <u, F, O>.$$

Таким образом, система, как функциональный объект, представляет собой сущность, характеризующуюся конкретным **узлом** (перекрестком связей) в структуре надсистемы, множеством (классом) **функций**, балансирующих данный узел путем преобразования входов в выходы, и множеством (классом) **объектов**, реализующих данные функции [Matorin, 2002].

При этом, очевидно, что данный кортеж соответствует, с одной стороны, системе как конкретному явлению (системе-явлению), так и системе как классу функциональных объектов (системе-классу).

Если ставить задачу создания формализованной дедуктивной теории, то необходимо основываться на определенных аксиомах. Рассмотрим один из возможных вариантов формального представления аксиом разрабатываемой теории, полученный путем аналитического описания категориальной иерархии классов с учетом выведенных понятий узел, функция и объект (см. рисунок 2.1.).

Упомянутая иерархии классов может быть представлена, например, следующей парой аналитических выражений:

$$\exists!\ S = \{O, Pr\}: Pr = \{F, Qp\} \rightarrow O = <S, F> \wedge Pr = <S, Qp>,$$

где представлены следующие классы ***S*** – Система; ***O*** – функциональный **Объект**; ***Pr*** – Свойство; ***F*** – **Функция** (Граничное свойство); ***Qp*** – Качественное свойство. И

$$U \subset Qp \rightarrow F = <Pr, U>,$$

где ***U*** – проточный **Узел**.

Эти выражения означают, что иерархия классов «Узлы–Функции–Объекты» представляет собой результат деления предельно широкого класса (предельно абстрактного понятия!) ***S*** на два класса ***O*** и ***Pr***, один из которых (в данном случае второй класс – ***Pr***) представим в виде двух подклассов ***F*** и ***Qp*** таких, что первый подкласс предельно широкого класса ***O*** является кортежем, состоящим из предельного класса и класса ***F***, а второй подкласс предельно широкого класса ***Pr*** является кортежем из предельного класса и класса ***Qp***. Кроме того, класс ***Qp***, в свою очередь, обладает подклассом ***U*** таким, что класс ***F***, в свою очередь, является кортежем из второго подкласса предельного класса и класса ***U***.

По сути дела, данные выражения, которые можно рассматривать как аксиомы разрабатываемой системной теории, позволяют задать правила для формально-логического (родовидового) определения символов системных объектов и их свойств, т.е. алгоритмическим образом (*конструктивно*) определить их семантику. Наличие алгоритма, позволяющего задавать семантику символов формальной системы, в соответствии с известной классификацией языков и теорий [Петров, 1977] позволяет утверждать, что данная формальная теория (система) должна рассматриваться как *формально-содержательная конструктивная система*.

При этом, исходя из аксиом, видно, что

$$O = <S, F>,$$

т.е. **Объект** есть **Система**, определяющаяся **Функцией**;

$$F = <Pr, U>,$$

т.е. **Функция** есть **Свойство**, определяющееся **Узлом**.

1.4. Алфавит элементов «Узел-Функция-Объект»

Как было отмечено выше в рамках системно-объектного подхода рассматриваются не абстрактные связи **L**, т.е. любые потоки вообще, а связи/потоки, имеющие определенное содержание. Это обусловлено введением в концепцию рассматриваемой системной теории *классификации связей*. В данной классификации абстрактный класс «*Связь* (**L**)» делится на непересекающиеся подклассы «*Материальная связь* (**M**)» и «*Информационная связь* (**I**)»; класс материальных связей делится на непересекающиеся подклассы «*Вещественная связь* (**V**)» и «*Энергетическая связь* (**E**)»; класс информационных связей – на непересекающиеся подклассы «*Связь по данным* (**D**)» и «*Управляющая связь* (**C**)». Данная классификация связей/потоков дополняется более конкретными потоками при описании систем определенной предметной области и является основой создания объектов (УФО-элементов) различных типов. Кроме того, предложенная классификация представляет собой механизм, обеспечивающий специализацию системных понятий и приложение этих понятий к конкретным предметным областям.

Использование классификации связей позволяет классифицировать элементы «Узел-Функция-Объект» (УФО-элементы) по их узлам первоначально на уровне материальных и информационных потоков (см. таблицу 2.2, где знак «**?**» соответствует входным потокам, а знак «**!**» – выходным).

Таблица 2.2

Классификация узлов материя/информация

	M!	**I!**	**M!, I!**
M?$_j$	*		
I?$_j$		*	
M?, I?			*

Данная классификация обусловлена, в первую очередь, тем, что материя не может преобразовываться в информацию, а информация не может преобразоваться в материю (классы материальных и информационных потоков не пересекаются).

Классифицирование УФО-элементы по их узлам на уровне вещественных и энергетических потоков (см. таблицу 2.3) обусловлено тем, что вещество не может преобразовываться в энергию в чистом виде, а энергия не может преобразоваться в вещество (классы вещественных и энергетических потоков не пересекаются).

Таблица 2.3

Классификация узлов вещество/энергия

	V!	**E!**	**V!, E!**
V?	*		
E?		*	
V?, E?			*

Классифицирование УФО-элементы по их узлам на уровне потоков данных и потоков управления (см. таблицу 2.4) обусловлено тем, что данные не могут преобразовываться в управление полностью, а управление не может преобразоваться в данные (классы данных и управленческих потоков не пересекаются).

Таблица 2.4

Классификация узлов данные/управление

	D!	**C!**	**D!, C!**
D?	*		
C?		*	
D?, C?			*

Таблица 2.5

Классификация смешанных узлов

	V!, D!	V!, C!	E!, D!	E!, C!
V?, D?	*			
V?, C?		*		
E?, D?			*	
E?, C?				*

Приведенные выше рассуждения показывают, что не может быть бесконечного разнообразия УФО-элементов, отражающих системность некоторой предметной области на нижнем уровне иерархии. На этом уровне системы должны представлять собой совокупности взаимодействующих «алфавитных» материальных УФО-элементов, представленных на рисунке 2.2, информационных УФО-элементов, представленных на рисунке 2.3 и смешанных – на рисунке 2.4.

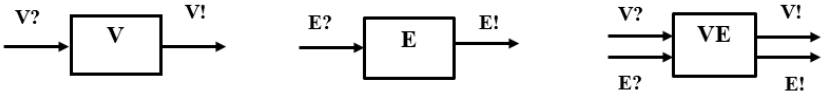


Рис. 2.2. Алфавитные материальные УФО-элементы.

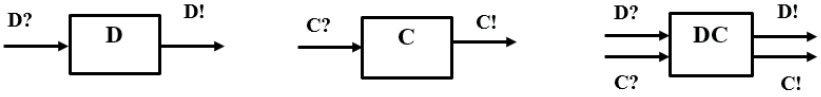


Рис. 2.3. Алфавитные информационные УФО-элементы.

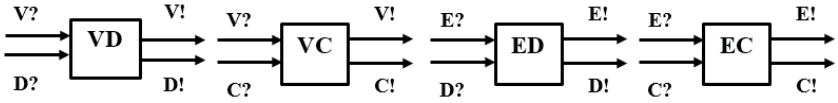


Рис. 2.4. Алфавитные смешанные УФО-элементы.

Таким образом, использование содержательной классификации связей позволяет рассматривать не бесчисленное множество видов систем, а весьма ограниченный их набор. Эти ограничения на возможные

преобразования связей возникают естественным образом в связи с делением связей на непересекающиеся классы, что соответствует основному принципу построения классификаций. При этом, чем конкретнее предметная область, тем конкретней набор алфавитных элементов.

1.5. Учет концептуальных систем средствами системно-объектного подхода

Бурное развитие системных исследований в прошлом столетии сменилось некоторым снижением интереса к данному научному направлению в настоящее время. Данная ситуация обусловлена сложностью создания содержательной действительно системной теории и сложностью ее формализации. Среди прочих недостатков существующих теоретических построений следует отметить, что в них не учитываются различные пути проявления системности. Говоря о системах, как правило, имеют в виду только конкретные объекты, явления. Однако, еще в работах [Ackoff, 1964; Шрейдер, 1982; Дубровский] обоснована необходимость учета в рамках теории систем не только конкретных материальных объектов. Например, в работах [Ackoff, 1964; Дубровский] отмечается, что теория систем не может претендовать на то, что она общая теория, если она не применима к концептуальным системам, а также подчеркивается, что как раз разработка системных принципов, применимых и к материальным, и к концептуальным системам является наиболее актуальной для преодоления пропасти, разделяющей естественные и гуманитарные науки.

При этом в соответствии с [Шрейдер, 1982] в зависимости от пути проявления целостности, как основного признака системности, имеет смысл рассматривать два вида систем: ***внутренние*** (т.е. материальный по Акоффу) и ***внешние*** (т.е. концептуальные по Акоффу). См. параграф 1.3.

Разрабатываемый нами системно-объектный подход как существенный аспект ноосферного этапа развития науки, в целом, и системных исследований, в частности, ориентирован на применение диалектических принципов ***целостности, системности, иерархичности*** и ***развития*** [Гвишиани, 1979]. Рассмотрим особенности проявление этих принципов для различных путей образования систем – внутреннего и внешнего, а также особенности отражения в сознании человека внутренних и внешних систем с точки зрения этих принципов.

Принцип целостности. С точки зрения онтологии внутренняя система проявляет свою целостность всегда как конкретный объект, у которого имеются граничные и качественные свойства [Мельников, 1978],

характеризующиеся пространственно-временной определенностью (единством) и многоаспектностью. Таким образом, внутренняя система названа «системой-явлением» не случайно, а вполне закономерно, так как представляет собой онтологическое воплощение философской категории «явление».

Целостность системы-класса – это всегда целостность не связанного никакими пространственно-временными ограничениями класса объектов, соответствующих определенному (одноаспектному) качественному свойству [Мельников, 1978]. Поэтому внешняя система может быть названа «системой-сущностью», так как представляет собой онтологическое воплощение философской категории «сущность».

С точки зрения гносеологии целостность внутренней системы может и будет отражаться в сознании человека как на уровне восприятия в виде текущего целостного конкретного образа, так и на уровне чувственного отражения (представления) в виде целостного обобщенного образа [Соловьев, 1973]. Целостность же системы-класса, как класса объектов, может быть отражена только на абстрактном уровне в виде понятия [Кондаков, 2012].

Принцип системности. В онтологии и внутренней и внешней системы могут быть выделены подсистемы, находящиеся со своей системой в отношении поддержания функциональной способности целого [Мельников, 1978]. Эта возможность наряду с целостностью собственно и делает их системами. Однако, у внутренней системы, как экземпляра конкретного объекта, подсистемами являются компоненты или элементы, также представляющие собой экземпляры конкретных объектов, но более глубокого яруса. У внешней же системы, как у класса объектов, подсистемами являются подклассы данного класса.

Особо следует обратить внимание на то, что у системы, как функционального объекта, подсистемы поддерживают ее функциональность. Поэтому не любой элемент конкретного объекта или подкласс данного класса объектов представляет собой подсистему данной системы.

Корпус, двигатель и ходовая часть являются подсистемами любого конкретного автомобиля, так как обеспечивают выполнение им его функции. Если же просто разрезать автомобиль на достаточное количество произвольных кусков, то эти элементы автомобиля не будут его подсистемами, так как не существует алгоритма сборки из них функционирующего автомобиля.

Аналогично, класс легковых автомобилей и класс грузовых автомобилей являются подсистемами внешней системы автомобильный транспорт, так как обеспечивают ее функциональную целостность. А класс

синих автомобилей и класс красных автомобилей не имеют отношения к функциональным свойствам автомобильного транспорта, поэтому подсистемами соответствующей внешней системы не являются.

Отношение поддержания функциональной способности целого между внутренней подсистемой и внутренней системой (или системой и надсистемой) отражается в сознании человека как отношение часть-целое. Это же отношение, отражающееся на материале систем-классов (подсистем, надсистем), предстает в сознании человека в виде отношения род-вид. Следовательно, отношения часть-целое и род-вид являются гносеологическими разновидностями существующего в онтологии отношения поддержания функциональной способности целого.

В этой связи необходимо подчеркнуть разницу между функционально-системологическим (*эволюционно-содержательным* [Косарев, 1978]), широко используемым в системно-объектном подходе, и традиционным формально-логическим подходами к отношению род-вид.

Традиционно виду приписываются кроме специфических видовых все свойства рода. Однако, это справедливо только с точки зрения формального подхода, при котором не учитывается содержательная сторона данного отношения. Учет содержательной же его стороны, соответствующей отношению поддержания функциональной способности целого, показывает, что вид (подсистема) не может обладать свойствами рода (системы), так как любая система обладает свойствами, принципиально не сводимыми к свойствам подсистем. Свойства рода являются более общими (абстрактными) по сравнению с видовыми свойствами не только в формальном, но и в содержательном смысле. Это свойства более высокого уровня целостности (функциональности), которые обеспечиваются благодаря наличию у видов частных свойств (частных функций) более низкого уровня иерархии.

Принцип иерархичности. Иерархия внутренних систем образуется за счет физического взаимодействия систем-явлений на каждом ярусе иерархии в пространстве и времени. Связи этих системы возникают благодаря наличию у них экстенциально проявленных свойств, что обеспечивает возможность их восприятия, а также приборного наблюдения. Иерархия же систем-классов есть взаимное соответствие свойств (ролей) систем различных уровней. Причем в данном случае системы связаны между собой интенциально (потенциально) благодаря свойствам, которые остаются непроявленными и принципиально не могут быть наблюдаемы [Пугачев, 1991].

Названные различия в проявлении принципа иерархичности внутренних и внешних систем приводят к различиям процессов и результатов

отражения и познания этих систем. Отражение иерархии внутренних систем имеет образный характер. Это позволяет в качестве результата познавательного процесса иметь партитивную классификацию этих систем или мерономию [Панова, 1975]. Отражение иерархии систем-классов имеет только абстрактный характер. Результат познания этих систем представляет собой родовидовую классификацию или таксономию [Панова, 1975].

Принцип развития. Развитие любых систем, с точки зрения данной системно-объектной концепции, есть процесс постоянного соотнесения и согласования внешней детерминанты системы с ее текущей внутренней детерминантой.

Становление, адаптация и развитие внутренней системы будут отражаться в сознании человека как сокращение избыточности ее свойств и свойств ее подсистем на все более глубоких ярусах иерархии системы, то есть в ее мерономии. Эти же процессы, происходящие с системой-классом, будут отражаться в виде удлинения таксономической цепочки взаимно согласованных подклассов все более глубоких уровней иерархии системы, а также в виде увеличения соответствия свойств их свойств ... и т.д. Последний аспект связан с особенностями таксономической структуры систем-классов.

Анализ особенностей отражения в сознании человека внутренних и внешних систем с точки зрения диалектических принципов показывает, что эти принципы соблюдаются для обоих видов систем. Это, в свою очередь, может служить дополнительным обоснованием правомочности введения систем-классов и рассмотрения их как функциональных объектов. Результаты анализа наглядно представлены на приведенных ниже рисунках (рис. 2.5 – 2.8).

В рамках системно-объектного подхода «Узел-Функция-Объект» изначально представление системы соответствует содержательному определению системы Г.П. Мельниковым как функционального объекта, функция которого обусловлена функцией объекта более высокого яруса (т.е. надсистемы). Очевидно, что это определение ориентировано на системы-явления (материальные системы по Акоффу).

Утверждаемая классиками необходимость и показанная нами возможность рассмотрения классов (концептуальных систем по Акоффу) как таких же систем обязывает уточнить упомянутое определение так, чтобы оно учитывало не только системы-явления, но и системы-классы. Специфика реальных систем-классов и отражаемых в сознании человека знаний о них может быть учтена, например, при представлении системы-класса как класса, роль которого обусловлена ролью класса более высокого яруса.

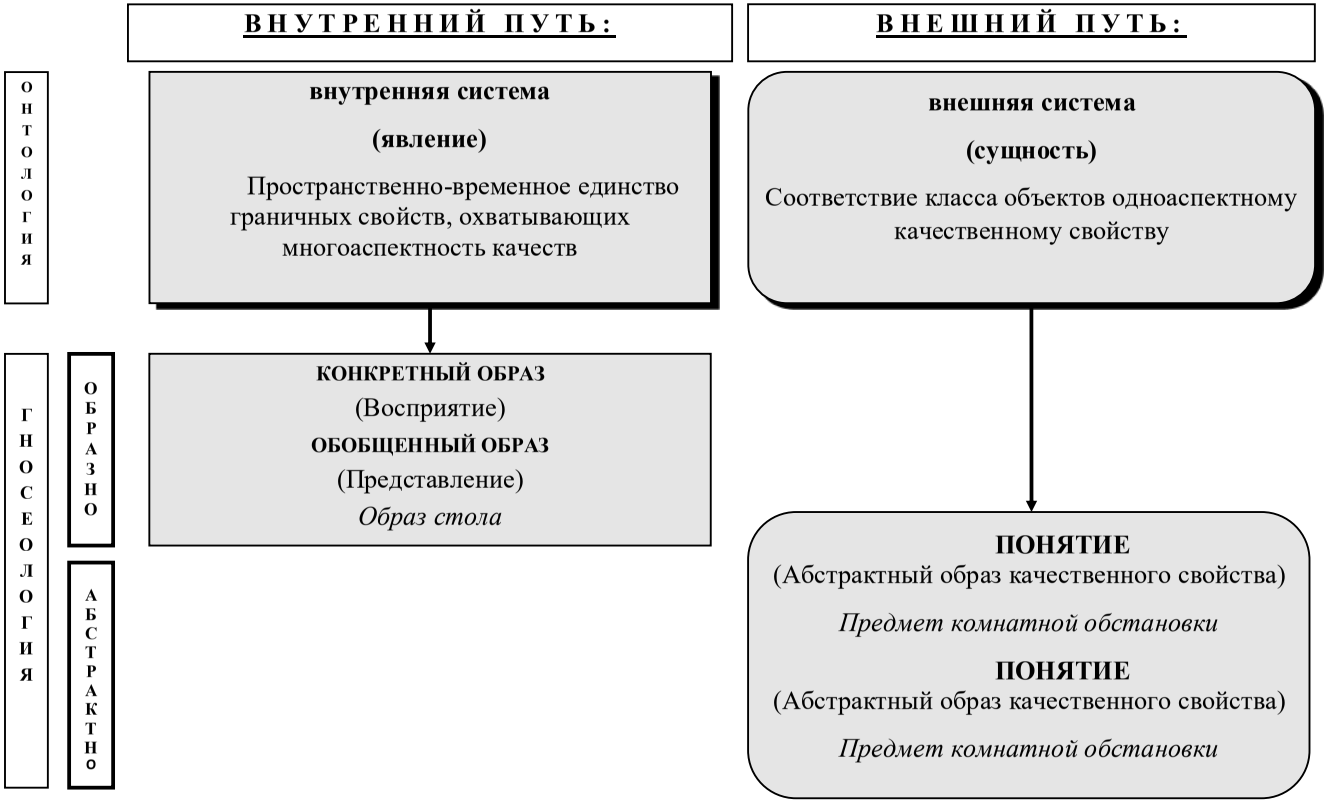


Рис. 2.5. Проявление и отражение принципа целостности

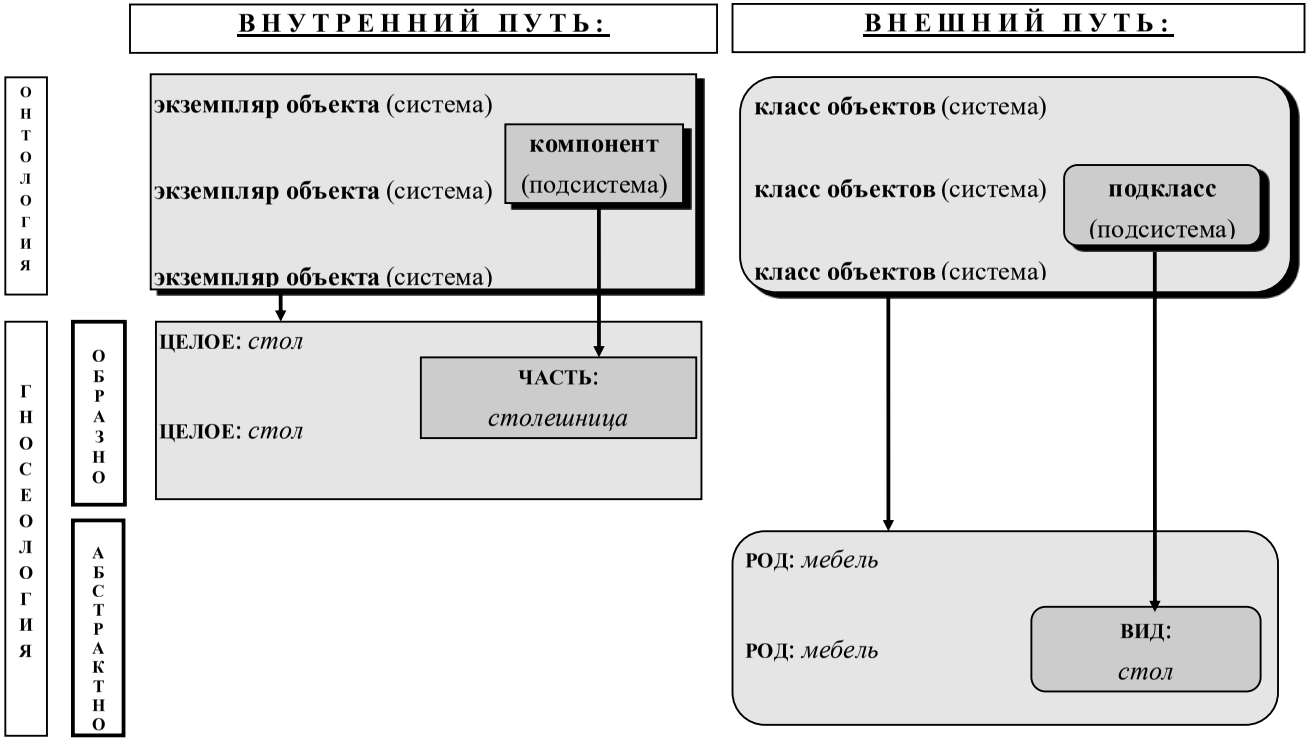


Рис. 2.6. Проявление и отражение принципа системности

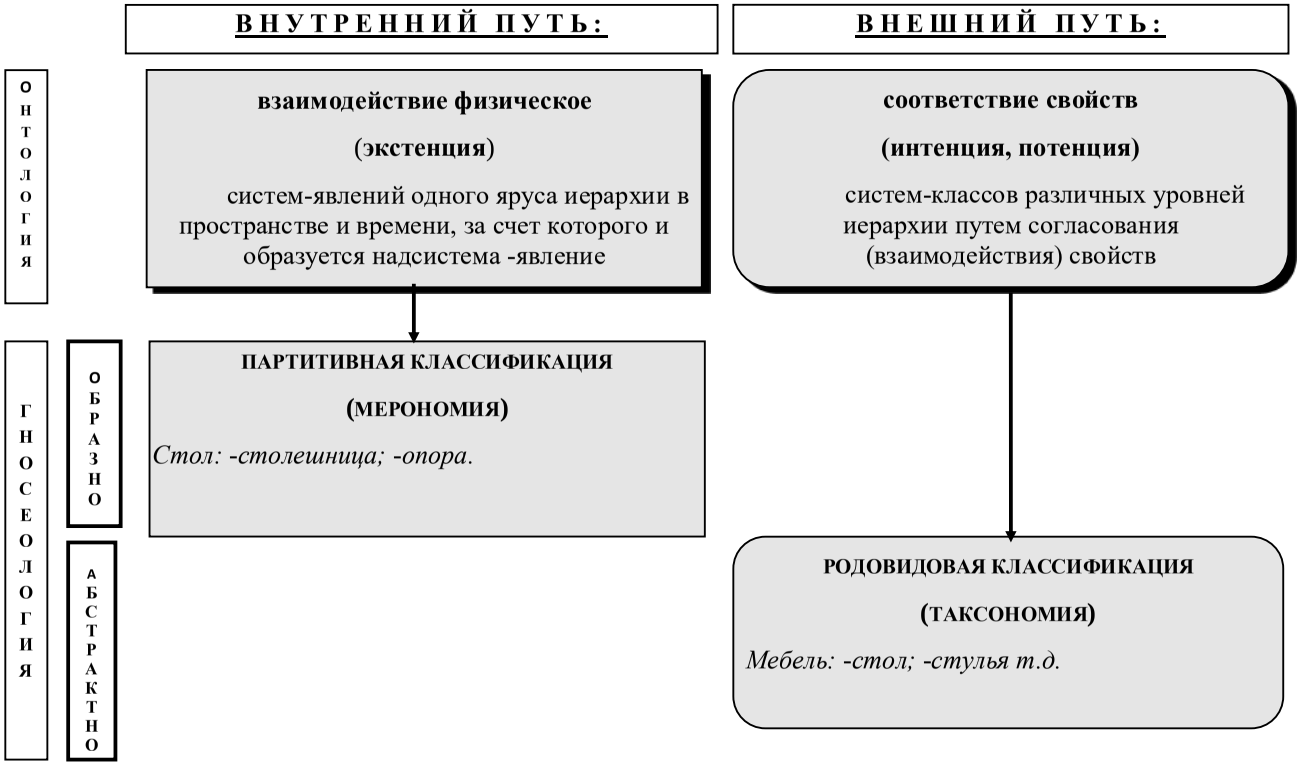


Рис. 2.7. Проявление и отражение принципа иерархичности

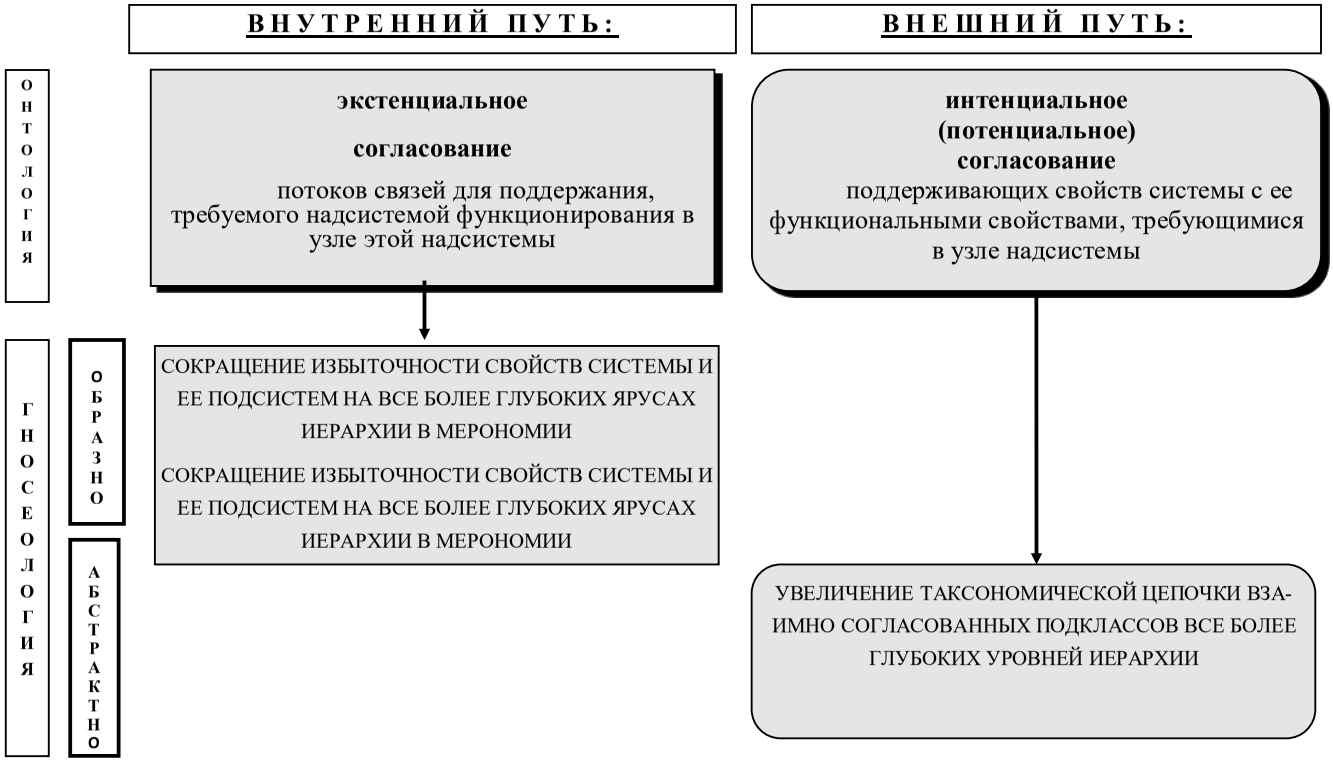


Рис. 2.8. Проявление и отражение принципа развития

Объединяя представления о системе-явлении и системе-классе, приходим к следующему универсальному определению, которое по сути уже было представлено выше:

Система есть явление (функциональный/материальный объект) или класс (концептуальная система), функция или роль которого обусловлены функцией явления или ролью класса более высокого яруса (т.е. надсистемой-явлением или надсистемой-классом).

2. Функциональные возможности теории систем, основанной на системно-объектном подходе

Как было отмечено выше, любая научная теория должна быть в состоянии выполнять некоторые функции. В частности, научная теория должна выполнять:

- Синтетическую функцию, т.е. способность объединять отдельные достоверные знания в единую, целостную систему.
- Объяснительную функцию, т.е. способность выявлять причинные и другие зависимости, многообразия связей объекта исследования данной теории, его существенные характеристики, законы его происхождения и развития, и т.п.
- Предсказательную функцию, т.е. на основании теоретических представлений о «наличном» состоянии известных явлений делать выводы о существовании неизвестных ранее фактов, объектов или их свойств, связей между явлениями и т.д.

Рассматриваемая в данном случае теория, основанная на системно-объектном подходе, выполняя синтетическую функцию, выявляет и обосновывает взаимосвязи общесистемных принципов и закономерностей, объединяя их в единую концептуальную модель.

Выполняя объяснительную функцию, данная теория обосновывает системное понимание картины мира и объясняет причины существования проблем с институтом семьи.

Выполняя предсказательную функцию, данная теория обосновывает, что современная цивилизация находится на этапе очередного эволюционного перехода.

Рассмотрим это подробно.

2.1. Учет и обоснование взаимосвязей общесистемных закономерностей

2.1.1. Учет общесистемных закономерностей системами-явлениями

Обзор современного состояния системных исследований и способов построения системных теорий (см., например, [Прангишвили, 2000]) показывает, что, как правило, эти теории не учитывают и не могут учитывать известные общесистемные закономерности, так как эти теории основаны на теоретико-множественном подходе (обоснование см., например, [Шрейдер, 1982]). При этом сами общесистемные закономерности выводятся авторами из практического опыта и здравого смысла первоначально, как правило, в рамках конкретных наук и предметных областей. Несмотря на приводимые примеры, они остаются не обоснованными, так как не выводятся из какой-либо концепции системного подхода и не описываются единым понятийным аппаратом. Кроме того, имеющие место попытки объединения этих закономерностей в группы не обосновывают их взаимосвязей и взаимозависимостей.

Предложенные же выше концептуальные положения теории систем, основанной на системно-объектном подходе, позволяют с единых системных позиций описать ряд общесистемных принципов и закономерностей [Маторин, 2016, 2 – Маторин, 2018, 1] и обосновать их взаимосвязи [Маторин, 2018, 2]. Исходные определения всей совокупности рассматриваемых закономерностей со ссылками на первоисточники представлены по адресу: https://ru.wikipedia.org/wiki/Общая_теория_систем. Далее при упоминании каждой закономерности в круглых скобках указаны авторы, впервые их описавшие.

1. Представление системы в рамках системно-объектного подхода в виде триединой конструкции «Узел-Функция-Объект» обеспечивает понимание того факта, что узловая (структурная) характеристика системы является основной и характеризует ее целостно как элемент (подсистему) системы более высокого яруса, т.е. надсистемы. Можно сказать, что узел входящих и выходящих связей характеризует миссию данной системы в структуре надсистемы, так как он определяет, что и от кого поступает в систему и что и кому поступает из нее, т.е. характеризует предназначение системы. Таким образом, системно-объектный подход естественным образом учитывает следующие общесистемные закономерности:

- принцип *коммуникативности* (Садовский, Юдин): система связана множеством коммуникаций с окружающей средой;

- принцип *иерархичности* (Берталанфи): любая система представляет собой иерархию ее подсистем и (что самое главное!) система на любом ярусе иерархии является частью системы более высокого яруса, т.е. надсистемы.

При этом последний уточняет принцип структурный коммуникативности. Дело в том, что иерархичность предполагает обязательное наличие связей между системами, т.е. существование некоторой структуры. Принцип иерархичности, таким образом, фиксирует тот факт, что структура связей систем подчинена определенной закономерности.

Названные принципы, по-видимому, являются основополагающими принципами. Дело в том, что они описывают основные структурные характеристики систем. Структурные же характеристики являются, как известно, первичными. Именно со структурными особенностями системы имеет дело в первую очередь наблюдатель при обнаружении системы в среде. Кроме того, функциональный запрос надсистемы на определенную систему (внешняя детерминанта системы, ее системообразующий фактор) представляет собой набор функциональных связей некоторого (первоначально вакантного) узла, т.е. система определяется структурной характеристикой надсистемы. Это и дает нам право рассматривать принципы коммуникативности, иерархичности как основополагающие для всех остальных принципов и закономерностей.

Рассмотрим далее остальные общесистемные закономерности, касающиеся структурных (узловых) свойств систем.

Принцип *моноцентризма* (Богданов): устойчивая система обладает одним центром, а полицентричность приводит к нарушению процессов координации, что в перспективе обуславливает потерю целостности.

С точки зрения системно-объектного подхода, учитывающего как внутренний путь проявления системности, так и внешний, данный принцип может рассматриваться в более широком контексте, что обосновано, например, в работах [Маторин, 1996; Маторин, 2017]: «*Иерархия систем обладает единственной вершиной* (*т.е. она моноцентрична*)». Рассмотрим обоснование этого факта.

Принцип иерархичности для внутренних систем (систем-явлений) позволяет проследить иерархию функциональных запросов (как

причин формирования свойств систем) в виде безграничной цепочки отношений «часть-целое». Этот же принцип для внешних систем (систем-классов), позволяет проследить иерархию функциональных запросов как цепочку отношений «вид-род», которая имеет принципиальное ограничение. Это ограничение связано с тем, что при переходе по иерархии от вида к роду (от подкласса к классу) происходит переход от свойства подсистемы к свойству системы (несводимому к свойствам подсистем) в сторону обобщения, т.е. с сокращением множества признаков (свойств), за счет которых системы-явления относятся к данному классу. Множество же систем-явлений, входящих в класс будет больше, чем множество систем-явлений, входящих в подкласс. Этот процесс соответствует закону обратного отношения объема и содержания понятий, в которых отражаются в нашем сознании системы-классы, т.е. с увеличением объема содержание уменьшается. Для самих систем-классов это соответствует уменьшению набора признаков, за счет которого формируется система-класс, при увеличении множества явлений, относящихся к данному классу. При этом, в связи с конечностью набора признаков, увеличение объема и сокращение числа признаков за конечное число шагов приводит к такому классу, для отнесения к которому не остается признаков, а объем которого становится бесконечно большим. Таким образом, рассматривая реальную действительность как иерархическую структуру систем-классов, можно прийти к выводу, что эта структура имеет единственный верхний узел, то есть, существует единственная Надсистема-класс. Если предположить обратное: надсистема не является единственной и имеется еще хотя бы одна система-класс того же уровня, то эти системы можно считать элементами системы более высокого уровня. Следовательно, Надсистема-класс, включающая в себя все виды систем, является единственной. Сделанные выводы подтверждаются положением о единстве универсума, который рассматриваться как общепринятый, и позволяют рассматривать принцип *моноцентризма* как общесистемный, являющийся следствием выполнения принципа иерархичности, т.е. уточняющим свойства системной иерархии.

Принцип *организационной непрерывности* (Богданов): между всякими двумя системами имеются звенья, вводящие их в одну «цепь ингрессии».

Справедливость данного принципа может быть обоснована только в том случае, если все системы существуют в рамках одной Надсистемы. Но это так и есть, в соответствии с системно-объектным

пониманием принципа моноцентризма. Таким образом, данный принцип, по сути дела, является прямым следствием иерархического моноцентризма. В работе [Маторин, 2018, 2] это доказано формальными средствами.

Принцип *прогрессирующей сегрегации* (Берталанфи): с учетом его уточнения, например, Ж. Делёз, фиксирует прогрессирующую потерю взаимодействия между элементами системы в ходе ее дифференциации при усилении связей с некоторым элементом, выступающим в роли системного центра.

Упомянутый принцип является следствием того факта, что причиной возникновения системы и ее становления является функциональный запрос надсистемы (т.е. внешняя детерминанта системы). Именно надсистема определяет функции своим системам, которые, в свою очередь, определяют функции своих подсистем. В данном случае имеет место ситуация, при которой некоторая часть системы начинает выполнять функцию надсистемы для остальных ее частей, требования которой включают в себя усиление взаимодействия всех с ней и ослабление связей между частями. Таким образом, принцип прогрессирующей сегрегации есть результат проявления принципа иерархичности в условиях дифференциации системы.

При этом надсистема может выставить требования по усилению связей между ее системами, что будет соответствовать принципу *прогрессирующей систематизации*, упоминаемому, например, В.В. Качала.

Принцип *обратной связи* (Эшби): устойчивость в сложных динамических системах достигается за счёт замыкания петель обратных связей.

Принцип *взаимно-дополнительных соотношений* или *комплиментарности* (Богданов): устойчивость системы достигается взаимно-дополнительными связями между её элементами в виде замкнутых контуров обратных связей

Первый принцип уточняет принцип коммуникативности, выделяя из всего многообразия связей определенный их вид. Таким образом, данный принцип является видовым по отношению к принципу коммуникативности.

Второй принцип констатирует факт возникновения обратных связей между элементами системы в случае системной дифференциации

(расхождения частей системы). Таким образом, взаимно-дополнительные соотношения – это конкретный вид обратных связей.

Приведенные формулировки позволяют утверждать, что данные принципы, учтены в концепции системно-объектного подхода в виде упомянутых выше правил системной композиции и, в частности, правила замкнутости.

Принцип *внешнего дополнения* (Бир): восходящие к системному центру воздействия координируемых элементов подвергаются своеобразному «обобщению», а нисходящие от системного центра координационные импульсы подвергаются «специфицированию» в зависимости от характера локальных процессов за счет обратных связей от этих процессов.

Данный принцип работает, естественно, только в иерархических структурах, обеспечивая «обобщение» и «специфицирование» взаимодействий между уровнями иерархии за счет обратных связей. В другом виде он может быть сформулирован следующим образом: «Любой элемент системной иерархии обладает функцией обобщения информации от нижележащих элементов для вышестоящих элементов и функцией специализации информации от элементов верхнего яруса иерархии для элементов нижнего яруса».

Очевидно, передаваемые вверх по иерархии воздействия имеет смысл «обобщать» только в том случае, если функциональные элементы верхнего уровня имеют более общие функции (решают более общие задачи). Соответственно «специфицировать» воздействия вниз по иерархии имеет смысл, если функциональные элементы нижнего уровня имеют более конкретные функции.

Перечисленные общесистемные принципы, касающиеся структурных (узловых) свойств систем обуславливают существование принципов, описывающих их функциональные характеристики, действия которых к тому же зависят от влияния универсального системообразующего фактора (внешней детерминанты системы), входящего в концепцию разрабатываемой системной теории.

2. Рассмотрим далее общесистемные закономерности, касающиеся функциональных (процессных) свойств систем.

Как уже было отмечено выше, каждая система характеризуется функциональными способностями (процессами, функциями), обеспечивающими преобразование «втекающих» по связям ресурсов в «вытекающие» ресурсы. Эти функциональные способности (процессы) обеспечивают баланс «притока» и «оттока» по функциональным

связям узла, занимаемого данной системой. При этом баланс одного и того же узла может быть обеспечен, в принципе, разными наборами функциональных способностей (наборами процессов), т.е. разными функциональными зависимостями выхода от входа. Формальная функциональная характеристика системы характеризует потенциальную возможность системы сбалансировать определенный узел.

Гипотеза *семиотической непрерывности* (Виноградов, Гинзбург): система есть образ её среды, т.е. система как элемент окружающей среды отражает некоторые существенные ее свойства.

Справедливость этой гипотезы подтверждается следующими соображениями. Главной составляющей среды, окружающей систему, в рамках системно-объектного подхода является ее надсистема. Эта надсистема «отображает» свою функциональность на функциональность системы с помощью функционального запроса на систему с определенной функцией (внешней детерминанты) и, таким образом, система своим функционированием (внутренней детерминантой) «отражает» некоторые функциональные свойства своей надсистемы, являющейся основной составной частью, окружающей систему среды. При этом функциональный запрос надсистемы (внешняя детерминанта системы) представляет собой набор связей запрашиваемой системы с другими системами, которые в свою очередь обеспечивают иерархическую системную структуру. Таким образом, гипотеза семиотической непрерывности справедлива вследствие наличия у системы на любом уровне иерархии внешней детерминанты (функционального запроса надсистемы) и внутренней детерминанты, и, таким образом, соблюдения принципа иерархичности.

Принцип *прогрессирующей механизации* (Берталанфи): части системы в ходе ее развития специализируются или становятся фиксированными по отношению к определенным функциям или механизмам.

Данный принцип является следствием того системологического (системно-объектного) факта, что причиной возникновения системы и ее становления является функциональный запрос надсистемы (т.е. внешняя детерминанта системы). Именно надсистема определяет функции своим системам, которые, в свою очередь, определяют функции своих подсистем. Адаптация всех этих систем и подсистем к своим внешним детерминантам (запросам) приводит к все большему соответствию их функционирования требованиям надсистемы и систем, требования которых становятся все более конкретными

(специализированными) с понижением уровня иерархии. Это и является необходимым и достаточным условием специализации или *прогрессирующей механизации*. Таким образом, прогрессирующая механизация – это процесс передачи сверху вниз (от надсистемы к системе и далее к подсистемам) функциональных запросов на формирование элементов с определенными функциями, что приводит к их адаптации к данным запросам с помощью механизма обратной связи. Следовательно, данный принцип является следствием проявления в иерархии систем семиотической непрерывности.

Принцип *актуализации функций* (Сетров): объект выступает как организованный (т.е. как система) лишь в том случае, если свойства его частей (элементов) проявляются как функции сохранения и развития этого объекта.

Данный принцип описывает учтенное в представленной системно-объектной концепции *отношение поддержания функциональной способности целого*. По сути дела, этот принцип описывает тот же самый процесс становления и развития системы, что и предыдущий принцип, но с противоположной стороны. Дело в том, что прогрессирующая механизация – это процесс передачи сверху вниз (от надсистемы к системе и далее к подсистемам) функциональных запросов, как было отмечено при описании предыдущего принципа. Актуализация же функций – это процесс, направленный от подсистем к системе и далее к надсистеме, представляющий собой процесс поддержания свойств более целого (снизу вверх). Следовательно, данный принцип также является следствием проявления в иерархии систем семиотической непрерывности.

Закономерность *самоорганизации* (Гиг): способность системы противостоять возрастанию энтропии и адаптироваться к изменяющимся условиям.

Содержательно данная закономерность соответствует понятию адаптации системы к запросу надсистемы в рамках системно-объектного подхода. Адаптация в данном случае рассматривается как приближения внутренней детерминанты системы к ее внешней детерминанте, т.е. как процесс все большего соответствия текущего функционирования системы функциональному запросу надсистемы. Адаптация, следовательно, приводит к увеличению степени поддержания функциональной способности надсистемы со стороны системы. При этом именно этот процесс и описывается с разных сторон

родственными принципами актуализации функций и прогрессирующей механизации.

Закон *иерархических компенсаций* (Богданов): рост разнообразия на верхнем уровне иерархии обеспечивается его ограничением на более низких уровнях, или уровень организации системного центра должен быть выше, чем уровень организации периферийных элементов.

Работа данного принципа также основана на системно-объектном механизме формирования внутренней детерминанты системы с помощью функционального запроса надсистемы или внешней детерминанты системы. Рост разнообразия у элементов верхнего уровня иерархии обеспечивается более общими функциями этих элементов (надсистем) по сравнению с конкретными функциями периферийных элементов (систем), так как в процессе обобщения функций увеличивается объем (разнообразие) решаемых при их выполнении задач. Соответственно ограничение разнообразия у периферийных элементов обеспечивается более конкретными функциями этих элементов по сравнению с более общими функциями элементов верхнего уровня, так как в процессе конкретизации функций объем (разнообразие) решаемых задач уменьшается. То есть, закон иерархических компенсаций является следствием дополняющих друг друга принципов прогрессирующей механизации и актуализации функций, а также согласуется с принципом внешнего дополнения (см. выше).

Закон *необходимого разнообразия* (Эшби): для создания системы, способной справится с решением проблемы, обладающей определенным разнообразием, необходимо обеспечить, чтобы система имела большее разнообразие возможностей, чем разнообразие решаемой проблемы.

Данный закон, по сути дела, учтен в представленной выше концепции теории систем, фиксирующей условия формирования требуемой системы из некоторого исходного материала. В соответствии с этими условиями область возможных состояний исходного материала всегда больше области требуемых надсистемой функциональных состояний (**ОВС > ОТФС**). Кроме того, это закон согласуется с предыдущим законом и, очевидно, использует его механизм.

Описанные выше общесистемные закономерности, касающиеся функциональных свойств систем, в свою очередь, обуславливают существование закономерностей, описывающих их объектные (субстанциальные) характеристики.

3. Рассмотрим далее общесистемные закономерности, касающиеся объектных (субстанциальных) свойств систем.

Принцип *совместимости* (Сетров): условием взаимодействия между системами является наличие у них относительной совместимости, то есть относительной качественной и организационной однородности.

Данный принцип учтен в обсуждаемой концепции системного подхода с помощью упомянутых выше правил системной композиции. При этом если, как показано выше, принцип организационной непрерывности описывает, по сути дела, структурную совместимость, гипотеза семиотической непрерывности функциональную совместимость, то принцип совместимости описывает совместимость системы в целом с учетом еще и субстанциального аспекта. Таким образом, обсуждаемый принцип является последовательно следствием организационной и семиотической непрерывности.

Эквифинальность (Берталанфи): способность системы достигать состояния, которое не зависит от времени и начальных условий, а зависит только от параметров системы.

В описаниях и пояснениях к данной закономерности при этом, как правило, упоминается еще и влияние окружающей среды. В рамках предлагаемого системного подхода надсистема является главной составляющей окружающей систему среды, а внешняя детерминанта системы является одним из важнейших ее параметров вместе с ее внутренней детерминантой. Таким образом, эквифинальность есть результат адаптации системы к функциональному запросу надсистемы, т.е. самоорганизации системы.

Закон *минимуму* (Богданов): устойчивость системы определяется устойчивостью ее самого слабого звена.

С точки зрения рассматриваемого системно-объектного подхода устойчивость любой системы напрямую зависит от степени адаптированности этой системы к функциональному запросу ее надсистемы. Это обстоятельство обусловлено тем, что с увеличением степени адаптированности системы увеличивается степень поддержки, которую оказывает система своей надсистеме. А чем лучше система поддерживает надсистему, тем больше надсистема в этой системе заинтересована и тем лучше она (надсистема) эту систему обеспечивает со своей стороны, что приводит к увеличению устойчивости этой системы.

Если рассматривать систему, подсистемы которой адаптированы к ее запросу в разной степени, то наименее адаптированная подсистема (она же наименее устойчивая) может быть легче других подсистем выведена (или выйти) из данной системы. При этом система в целом, даже если будет продолжать выполнять свою функцию в надсистеме, все равно не останется такой же системой какой она была до утраты одной из своих подсистем. Таким образом, устойчивость системы, как именно такой системы, определяется устойчивостью самой слабо адаптированной, самой неустойчивой подсистемы, что соответствует закону минимума, который в нашем понимании имеет некоторую зависимость от закономерности самоорганизации.

Закон *расхождения* (Спенсер): различные части однородной системы подвержены действию сил, различающихся по качеству и величине, вследствие чего они изменяются различно. Это увеличивает разнообразие и обеспечивает универсум разнообразным исходным материалом.

Данный закон соответствует ситуации, при которой две тождественные системы имеют две различные внешние детерминанты (два разных функциональных запроса). Естественно, в процессе адаптации к различным запросам данным системам будет свойственно прогрессирующее накопление различий в виде различных внутренних детерминант. По-видимому, справедливо утверждать, что данный закон уточняет законы иерархических компенсаций и необходимого разнообразия, описывая ситуацию, при которой разнообразие увеличивается в случае дифференциации системы. Таким образом, закон расхождения является видом закона иерархических компенсаций или необходимого разнообразия с учетом закономерности самоорганизации.

Закон *опыта* (Эшби): единообразное воздействие на некоторое множество элементов уменьшает разнообразие состояний этого множества.

Данный закон является противоположным по отношению к предыдущему закону и уменьшает разнообразие универсума. Закон опыта соответствует ситуации, при которой системы (или одна система) подвергаются воздействию одного и того же (или постоянного) функционального запроса (внешней детерминанты), что в процессе адаптации приводит к сближению внутренних детерминант или существенному сокращению области возможных состояний детерминируемой системы. Таким образом, закон опыта является видом закона

необходимого разнообразия или иерархических компенсаций с учетом закономерности самоорганизации.

Представленное выше описание общесистемных принципов и закономерностей, а также из взаимосвязей позволяет описать эти связи в виде иерархической схемы (см. рис. 2.9). Данная схема показывает однозначную зависимость закономерностей, касающихся функциональных характеристик систем (на схеме с цифрой 2. …), от закономерностей, описывающих их структурные характеристики (на схеме с цифрой 1. …), и, соответственно, зависимость закономерностей, связанных с объектными характеристиками систем (на схеме с цифрой 3. …), от закономерностей функциональных.

Кроме того, выявленные взаимосвязи между общесистемными принципами и закономерностями позволяют установить необходимые и достаточные условия возникновения системного эффекта или явления *эмерджентности*.

Дело в том, что если в основе всех системных отношений лежит иерархия, в которой подсистема к системе и система к надсистеме находятся в отношении поддержания функциональной способности более целого, приводящее к обобщению и увеличению разнообразия на более верхних уровнях, то появление свойств нижнего уровня на верхнем или их аддитивность для верхнего уровня, естественно, исключены. Следовательно, свойства системы в целом не могут быть обнаружены у ее подсистем и не могут быть получены путем сложения свойств последних. При этом, по-видимому, структурные (узловые) общесистемные закономерности (в первую очередь принцип иерархичности) можно рассматривать как необходимые условия эмерджентности, а функциональные (принципы актуализации функций, а также закон иерархических компенсаций) – как достаточные.

2.1.2. Учет общесистемных закономерностей системами-классами

Представленные выше описания общесистемных принципов и закономерностей не оговаривают явно, к каким системам эти закономерности относятся. Очевидно, по умолчанию, имеются в виду системы-явления, так как они используются их авторами для описания соответствующих примеров. Однако, в интересах теории систем актуальным является обеспечение учета общесистемных закономерностей и концептуальными системами. Рассмотрим далее возможности учета тех же общесистемных закономерностей (определения см. п. 2.1.1.) в рамках

концептуальных систем, т.е. систем-классов, учитывая при этом, что субстанциальных характеристик у этих систем нет.

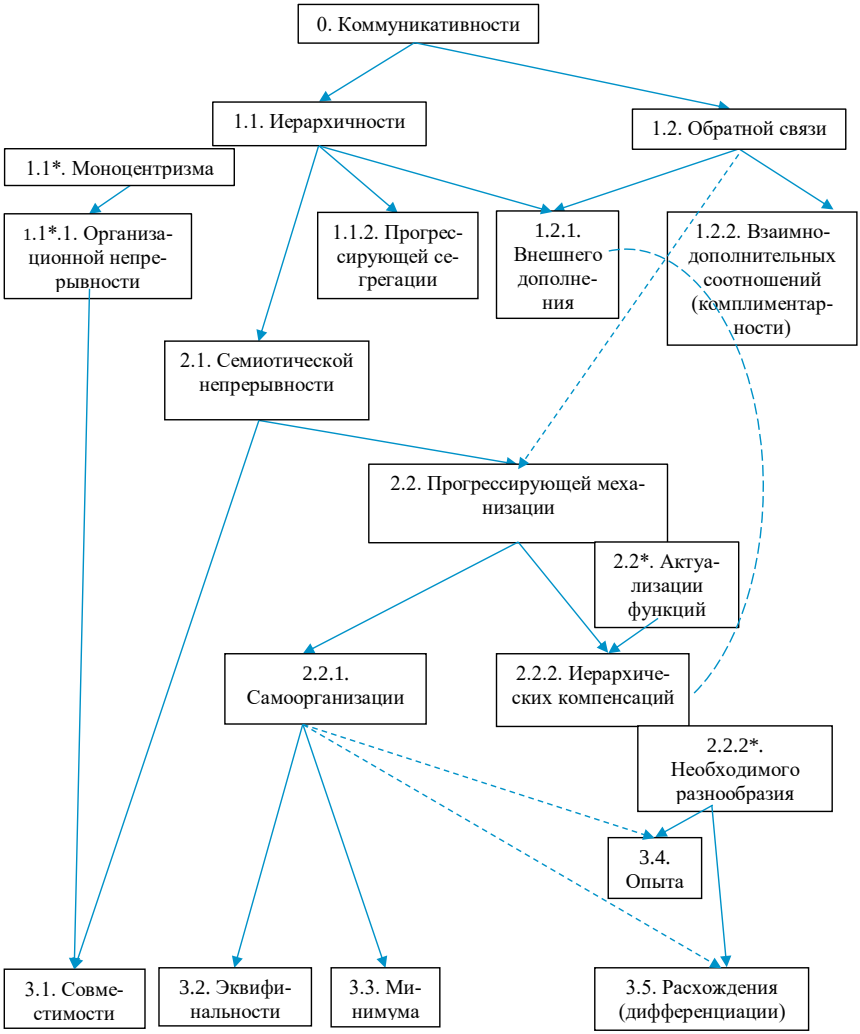


Рис. 2.9. Схема связей общесистемных закономерностей

1. В первую очередь рассмотрим закономерности, связанные со структурными (узловыми) характеристиками систем-классов, так как именно узловая характеристика системы рассматривается в качестве

универсального системообразующего фактора в рамках системно-объектного подхода.

Из содержательного определения системы-класса следует очевидное выполнение ***принципа коммуникативности*** и ***принципа иерархичности***. При этом принцип иерархичности работоспособен только при выполнении принципа коммуникативности, что хорошо согласуется с содержательной трактовкой этих принципов в терминах системно-объектного подхода.

Выполнение принципа иерархичности приводит к выполнению ***принципа моноцентризма***. С точки зрения авторов, данный принцип можно и нужно понимать более широко, чем в работах Богданова А.А. (см., например, [Маторин, 1996; Маторин, 2017]) особенно если речь идет о системах-классах. Т.е. *иерархия систем-классов имеет одну единственную вершину*. Обоснование см. пункт 2.1.1.

Расширенное понимание принципа моноцентризма приводит к выполнению ***принципа организационной непрерывности***, что доказано нами для систем-явлений в работе [Маторин, 2018, 2], а для систем-классов совершенно очевидно. Таким образом, принцип организационной непрерывности соблюдается только в случае выполнения принципа моноцентризма.

Для обоснования выполнения ***принципа обратной связи*** в рамках систем-классов необходимо рассмотреть особенности динамики таких систем. К динамическим явлениям в системах-классах можно отнести следующие процессы (первые из них будем называть «адаптационными», а вторые – «эволюционными»):

- изменения ролей (функций) классов в классах более высокого яруса иерархии;
- возникновение новых классов в классах более высокого яруса иерархии.

Примеры таких процессов можно увидеть, анализируя процессы изменения свойств видов мебели или технических систем (автомобилей) в ходе их совершенствования по требованиям общества. При этом процессы совершенствования существующих видов, в конце концов, приводят к появлению новых видов (систем-классов) мебели или технических систем. Например, возникновение нового класса автомобилей (карьерный самосвал, автомобиль-дача и т.п.) для перевозки специфических видов грузов в соответствии с функциональным запросом системы-класса «автомобильный транспорт», адаптирующегося к изменяющимся запросам человеческого сообщества.

Такие же процессы изменения свойств видов и появления новых видов (классов) можно проследить на примерах адаптации и эволюции биологических систем, т.е. видов этих систем. Например, возникновение новых видов домашней птицы (индоутка и т.п.) в соответствии с функциональным запросом системы-класса «домашняя птица», адаптирующегося к изменяющимся запросам человеческого сообщества.

Упомянутые адаптационные и эволюционные процессы обусловлены согласованием, с одной стороны, функционального запроса системы-класса более высокого яруса иерархии к системе-классу нижнего яруса и, с другой стороны, отношения поддержания функциональной способности системы-класса более высокого яруса со стороны системы класса нижнего яруса. Таким образом, любая система-класс существует в условиях постоянно действующих обратных связей. При этом, естественно, принцип обратной связи работоспособен только при выполнении принципов коммуникативности и иерархичности.

Для обоснования выполнения в рамках систем-классов ***принципа прогрессирующей сегрегации*** рассмотрим, что представляет собой дифференциация системы-класса.

Дифференциация (по Г. Спенсеру, который первым ввел это понятие) – разделение в процессе эволюции однородной системы (биологические организмы, представители определенной профессии и т.п.) на две или несколько групп, отличающиеся по своим параметрам. Подобное дробление может иметь несколько иерархических уровней. Для системы-класса это означает образование (объединение) из множества видовых по отношению к данному классу систем одной или нескольких систем-классов нижнего яруса иерархии по сравнению с данным уровнем (подсистем-классов). При этом ролевые связи видовых систем, образующих подсистему-класс, с системой-классом ослабевают, но усиливаются связи с данной подсистемой. Примеры действия данного принципа соответствуют примерам, приведенным при обсуждении принципа обратной связи. Принцип прогрессирующей сегрегации в рамках систем-классов выполняется при выполнении принципа иерархичности.

Выполнение принципов иерархичности и обратной связи приводит к выполнению ***принципа внешнего дополнения***. Предложенная выше формулировка данного принципа: «Любой элемент системной иерархии обладает функцией обобщения информации от нижележащих элементов для вышестоящих элементов и функцией специализации информации от элементов верхнего яруса иерархии для элементов нижнего яруса» [Маторин, 2016, 2], соответствует сути родовидовых

отношений в иерархии систем-классов (таксономии). Таким образом, принцип внешнего дополнения в иерархии систем-классов выполняется естественным образом.

Принцип взаимно-дополнительных соотношений/комплиментарности работоспособен в рамках систем-классов при выполнении принципа обратной связи. По сути дела, это один и тот же принцип. Дело в том, что в иерархии систем-классов каждая система связана с другими системами двумя видами взаимно-дополнительных отношений: обобщение и специализация, т.е. системы-классы существуют в замкнутых контурах обратных связей. При этом первая связь соответствует отношению поддержания функциональной способности надсистемы-класса со стороны системы-класса, а вторая – функциональному запросу надсистемы-класса на систему-класс с определенной ролью.

2. Рассмотрим далее закономерности, связанные с функциональными характеристиками систем-классов, которые в рамках системно-объектного подхода являются следствием узловых.

Концепция системно-объектного подхода предполагает, что главной составляющей среды, окружающей систему, является ее надсистема независимо от пути проявления системности. Эта надсистема «отображает» свою функциональность на функциональность системы с помощью функционального запроса на систему с определенной функцией (внешней детерминанты) и, таким образом, система своим функционированием (внутренней детерминантой) «отражает» некоторые функциональные (т.е. самые существенные) свойства своей надсистемы. Таким образом, системно-объектный подход предполагает, что ***гипотеза семиотической непрерывности*** является справедливой как для систем-явлений, так и для систем-классов. При этом для последних функциональный запрос надсистемы (внешняя детерминанта системы) представляет собой не набор связей запрашиваемой системы с другими системами, как для систем-явлений, а роль системы-класса верхнего уровня, требующую поддержки со стороны системы-класса нижнего уровня. Таким образом, семиотическая непрерывность является следствием иерархичности систем-классов.

Представленное понимание семиотической непрерывности естественным образом обеспечивает выполнение ***принципа прогрессирующей механизации,*** как для систем-явлений, так и для систем-классов. Дело в том, что и в том и в другом случае и части системы-явления, и виды системы-класса приобретают свою функциональность или роль под влиянием соответствующей надсистемы, что и обеспечивает их определенную специализацию.

Таким же образом обеспечивается выполнение ***принципа актуализации функций***. Как и в рамках систем-явлений, так и в рамках систем-классов принцип актуализации функций и принцип прогрессирующей механизации описывают одно и то же явление функционального соответствия систем разного уровня иерархии, но с разных сторон. Прогрессирующая механизация описывает соответствие систем сверху в низ (от надсистемы к системе; внешняя детерминанта), актуализация функций – снизу вверх (от системы к надсистеме; поддержание функциональной способности более целого). Соответственно данный принцип работает только при выполнении предыдущего, так как существование любой системы-класса обусловлено запросом надсистемы-класса на систему-класс с определенной ролью в данной надсистеме.

Выполнение принципа прогрессирующей механизации обеспечивает выполнение ***принцип самоорганизации***, так как, по сути дела, оба принципа описывают один и тот же процесс адаптации системы к запросу надсистемы и для систем-явлений, и для систем-классов.

Закон иерархических компенсаций уточняет действие принципа прогрессирующей механизации с учетов влияния принципа внешнего дополнения, так как естественным образом учитывает суть родовидовых отношений в иерархии систем-классов (таксономии).

Следствием закона иерархических компенсаций является ***закон необходимого разнообразия***, так как для обеспечения необходимого разнообразия используется механизм иерархических компенсаций. Эти две закономерности соотносятся друг с другом также как принципы прогрессирующей механизации и актуализации функций.

Приведенные рассуждения показывают, что основные известные общесистемные закономерности (структурные и функциональные) выполняются как для систем-явлений, так и для систем-классов и их взаимодействия соответствуют приведенной в предыдущем пункте схеме. Поэтому в теорию систем, основанную на системно-объектном подходе, естественным образом могут быть включены и материальные, и концептуальные системы.

2.2. Системно-объектная картина мира

Как было показано выше применение системного (системно-объектного) подхода к классам (концептуальным или внешним системам) наравне с объектами (конкретными или внутренними системами) вполне возможно. Кроме того, это весьма целесообразно. Дело в том, что

системный подход к классам объектов позволяет по-новому взглянуть на принцип детерминизма и положение о бесконечности мира. Используем при этом результаты предварительного анализа [Маторин, 1996].

По данным современной космологии, Вселенная (материальный мир, рассматриваемый в аспекте пространственно-временного распределения масс) бесконечна в пространстве и времени. Детерминизм же как учение о взаимосвязи и взаимной определённости всех явлений и процессов утверждает, что понимать и объяснять вещи и процессы – значит выяснять их причины. Принцип причинного объяснения считается единственно эффективным в научном познании мира и человека.

Как было сказано выше системно-объектный подход, рассматривает систему как функциональный объект, функция которого обусловлена функцией объекта более высокого яруса (надсистемы). Детерминантный анализ, как составная его часть, позволяет утверждать, что явление обуславливания функции системы функцией надсистемы (т.е. функциональный запрос надсистемы на систему с определенной функцией или внешняя детерминанта системы) представляет собой причину возникновения любой системы. Система же при этом находится с надсистемой в отношении поддержания функциональной способности целого. Т.е. причина существования системы и наличия у нее определенных свойств обусловлена ее надсистемой. Так как «система иерархична по своей природе, каждый ее компонент, в свою очередь, рассматривается как система, а сама исследуемая система представляет собой лишь один компонент еще более широкой системы» [Бреховских, 1986, с.19], то причина системы находится не просто в надсистеме, а в иерархии надсистем. При этом данная иерархия, с одной стороны (сверху в низ), представляет иерархию потребностей надсистем в поддержании их свойств как целого, а, с другой стороны (с низа вверх), иерархию систем, поддерживающих своими свойствами соответствующие надсистемы.

Исследование упомянутой выше иерархии для систем-явлений (конкретных или внутренних систем) от некоторого конкретного явления в сторону цепочки все более масштабных надсистем (от части к целому) показывает ее бесконечность, что соответствует положению о бесконечности мира. Концепция же детерминизма на основании принципов причинности и закономерности утверждает, что все объекты, явления, процессы и их свойства возникают и развиваются в результате действия определенных причин, имеющих объективный характер. Т.е. для реальных вполне определенных свойств любой системы должна существовать вполне определенная причина их возникновения.

Следовательно, имеет место противоречие между тезисом в виде принципа детерминизма и концепцией бесконечности мира как антитезисом. Это противоречие обусловлено тем, что в соответствии с принципом детерминизма причиной наличия у системы определенных свойств является запрос наднад ... системы, которая в соответствии с концепцией бесконечности мира отстоит от своего следствия на бесконечное число ярусов в бесконечной иерархии систем-явлений (внутренних систем).

Использование иерархии систем-классов (внешних или концептуальных систем), аналогичной предыдущей позволяет устранить выявленное выше противоречие.

Дело в том, что исследование иерархии систем-классов от некоторой системы (например, той же, что и для предыдущей иерархии, как системы-класса единичного объема) в сторону цепочки все более общих надсистем (от вида к роду) показывает, что такая иерархия имеет естественное ограничение. Выше при обосновании общесистемного принципа моноцентризма данный факт уже был рассмотрен. Здесь приведем его кратко.

Ограничение иерархии систем-классов связано с тем, что при переходе по иерархии от вида к роду (от подкласса к классу) происходит переход от свойства подсистемы к свойству системы (несводимому к свойствам подсистем) в сторону обобщения, т.е. с сокращением множества признаков (свойств), за счет которых системы-явления относятся к данному классу. Множество же систем-явлений, входящих в класс будет больше, чем множество систем-явлений, входящих в подкласс. Этот процесс соответствует закону обратного отношения объема и содержания понятий, в которых отражаются в нашем сознании системы-классы [Кондаков, 2012], т.е. с увеличением объема содержание уменьшается. Для самих систем-классов это соответствует уменьшению набора признаков, за счет которого формируется система-класс, при увеличении множества явлений, относящихся к данному классу. При этом, в связи с конечностью набора признаков [Маторин, 1996], увеличение объема и сокращение числа признаков за конечное число шагов приводит к такому классу, для отнесения к которому не остается признаков, а объем которого становится бесконечно большим. Т.е., рассматривая иерархическую структуру систем-классов, можно прийти к выводу, что эта структура имеет единственный верхний узел, то есть, существует единственная Надсистема-класс. Если предположить обратное: надсистема не является единственной и имеется еще хотя бы одна система-класс того же уровня, то эти системы можно

считать элементами системы более высокого уровня. Таким образом, Надсистема-класс, включающая в себя все виды систем, является единственной.

Следовательно, иерархия систем-классов, не противоречащая положению о бесконечности мира (по объему классов), не противоречит, при этом, концепции детерминизма, так как однозначно указывает на исходную причину существования конкретной системы.

Кроме того, вхождение всего существующего в одну Надсистему обнаруживается в результате сопоставления некоторых известных общесистемных закономерностей, исследованных еще А.А. Богдановым, что было выше нами показано. Например, принципа организационной непрерывности, констатирующего факт наличия между всякими двумя системами звеньев, вводящих их в одну «цепь ингрессии», и принципа моноцентризма. Нами в работе [Маторин, 2018, 2] формально доказано, что первый из названных принципов справедлив только при выполнении второго на уровне универсума.

Таким образом, иерархия систем-классов подтверждает также положение о единстве универсума, который рассматриваться как общепринятый. Различные формы действительности благодаря своему существованию образуют целостное единство бесконечного, непреходящего мира. Принцип единства мира, рассматривается в философии как один из основополагающих принципов диалектики, имеющий первостепенное мировоззренческое и познавательное значение. Для обозначения этого единства в философии употребляется специальный термин: «*бытие*». Понятие (класс) *бытие* отражает всеобщий процесс, объемлющий различные формы мироздания, и в том числе человеческое существование, в их взаимосвязи. Оно (понятие *бытие*) ориентировано на предельные общие характеристики существующего. Вместе с тем *бытие,* как обобщение, так или иначе, указывает на связь, порядок или иерархию различных видов процессов, вещей, событий и т.д. [Современный …].

Приведенные рассуждения позволяют сделать следующие выводы:

- Полноценная теория систем должна учитывать как системы-явления (внутренние системы), так и системы-классы (внешние системы).
- В рамках внутренних систем (систем-явлений) принцип детерминизма и концепция бесконечности мира вступают в противоречие.
- В рамках внешних систем (систем-классов) принцип детерминизма в противоречие ни с чем не вступает.

- Исходная причина существования систем и наличия у них определенных свойств находится в иерархии внешних систем (систем-классов).
- Реальный мир представляет собой объектно-ориентированную систему, классы которой представляют собой внешние (концептуальные) системы, определяющие свойства объектов, а объекты – внутренние (конкретные, материальные) системы, осуществляющие реальные взаимодействия.

2.3. Системно-объектное понимание эволюции общества

На сегодняшний день является общепризнанным, что традиционная схема развития цивилизации ведет человечество к глобальной катастрофе. Конференция ООН по окружающей среде и развитию уже в 1992-м году (КОСР-92) признала модель развития современной цивилизации моделью *неустойчивого развития.* Высший орган мирового сообщества призвал правительства и народы всех стран к скорейшему переходу на модель *устойчивого развития,* которая должна характеризоваться, во-первых, возможностью обеспечения выживания и непрерывного развития человечества, и, во-вторых, возможностью обеспечения сохранения биосферы, то есть *биосфероцентричностью*.

Обоснования данного факта, получены в ходе моделирования глобального развития (*глобального моделирования* [см., например, Гвишиани, 1977]). «На математических моделях установлено, что мощь воздействий человека на природу стала сопоставима с мощью самих природных сил и возникла угроза глобального кризиса (демографического, экологического, энергетического, сырьевого и т.д.). Объединение, таким образом, усилий всего человечества и всей науки для решения этих глобальных проблем стало жизненной необходимостью. При этом для обеспечения перехода к новой модели развития, в первую очередь, необходима переориентация сознания и интересов людей на новые ориентиры и цели, отказ от многих потребностей, которые до сих пор считались общепринятыми» [Урсул, 1996, с.8].

Разработчики концепций устойчивого развития видят решение названных проблем в смещении акцентов развития цивилизации с вещественно-энергетических на информационные. Это смещение, в свою очередь, ведет к становлению *информационного общества* как первой ступени *ноосферы* [Урсул, 1996]. Предполагается, что переход от вещественно-энергетического общества к информационному будет

способствовать переходу к модели устойчивого развития, а экономика нового общества будет, прежде всего, основана на знаниях и научной информации [Урсул, 1993]. При этом одним из признаков начала становления информационного общества и ноосферы рассматривается появление такого научного направления как «системные исследования», которые используются как средство «диалектической обработки» создаваемого конкретной наукой знания, то есть как средство «организации» ноосферы [Шрейдер, 1983, с. 191].

Рассмотрим возможности применения последних достижений системных исследований для обоснования системных причин и вероятных последствий, упомянутого выше глобального кризиса, в том числе антропологического, используя работу [Matorin, 2013].

Системно-объектный подход, как было показано выше, трактует эволюцию системы как ее адаптацию к изменяющемуся функциональному запросу или внешней детерминанте. А адаптацию – как приближение внутренней детерминанты системы к внешней. Представленное системное понимание эволюции позволяет определить механизм эволюции, в данном случае, системы **человек**. Данный механизм выявляется путем определения внешней и внутренней детерминант этой системы и их взаимосвязи, т.е. путем проведения детерминантного анализа.

Анализ внутренней детерминанты системы ЧЕЛОВЕК.

С точки зрения внутренней детерминанты функциональные способности человека хорошо прослеживаются в результате рассмотрения в качестве подсистем, составляющих систему **человек**, так называемых компонент (или уровней) *сенсомоторной активности*. Сенсомоторное управление образует кибернетическую (замкнутую через внешнюю среду) схему функционирования системы. Управление в этой схеме осуществляется путем взаимодействия трех блоков: *сенсорного распознавания*, *инстанции принятия решений* и *моторной активности* [Кликс, 1985]. В данном случае эти блоки рассматриваются на нескольких уровнях, обусловленных существующими у человека уровнями отражения (сенсорного распознавания), мышления (инстанции принятия решений) и активности. В соответствии с данными теории отражения и психологии мышления [Тихомиров, 1984] целесообразно учитывать четыре уровня сенсомоторной активности. Следовательно, согласно данному подходу, система **человек** представляет собой четыре взаимодействующие между собой подсистемы, функционирование которых поддерживает функциональную способность человека в целом. Рассмотрим, используя работу [Маторин, 1998], эти подсистемы,

каждая из которых состоит из трех упомянутых выше блоков, и их предназначение более подробно (см. рис. 2.10).

Подсистема №1 обеспечивает прием сигналов из внешней и внутренней среды, физические отправления и физиологические рефлекторные реакции (сердцебиение, дыхание, потоотделение, сужение и расширение сосудов и зрачков, сокращение мышц, работа системы выделения и т.д.).

1.1. Сенсорное распознавание сигналов внешней и внутренней среды на уровне отдельных ощущений (зрительных, слуховых, обонятельных, осязательных, вкусовых и пр.).

1.2. Управление путем гомеостатического регулирования, т.е. путем поддержания постоянства характеристик внутренней среды подсистемы.

1.3. Моторная активность в виде физиологических реакций.

Подсистема №2 обеспечивает первичную семантическую обработку сенсорной информации; восприятие удовольствий и боли; формирование текущих образов окружающих объектов и реализацию безусловных рефлексов (сосательного, пищевого, размножения и т.д.).

2.1. Распознавание на уровне восприятия информации различной модальности, поступающей из подсистемы №1, путем предварительного формирования целостного (интегрального) конкретного образа текущей ситуации.

2.2. Выработка и принятие решений в текущей ситуации на уровне наглядно-действенного мышления, осуществляемого с помощью реального, физического преобразования ситуации и опробования свойств объектов.

2.3. Активность в текущей ситуации в виде инстинктивного поведения на основе безусловных (врожденных) рефлексов.

Подсистема №3 обеспечивает вторичную семантическую обработку сенсорной информации; существование эмоций, желаний, стремлений, влечений, чувств, воображения, волевых импульсов и оценок; формирование и хранение обобщенных образов окружающих объектов; мотивированное поведение на основе условных (приобретенных) рефлексов (прошлого опыта).

3.1. Распознавание на уровне чувственного отражения «по ассоциации по сходству» сенсорной информации, поступающей из подсистемы №2, путем предварительного формирования обобщенных образов, имеющих сходство с отражаемыми объектами окружающей среды. Это осуществляется путем наложения друг на друга текущих образов и запоминания повторяющихся деталей, с помощью так называемого «*механизма суммации*».

Рис. 2.10. Уровни сенсомоторной активности человека

3.2. Выработка и принятие решений на уровне образного мышления, осуществляемого с помощью образного представления ситуации и воображения ее изменений с учетом многообразия различных воспринятых и отраженных характеристик объектов.

3.3. Активность в виде мотивированного поведения на основе условных (приобретенных) рефлексов (прошлого опыта).

Подсистема №4 обеспечивает самосознание, работоспособность интеллекта, понимание, рациональное поведение, синтез идей и выработку гипотез.

4.1. Распознавание на уровне абстрактного отражения «по ассоциации по смежности» информации, поступающей из подсистемы №3, путем предварительного формирования образов, имеющих знаковую природу (в первую очередь слов естественного языка – понятий) и отражающих определенные не обязательно чувственно воспринимаемые свойства объектов. Это осуществляется путем выдвижения и проверки гипотез о существенных свойствах объектов с помощью так называемого «*механизма активного поиска*».

4.2. Выработка и принятие решений на уровне словесно-логического мышления, осуществляемого с помощью понятий и логических конструкций на базе языковых средств, а также с помощью обобщения путем абстрагирования.

4.3. Активность в виде сознательного, рационального поведения на основе целеполагания, с учетом планирования и прогнозирования развития и изменения данной ситуации.

Из представленной схемы хорошо видно, что каждая следующая по номеру подсистема, представляя собой, конечно, новое качество, не может функционировать без информации, вырабатываемой предыдущей подсистемой. Можно даже утверждать, что каждая следующая подсистема является эволюционной надстройкой над предыдущей. Например, целеполагание и планирование поведения на уровне подсистемы №4 осуществляется благодаря наличию определенных мотивов на уровне подсистемы №3. Соответственно эти мотивы поведения есть результат наличия на уровне подсистемы №2 возможности восприятия удовольствия и боли. Распознавание удовольствия и боли, в конце концов, основано на способности обеспечивать постоянство характеристик внутренней среды (самосохранение) подсистемой №1.

Результаты рассмотрения подсистем системы человек и их функционирования позволяют предположить, что в настоящее время основная функциональная способность человека (как вида) заключается в

рациональном поведении, которое основано на абстрактном отражении, словесно-логическом мышлении, целеполагании и планировании.

Этот вывод хорошо согласуется с современным представлением о человеке и его проявлении. Такие черты человеческого сообщества (антропосферы) как рациональность, интеллектуальность и превращение их продуктов – науки и технологии – в геологический фактор эволюции всей планеты [Вернадский, 1991] однозначно свидетельствуют в пользу этого вывода. Более того, наличие рационального сознания, интеллекта, словесно-логического мышления часто рассматривается как отличительный видовой признак человека [Косыгин, 1995]. Предлагаемая модель уровней сенсомоторной активности человека, кроме согласования с современными научными данными, также хорошо согласуется с представлениями о человеке мыслителей прошлого, зафиксированными так называемой эзотерической психологией [Ошо, 1992].

Анализ внешней детерминанты системы ЧЕЛОВЕК.

Изменение функциональной способности системы есть изменение ее внутренней детерминанты. Изменение же внутренней детерминанты системы может происходить, как отмечено выше, либо при неизменном функциональном запросе надсистемы (адаптация), либо в условиях изменения внешней детерминанты данной системы (эволюция). Следовательно, в общем случае, анализу изменения функциональной способности системы должен обязательно предшествовать анализ внешней детерминанты этой системы, т.е. детерминантный анализ функционального запроса надсистемы.

При построении схемы уровней сенсомоторной активности в качестве текущей внутренней детерминанты системы рассматривалась функция одного из этих уровней. Формирование же уровней сенсомоторной активности системы происходит вследствие изменения особенностей жизнедеятельности данной системы. Эти особенности и следует относить к проявлению внешней детерминанты этой системы. Таким образом, для сопоставления изменений внутренней и внешней детерминант необходимо проследить, как изменение условий жизнедеятельности предков человека связано с изменением их сенсомоторной активности. Для этого необходимо использовать результаты антропологии, палеонтологии и археологии, наиболее значимыми из которых, на наш взгляд, являются данные о смене палеолита (древнего каменного века) неолитом (новым каменным веком). Смена названных этапов представляет особый интерес, так как именно она связана с появлением человека современного типа и являет собой «самый

критический и величественный из всех периодов прошлого – период возникновения цивилизации» [П. де Шарден, 1987, с. 164]. Следовательно, детерминантный анализ системообразующих факторов этого периода должен быть чрезвычайно полезен для выявления направления эволюции системы **человек**.

Названный период вызывает наибольший интерес с точки зрения так называемой «неолитической метаморфозы» (П. де Шарден) или «неолитической революции» (Г. Чайлд), которая представляет собой смену *непроизводящей* хозяйственной деятельности *производящей*. Данные о переходе от собирательства к производящему хозяйству при переходе от палеолита к неолиту, в общем, хорошо известны. В настоящее время эти данные широко используются в ходе разработки концепции выживания и устойчивого развития человечества [Урсул, 1993]. Приведем некоторые из этих данных с целью подчеркнуть важные, с нашей точки зрения, аспекты этого перехода и, в частности, происходящий в то время экологический кризис.

Во-первых, обратим внимание на классическое и почти поэтическое описание этого процесса П. де Шарденом. «Прежде всего, непрекращающийся прогресс размножения. Вследствие быстрого роста числа индивидов свободная территория уменьшается. Группы сталкиваются между собой. По этой причине амплитуда перемещений уменьшилась, и встал вопрос об извлечении наибольших результатов из все более ограниченных владений. Можно предположить, что под давлением этой необходимости возникла идея сохранения и воспроизведения на месте того, что раньше приходилось искать и преследовать вдали. Разведение скота и обработка земли заменяют сбор плодов и охоту. Пастух и земледелец. Из этой важной перемены вытекает все остальное» [П. де Шарден, 1987, с. 165].

Во-вторых, приведем более современные и конкретные данные из исследования по истории хозяйственно-культурного развития человечества. «Истребление крупных животных и дальнейший рост населения к концу верхнего палеолита привели к острому кризису охотничьего хозяйства, охватившему громадные территории теплого и умеренного климатических поясов. В отдельных районах кризисная ситуация была катастрофической реальностью. ... Там, где кризис охотничьего хозяйства проявлялся особенно остро и где отсутствовала возможность миграции или освоения новых площадей, охотники и собиратели были вынуждены переходить к более интенсивному использованию природных ресурсов, совершенствуя свою техноло-

гию. Прогресс в развитии орудий труда и жизнеобеспечивающих систем ускорил переход к производящим формам хозяйства. ... Археологам удалось на уникальном археологическом материале проследить многие стадии перехода к растениеводству в Мезоамерике. В условиях острого экологического кризиса 12–10 тыс. лет назад добычей охотников сделались даже грызуны. Население перешло к интенсивному собирательству съедобных растений, что положило начало развитию земледельческих навыков. ... решающим в те времена стало стремление «собирателей урожая» восполнить искусственными посадками убывающие в процессе эксплуатации естественные заросли съедобных растений. Этот тезис был экспериментально доказан на материалах пещерного поселения в долине Оаксака ... Процесс перехода к скотоводству и растениеводству был выявлен также на археологических памятниках Юго-Западного Ирана в долине Дех-Луран ...» [Андрианов, 1989, с. 76–78].

Приведенных данных вполне достаточно для того, чтобы процесс смены палеолита неолитом определить в терминах детерминантного анализа. Этот процесс с системной точки зрения представляет собой смену определенных аспектов внешней детерминанты системы **человек**, которая выразилась в изменении условий и способов жизнедеятельности существовавших тогда предков человека.

Анализ изменения внутренней детерминанты системы ЧЕЛОВЕК

Изменение внешней детерминанты системы всегда приводит к изменению предельной внутренней детерминанты этой системы, что, в свою очередь, влечет за собой соответствующее изменение ее текущей внутренней детерминанты. Рассмотрим этот процесс в связи с изменением внешней детерминанты системы **человек** в ходе смены палеолита неолитом (неолитической революции).

Сравнивая условия жизнедеятельности до и после неолитической революции можно видеть, что до этой революции непроизводящее хозяйство обуславливало использование человеком в основном своего прошлого опыта, а после этой революции производящее хозяйство потребовало использования, в первую очередь, планирования и целеполагания. Следовательно, можно утверждать, что в ходе смены палеолита неолитом и последовавшей за этим неолитической революции внешняя детерминанта системы **человек**, детерминировавшая прежде развитие третьего уровня сенсомоторной активности, изменилась и потребовала активизации и развития четвертого уровня.

Для более детального анализа изменений текущей внутренней детерминанты системы **человек** в этот период необходимо проследить изменения особенностей сенсомоторной активности предков человека в тот же период. Об этом в литературе по антропологии и палеонтологии имеется достаточное количество сведений. Самыми существенными из них, по нашему мнению, являются данные о том, что в это время происходит смена на исторической сцене палеоантропа, в частности древнего человека *неандертальского* типа, неоантропом, то есть человеком современного типа, так называемым *кроманьонцем*. Результаты сравнения особенностей сенсомоторной активности неандертальца и кроманьонца по данным [Андрианов, 1989; Донских, 1988; Кликс, 1985; Косыгин, 1995] представлены в таблице №2.6.

Таблица 2.6

Сравнение особенностей сенсомоторной активности неандертальца и кроманьонца

Неандертальцы	***Кроманьонцы***
Преимущественно образная репрезентация информации в памяти, при недостаточной ее логико-понятийной обработке. Интеллектуальные способности в зачаточном состоянии	Присущее им архаическое мышление имеет все предпосылки рационального постижения реальности. Начало существования интеллектуальных способностей
Речевые способности имеются, но плохо развиты. Не мог произносить многие звуки современных языков	Речевые способности хорошо развиты. Мог свободно произносить все звуки современных языков
Каких-либо знаков или символов на стенах пещер, орудиях труда не обнаружено	Используют знаки и символы для обозначения вещей. Украшают орнаментом орудия труда. Рисунки на стенах пещер
Основной тип орудия варьировался не значительно	Орудия делятся более чем на сотни типов
Примитивная технология изготовления орудий, не требующая высокой квалификации, которой мог воспользоваться любой взрослый	Резкое повышение эффективности технологии изготовления орудий, потребовавшей значительной квалификации, высокого мастерства и времени на овладение ею
Признаков функциональной специализации не обнаружено	Деятельность приобретает характер профессии

Приведенные данные свидетельствуют о том, что неандерталец, который занимался, в основном, охотой и собирательством, активно использовал три уровня сенсомоторной активности (причем третий уровень явно был доминирующим), а кроманьонец, осуществивший переход к скотоводству и растениеводству, явно приступил к активизации и использованию четвертого уровня.

Результаты сравнения процесса изменения внешней детерминанты системы **человек** и процесса изменения внутренней детерминанты данной системы при переходе от неандертальцев к кроманьонцам представлены в таблице №2.7.

Таблица 2.7

Сравнение процессов изменение внешней и внутренней детерминант системы ЧЕЛОВЕК при переходе от неандертальцев к кроманьонцам

Этапы эволюции	Тип человека	Особенности жизни и деятельности (внешняя детерминанта)	Особенности сенсомоторной активности (внутренняя детерминанта)
палеолит (мезолит)	Неандертальцы (палеоантропы)	Непроизводящая деятельность, опирающаяся в основном на прошлый опыт (охота и собирательство). Отсутствует стабильное разделение труда	Преобладание чувственного отражения. Архаическое мышление на основе классифицирования по воспринимаемым признакам. Интерпретация неизвестного по аналогии с известным. Преобладание условнорефлекторной деятельности на основе прошлого опыта. Преобладающее использование символов (а не знаков), речевые способности слаборазвиты
неолит	Кроманьонцы (неоантропы)	Производящая деятельность, в основном опирающаяся на планирование (земледелие и скотоводство). Стабилизация разделения труда	Становление способности к абстрактному отражению. Развитое архаическое мышление с элементами рациональности. Формирование первичных понятий, логического мышления на основе простейших индуктивных и дедуктивных умозаключений. Становление деятельности на основе планирования и целеполагания. Использование символов и знаков для обозначения вещей, речевые способности хорошо развиты

Из приведенной таблицы хорошо видна взаимосвязь этих процессов. На этапе верхнего палеолита внешняя детерминанта живших тогда неандертальцев соответствовала непроизводящей деятельности. Данная внешняя детерминанта обуславливала наличие внутренней детерминанты в виде достаточно развитого третьего уровня сенсомоторной активности, которую, видимо, следует рассматривать и как текущую, и как предельную. В начале неолита внешняя детерминанта живших тогда людей изменилась и стала соответствовать производящей

деятельности. Данная внешняя детерминанта обуславливала появление внутренней детерминанты в виде становящегося (формирующегося) четвертого уровня сенсомоторной активности, которая для кроманьонцев была предельной. Текущая внутренняя детерминанта кроманьонцев представляла собой переходную фазу от доминирующего третьего уровня к доминирующему четвертому. Однако эта текущая внутренняя детерминанта кроманьонцев в большей степени соответствовала действующей в то время внешней детерминанте, чем предельная внутренняя детерминанта неандертальцев, что и обусловило уход последних с исторической сцены.

Если усложнение условий жизнедеятельности в рамках глобального кризиса приводит к активизации нового уровня сенсомоторной активности системы **человек**, то разумно предполагать, что современные глобальные проблемы цивилизации и, в первую очередь антропологический кризис, также приведут к изменению сенсомоторной активности данной системы.

Рассмотрим вероятное направление этого изменения.

При исследовании уровней сенсомоторной активности системы **человек**, являющихся ее подсистемами, других способов отражения, мышления и т.д., кроме четырех, представленных на рисунке 2.10, не выявлено. Что же тогда представляет собой упомянутое изменение сенсомоторной активности? Для того, чтобы лучше это понять рассмотрим существующую сегодня у человека проблему с сенсомоторной активностью.

Эта проблема в литературе описывается следующим образом. «Большая часть наших поступков определяется непроизвольными мотивами. Вся жизнь слагается из мелочей, которым мы непрерывно поддаемся и служим. Наше «я» непрерывно, как в калейдоскопе, меняется. Любое внешнее событие, поражающее нас, любая внезапно возникшая эмоция становится калифом на час, начинает строить и управлять, и, в свою очередь, неожиданно свергается и заменяется чем-то другим. А внутреннее сознание, не стремясь рассеять иллюзорность этого калейдоскопа и не понимая того, что сила, которая решает и действует, вовсе не оно само, ставит на всем свою подпись и говорит о разных моментах жизни, в которых действуют самые разные силы: «это я, и это я» [Успенский, 1993]. Здесь же по вопросу развития человека и его внутреннего роста говорится, что «действительный рост заключается в гармоническом развитии ума, чувств и воли». Кроме того, отмечается, что «вопрос о достижении единства – самый существенный вопрос внутреннего развития человека, если он

не достиг внутреннего единства, человек не может иметь никакого «я», лишен воли» [там же].

Таким образом, разумно предполагать, что упомянутое выше изменение сенсомоторной активности представляет собой не что иное, как становление функциональной целостности системы **человек**, возникающее вследствие гармонизации взаимодействия всех 4-х уровней сенсомоторной активности.

Результаты сравнения предполагаемого процесса изменения внешней детерминанты системы **человек** и предполагаемого процесса изменения внутренней детерминанты данной системы в связи с современными глобальными проблемами цивилизации представлены в таблице №2.8.

Таблица 2.8

Сравнение предполагаемого процесса изменения внешней детерминанты системы ЧЕЛОВЕК и предполагаемого процесса изменения внутренней детерминанты данной системы

Этапы эволюции	Тип человека	Особенности жизни и деятельности (внешняя детерминанта)	Особенности сенсомоторной активности (внутренняя детерминанта)
современный	Homo sapiens	Модели «неустойчивого развития» (индустриальная и сельскохозяйственная; капиталистическая и социалистическая). Разрушающая хозяйственная деятельность	Доминирование абстрактного отражения. Развитое словесно-логическое, рациональное мышления. Преобладание деятельности на основе планирования и целеполагания. Зависимость от результатов классифицирования по различным прагматическим признакам (парадигмы, идеологии и т.д.) и от привязанности к прошлому опыту (традициям). Естественные и искусственные языки. Отсутствие координации уровней сенсомоторной активности
ноосферный	Homo noosphere? (гармоничный, целостный)	Модели «устойчивого развития». Неразрушающая хозяйственная деятельность	Гармоничное взаимодействие уровней сенсомоторной активности. Целостное отражение и мышление с учетом результатов работы всех уровней. Отсутствие привязки к прошлому опыту, а также результатам рационализации и классифицирования

В настоящее время функциональной способностью человека (текущей внутренней детерминантой) является развитое абстрактное отражение, словесно-логическое мышление и моторная активность на основе целеполагания и планирования. Однако, поведение человека в настоящее время характеризуется отсутствием координированного, гармоничного, целостного взаимодействия уровней сенсомоторной активности, представляющих собой его подсистемы.

Данная внутренняя детерминанта соответствовала внешней детерминанте (функциональному запросу надсистемы), которая требовала организации и внедрения производящего хозяйства. В современных условиях внешняя детерминанта системы **человек** изменилась и представляет собой функциональный запрос на преодоление глобального кризиса и обеспечение перехода к устойчивой модели развития цивилизации (к *неразрушающему* хозяйству).

Анализ предыдущего эволюционного этапа системы **человек** позволяет предполагать, что современный кризисный переходный период развития цивилизации соответствует эволюционной смене вида homo sapiens на новый вид «*человека ноосферного*», который будет соответствовать изменившейся внешней детерминанте. Причем, учитывая постоянное временное уплотнение и ускорение исторического процесса, разумно предполагать, что происходить эта смена будет очень быстро, буквально в течение двух–трех десятилетий. Это подтверждает прогнозы различных специалистов, предсказывающих возникновение в ближайшее время катастрофических явлений, которые, как следует из проведенного анализа, предназначены для стимулирования смены отработавшего свое время вида человека на новый перспективный и расчистки, в прямом смысле этого слова, для нового человека жизненного пространства. Исходным материалом для новой системы («**человека ноосферного**») будут, очевидно, представители homo sapiens, способные обеспечить целостное гармоничное функционирование всех уровней сенсомоторной активности.

Целостным (системным, гармоничным) алгоритмом функционирования системы **человек** является единовременный учет в любой ситуации (см. рис. 2.10):

- состояния внешней и внутренней среды (подсистема №1);
- текущей обстановки (подсистема №2);
- накопленного прошлого опыта (подсистема №3);

- целей функционирования с учетом прогнозирования будущего развития и изменения ситуации (подсистема №4).

При этом результаты анализа показывают, что человек потенциально может функционировать более целостно с учетом гармоничного взаимодействия всех уровней сенсомоторной активности без доминирования какого-либо одного из них. Жизненные примеры Духовных Учителей доказали возможность такого развития человека [Ошо]. Эта возможность в настоящее время превращается в необходимость, без реализации которой современному человеку не преодолеть надвигающейся глобальной кризисной ситуации.

2.4. Системно-объектный подход к личной жизни

Системный подход, теория систем и системный анализ предназначены для решения сложных, слабо структурированных и плохо формализуемых проблем. Однако, явления, происходящие с человеком в процессе его личной жизни, оказываются не менее сложными. Решение личных проблем человека также не может быть найдено формальными средствами. Это вынуждает применять для решения личных проблем системный подход.

Используем понятия системно-объектного подхода для анализа и объяснения особенностей жизненных циклов мужчин и женщин [Маторин, 2014].

Жизненный цикл (ЖЦ) любой системы состоит из нескольких этапов. В литературе встречаются разные термины для обозначения этапов ЖЦ системы, но суть их от этого не меняется, что видно из графического представления ЖЦ (см. рис. 2.11). Естественно у каждого вида систем своя продолжительность этапов и свой достигаемый на них потенциал.

На рисунках 2.12 и 2.13 представлены графики ЖЦ среднестатистического мужчины и среднестатистической женщины, иллюстрирующие подтверждаемые статистикой факты более раннего взросления женщин и более быстрого старения мужчин. Для того, чтобы объяснить эти известные из психологии и антропологии факты с помощью упомянутых выше понятий системно-объектного подхода, необходимо проанализировать внешние детерминанты системы «Мужчина» и системы «Женщина».

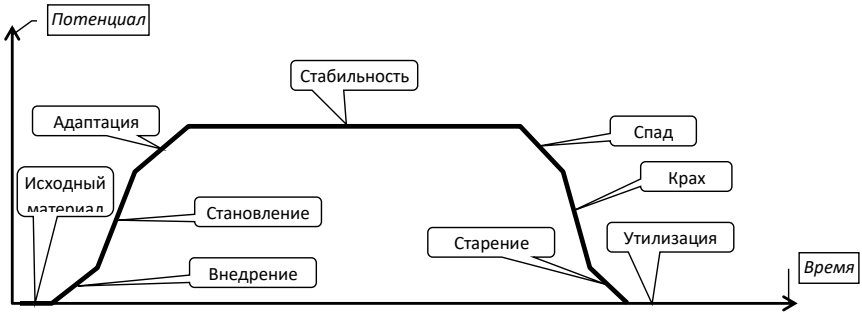


Рис. 2.11. Жизненный цикл системы

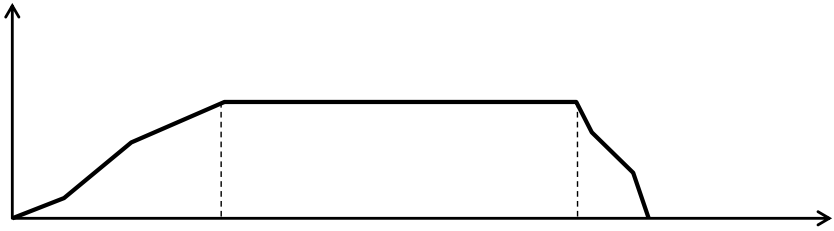


Рис. 2.12. Жизненный цикл мужчины

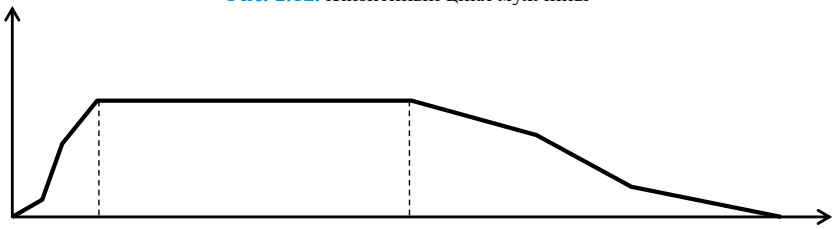


Рис. 2.13. Жизненный цикл женщины

Используем для этого представление внешней детерминанты системы «Человек» в виде схемы, изображенной на рисунке 2.14. Данная схема иллюстрирует тот факт, что любая система имеет, как правило, несколько надсистем. Система же «Человек» является частью систем «Семья», «Дело» и «Общество», которые выступают в роли надсистем, выставляющих системе «Человек» функциональные запросы («*семейно-бытовой*», «*деловой*» и «*общественный*»). Очевидно, что у разных групп людей количественные оценки этих запросов существенно различаются.

Из графика ЖЦ на рисунке 2.11 видно, что система переходит в стабильное состояние после завершения этапа адаптации к функциональному запросу. При этом очевидно, что адаптация к «*деловому*» запросу процесс длительный, так как требует (особенно, в наше время) соответствующей подготовки, которая включает в себя продолжительное обучение, приобретение профессионального опыта и т.д. Адаптация же к требованиям «*семейно-бытового*» запроса процесс, очевидно, менее длительный особенно для женщин с учетом изначального их физиологического соответствия этому запросу.

Для объяснения различий ЖЦ мужчин и женщин нам необходимо учесть и то обстоятельство, что традиционно (в нашем обществе) у женщин в общем запросе вес «*семейно-бытового*» функционального запроса, как правило, преобладает, а у мужчин, как правило, преобладает вес «*делового*» запроса. Это различие запросов, кроме того, естественно, вытекает из физиологических отличий мужчин и женщин.

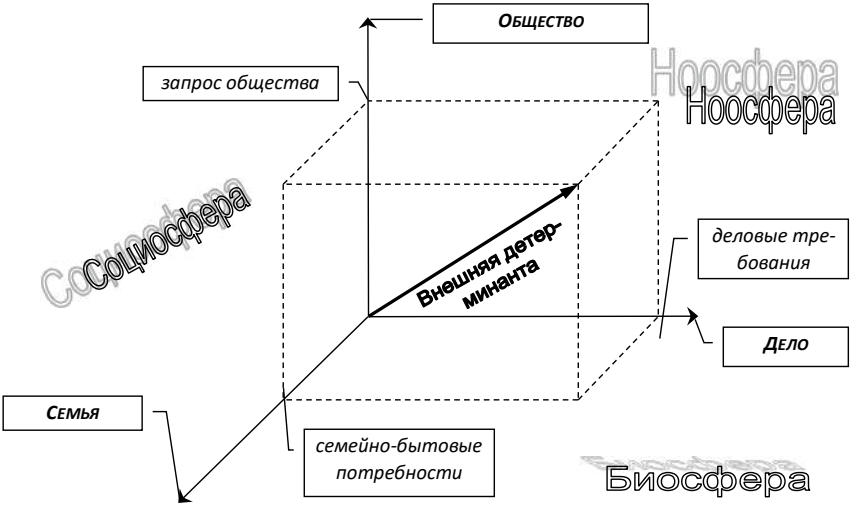


Рис. 2.14. Векторное представление внешней детерминанты Человека

Представленные рассуждения и приводят нас к системному пониманию причин различий левых частей ЖЦ мужчин и женщин, которые статистика констатирует своими средствами. Эти различия, состоящие в опережающем движении женщин к стабильному состоянию и отставании в этом вопросе мужчин, представляют собой системные закономерности для систем «Мужчина» и «Женщина»,

обусловленные различием времени адаптации систем к функциональным запросам соответствующих надсистем.

Явления, соответствующие правым частям ЖЦ мужчин и женщин, также обусловлены системными закономерностями, связанными, в данном случае, со снятием функциональных запросов для системы «Мужчина» и для системы «Женщина». Только у мужчин функциональный запрос снимается быстрее, чем у женщин. Это обусловлено, с одной стороны, тем, что снятие «*делового*» функционального запроса (более существенного для мужчин) после выхода на пенсию является более радикальным, чем снятие «*семейно-бытового*» (более существенного для женщин). С другой стороны, остатки «*семейно-бытового*» функционального запроса после взросления детей (например, в виде внуков) мужчинами (дедушками), как правило, воспринимаются в меньшей степени, чем женщинами (бабушками). Таким образом, мужчины вполне закономерно, как правило, движутся в сторону этапа утилизации быстрее, чем женщины.

Понимание системных закономерностей ЖЦ мужчин и женщин позволяет сформулировать ряд практически полезных для личной жизни человека рекомендаций.

Во-первых, можно обосновать временной промежуток, в котором партнерам имеет смысл создавать семью, вероятность распада которой будет минимальной. В настоящее время, по некоторым оценкам, эта вероятность достигает 80%. И поэтому среди соответствующих специалистов даже обсуждается проблема отмирания института семьи. Хотя, как станет ясно из дальнейшего изложения, проблема состоит всего лишь в том, что семья создается, как правило, без учета системных закономерностей. Во-вторых, можно обосновать системный способ продления жизни человека, действие которого опирается не на случайные факторы (питание, воспитание, экология и т.д.), а на фундаментальные (системные) причины его существования.

По поводу создания семьи заметим, что в соответствии с принципами системного подхода стабильность, устойчивость некоторой системы, создаваемой путем соединения других систем (подсистем), будет зависеть как от взаимодействия подсистем, так и от их собственной стабильности. Следовательно, если необходимо создать семью, вероятность распада которой будет минимальной (т.е. стабильную систему из двух подсистем), то лучше это делать тогда, когда оба партнера находятся на стабильном этапе своего ЖЦ. Если верить статистике в настоящее время женщины приходят к этому этапу в возрасте 20–25 лет, мужчины – в возрасте 35–40 лет. Это обусловлено всей системой

воспитания, образования, трудоустройства, профессионального и карьерного роста, а также психологией современного человека.

Мужчины и женщины с довольно раннего возраста готовы производить потомство, но не сразу готовы к семейной жизни, т.е. к созданию устойчивой системы, выставляющей совершенно новый для них функциональный запрос, так как не достигли собственной стабильности. Массовое создание семей без учета рассмотренной системной закономерности и приводит к массовому их распаду. По мнению автора, основная ответственность за несоблюдение этих «необходимых условий» должна лечь на плечи мужчин, так как именно они, в настоящее время, чаще всего, вступают в брак, не имея «за душой» ни профессиональной, ни материальной, ни психологической стабильности, а просто идя на поводу сиюминутной чисто физиологической потребности.

Сформулированные выше рекомендации по созданию семьи предлагается рассматривать как «необходимые условия» ее стабильности. Используя системно-объектный подход (УФО-подход) можно сформулировать «достаточные условия». Дело в том, что семья для государства является, в первую очередь, хозяйственной ячейкой общества, т.е. организационной системой.

Рассмотрим процедуру приема на работу на вакантную должность так, как это делается в настоящее время в солидных фирмах в соответствии с рекомендациями современных рекрутинговых консалтинговых компаний. При отборе кандидатов в таком случае осуществляется в первую очередь анализ желаний кандидата для выяснения причин, по которым он предлагает себя в качестве работника на данной должности в данной фирме. И только потом анализируются его возможности с точки зрения требований данной вакантной должности. Т.е. прежде чем выяснять возможности кандидата по выполнению «делового» функционального запроса со стороны вакантной должности в конкретной фирме, выясняются его желания выполнять именно этот запрос и причины их существования. Данные причины могут быть самыми различными и совершенно ясно, что они могут существенно сказываться на результатах деятельности независимо от квалификации кандидата. Например, желание работать в данной должности на данной фирме может быть обусловлено тем, что «а больше негде» или «удобством территориального расположения фирмы к месту жительства кандидата», а может быть обусловлено внутренними стремлениями кандидата к такому виду деятельности и приверженностью (а может и интересом) к данной фирме. Возможности кандидата, конечно, проверяются различными способами, но в первую очередь выясняются его

желания решать проблемы фирмы, жить ими. Иначе можно принять на работу квалифицированного, но не надежного сотрудника, который уйдет к конкурентам при первой возможности. Таким образом, с точки зрения системного подхода сначала анализируются узловые характеристики кандидата, а потом – функциональные. И только их общая удовлетворительная оценка позволяет кандидату стать объектом, занимающим ранее вакантный узел (должность) в структуре фирмы.

Процедуру создания семьи можно рассматривать как симметричную процедуру приема партнерами кандидата на вакантную должность жены и (или) мужа. И, следовательно, применить к этой процедуре тот же системно-объектный подход «Узел-Функция-Объект». В соответствии с этим подходом, как показано в приведенном выше примере приема на работу, необходимо сначала оценивать узловые характеристики кандидата, т.е. его желания, потребности решать проблемы семьи, удовлетворять требования партнера, а затем и возможности это делать.

Данный подход предлагает по-новому посмотреть на известную шутку о том, что брак по расчету может быть счастливым, если расчет правильный. Если в расчете учтены узловые характеристики кандидата и выявлены его желания путем создания семьи решать не свои собственные проблемы, а проблемы партнера, то, другими словами, учтены правильные отношения между людьми, а не только отношения между полами. Т.е. учтена настоящая любовь данного кандидата к партнеру. Примечательно, что это соответствует определению любви Э. Фроммом: «Любовь – это желание счастья объекту любви и деятельность по обеспечению этого счастья» [Фромм, 1990]. Если, при этом, анализ функциональных возможностей кандидата показывает, что он может выполнить функциональный запрос партнера, то тогда «достаточные условия» для создания семьи, с точки зрения данного партнера, оказываются выполненными. Эти условия должны быть выполнены с обеих сторон.

По мнению автора, за несоблюдение, как правило, описанных выше «достаточных условий» создания семьи ответственность в основном ложиться на женщин, которые зачастую стремятся замуж, не обращая внимания ни на какие жизненные обстоятельства и особенности своего партнера.

Таким образом, использование системного подхода и, в частности, учет «необходимых и достаточных условий» создания семьи, позволит избавиться человеку от многих личных проблем и значительно сократит количество разводов.

По поводу продолжительности жизни заметим, что в соответствии с принципами системного подхода система возникает вследствие наличия функционального запроса и существует, пока для нее существует хоть какой-то функциональный запрос, который она на должном уровне отрабатывает. Т.е., если система существует, то у нее есть один или несколько функциональных запросов, которым она в какой-то степени соответствует. Если это соответствие станет меньше некоторого минимального уровня или функциональные запросы по какой-либо причине будут сняты, то система начнет двигаться в сторону этапа утилизации. Используя именно эту системную закономерность, организационные системы обеспечивают продолжение своего существования путем формирования любыми путями потребности в результатах своей деятельности у клиентов, т.е. путем обеспечения наличия функционального запроса. Эта задача решается, в частности с помощью рекламы.

Такой же стратегии только без рекламы может придерживаться любой конкретный человек. Во-первых, любой человек после выхода на пенсию может начать участвовать в некоторой новой деятельности, в которой он востребован и из которой будет исходить для него новый «*деловой*» функциональный запрос. Во-вторых, любой человек может завести каким-либо образом новых детей или родственников, которым он будет действительно нужен, т.е. со стороны которых будет исходить для него новый «*семейно-бытовой*» функциональный запрос. И пока будет реально существовать функциональный запрос на этого человека со стороны надсистемы (или надсистем) и она будет «заинтересована» в человеке, который соответствует ее запросу и хорошо ее поддерживает, будет продолжаться и жизнедеятельность этого человека. Такой подход (системный) к продлению жизни использует саму причину существования человека. Если у человека не будет функционального запроса, все другие способы продления будут все равно бесполезны.

3. Формализация теории систем, основанной на системно-объектном подходе

3.1. Формальное описание системы как элемента «Узел-Функция-Объект»

Для системно-объектного подхода (УФО-подхода) предложено несколько способов его формализации с помощью:

1. Теории паттернов Гренандера [Маторин, 2002; Маторин, 2006].
2. Исчисления процессов Милнера в варианте ПИ-исчисления [Михелев, 2010].
3. Исчисления процессов Милнера на основе процессных графов (Calculus of Communication Systems – CCS) [Жихарев, 2011; Зимовец, 2012, 1].
4. Исчисления объектов Абади Кардели [Жихарев, 2013].

Рассмотрим суть и возможности этих способов.

1. Понятия *теории паттернов* Гренандера (РТ), в основе которой лежит графический формализм – *образующая* **g**, с помощью которого можно создавать более сложные конструкции – *конфигурации* **c** и *изображения* **I**, в определенной степени соответствуют понятиям системно-объектного УФО-подхода. При этом изображение **I**, как совокупность незамкнутых связей некоторой конфигурации **ext(c)**, хорошо моделирует узловую характеристику системы. Конфигурация **c**, как конструкция, перемыкающая незамкнутые связи изображения **I**, моделирует структурно-функциональную характеристику системы. Образующая **g**, как объект, обладающий некоторыми признаками **α**, со связями (в свою очередь характеризующимися некоторыми показателями **β**) – объектную (субстанциальную) характеристику данной системы.

Как было показано выше, категориальная иерархия классов (рис. 2.1) позволяет представить систему как УФО-элемента в виде кортежа **s = <u, *F*, *O*>**. Так как основными характеристиками узла **u** и функции ***F*** являются входные и выходные связи ***L***, можно данный картеж представить в виде **s = <(L)L, *L*(*L*), *O*>**, где **(L)L** – конкретный **узел** в структуре надсистемы (**(L)** – множество входных связей, **L** – множество выходных связей); ***L*(*L*)** – класс **функций**, балансирующих данный узел (**(*L*)** – множество аргументов, ***L*** – множество функций); ***O*** – класс субстанциальных характеристик.

Названное представление системы хорошо согласуется с упомянутым выше понятием образующей (см. рис. 2.15).

Рис. 2.15. Графические формализмы: образующая и УФО-элемент

В нашем случае, можно рассматривать классы ***O*** и ***L(L)*** как признаки образующей, а **(L)** и **L** как связи, показатели которых есть типы **L** (см. базовую классификацию связей). Таким образом, образующая $\mathbf{g_i}$ как УФО-элемент имеет вид: $\mathbf{g_i = <(L^i_1)L^i_2, L^i_2(L^i_1), O>}$.

Теория паттернов предполагает наличие *источника, генерирующего множество образующих* $\mathbf{G = \{g_i\}}$. В нашем случае таким источником, генерирующим УФО-элементы (образующие), и является показанная на рис. 2.1 категориальная иерархия классов. Для генерации образующих при решении конкретных задач необходимо иметь возможность формирования алфавитных символов (УФО-элементов, т.е. образующих), обладающих не абстрактной, а конкретной семантикой, соответствующей данной предметной области. При этом, естественно, должна существовать возможность специализации абстрактной концептуальной модели и построения модели конкретной предметной области. Алгоритм построения аналитической концептуальной модели конкретной предметной области основан на учете видов связей (подклассов класса **L**), обеспечивающих в рассматриваемой системе взаимодействие ее подсистем.

Данный вариант формализации понятия «система», как трехэлементной конструкции «Узел-Функция-Объект» позволяет обеспечивать агрегацию (синтез) и декомпозицию (анализ) систем формализованными средствами (см., например, работу [Маторин, 2006]). Однако, полноценно можно учесть только структурные свойства систем, используя *алгебру изображений*, входящую в теорию паттернов, которая позволяет соединять и разъединять изображения (как и образующие) на основании показателя взаимного соответствия связей **ρ**. Если же необходим учет функциональных свойств системы, то необходимо переходить к конфигурации образующих, раскрывающей внутреннюю структуру изображения. При этом может быть выявлена только структура функций/процессов. Учет же субстанциальных свойств системы сводится к рассмотрению самих образующих.

При этом образующая представляет собой просто часть конфигурации. Таким образом, целостного описания системы, учитывающего одновременно все характеристики (и функциональные, и субстанциальные, в том числе) с помощью теории паттернов не происходит. Так как на любом уровне, по сути дела, мы оперируем все равно только структурными характеристиками системы и одним и тем же абстрактным понятием образующей, которая раскрывается через образующие же.

2. С целью использования для формализации системно-объектного УФО-подхода *исчисления процессов* Милнера (CCS) система s_i как УФО-элемент представляется в виде следующего выражения: $s_i = <(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>$, где

$(L?_i, L!_i)$ – «Узел: **Us**» УФО-элемента, $L?_i \subset L$ – множество входных связей, $L!_i \subset L$ – множество выходных связей;

$(P_i, P^0_i, L\tau_i)$ – «Функция: **Fs**» УФО-элемента, где P_i – множество подпроцессов процесса, соответствующего «Функции», которые реализуются УФО-элементами нижнего яруса иерархии; $P^0_i \subset P_i$ – множество интерфейсных подпроцессов (входных $P?_i$ и выходных $P!_i$, причем $P^0_i = P?_i \cup P!_i$; в число входных связей $P?_i$ входит $L?_i$, в число выходных связей $P!_i$ входит $L!_i$); $L\tau_i$ – множество внутренних связей/переходов в P_i, осуществляемых путем передачи, ввода и вывода элементов глубинного яруса связанных подпроцессов;

$(n_i, \alpha_i, \beta?_i, \beta!_i)$ – «Объект: **Os**» УФО-элемента, где n_i – имя «Объекта» ($n_i \in N$); α_i – множество признаков «Объекта» n_i; $\beta?_i$ – множество показателей $L?_i$; $\beta!_i$ – множество показателей $L!_i$.

Такое представление системы, являющееся интеграцией РТ и CCS, позволило предложить по аналогии с исчислением процессов CCS для описания функций в рамках УФО-подхода *алгебру процессного подхода* (или *исчисление функций* УФО-элементов) [Зимовец, 2014]. При этом определены и описаны алгебраические операции на функциях элементов «Узел–Функция–Объект» по аналогии с операциями на процессах в исчислении процессов CCS. Применение операций исчисления функций позволяет формализовать процедуры декомпозиции и агрегации элементов (как с линейным порядком соединения, так и с порядком соединения «дерево») графоаналитических (визуальных) моделей. Сформулированные операции на функциях УФО-элементов могут быть использованы для алгебраического описания процессов в рамках любой графической нотации, соответствующей процессному подходу. См. пункт 4.2.

3. Интеграция с РТ Гренандера *ПИ-исчисления* Милнера позволяет представить функцию УФО-элемента в виде следующего выражения: $F = <c(x).P, \tau_p, t(y).P>$, где

$c(x)$ – входной префикс, получение данных x из канала c процессом P;

τ_p – внутренние действия процесса P, соответствующего функции F;

$t(y)$ – выходной префикс, передача данных y по каналу t процессом P.

В данном случае система s_i как УФО-элемента будет описываться следующим выражением:

$$s_i = <(L?_i, L!_i), ((L?_i(\beta?_i).P_i), \tau_p, (L!_i(\beta!_i).P_i)), (n_i, \alpha_i, \beta?_i, \beta!_i)>,$$

или по правилам ПИ-исчисления:

$$s_i = <(L?_i, L!_i), (L?_i(\beta?_i).\tau_p.L!_i(\beta!_i)), (n_i, \alpha_i, \beta?_i, \beta!_i)>, \text{ где}$$

$L?_i(\beta?_i)$ – входной префикс,

$L!_i(\beta!_i)$ – выходной префикс процесса P_i, а остальные обозначения представлены выше.

Оба упомянутых исчисления учитывают функциональные характеристики системы и связи процессов (функций) и не учитывают при этом ее субстанциальные (объектные) характеристики и структуру субстанции.

4. Следующий вариант формализации УФО-подхода использует *исчисление объектов* Абади-Кардели. В данном исчислении абстрактный объект представляет собой набор методов и полей. Использование метода – это вызов метода, изменение метода – это переопределение. Поле – частный случай метода. Изменение значения поля является частным случаем переопределения метода. Методы выполняются в контексте некоторого объекта, то есть имеют ссылку на объект. По аналогии с данным исчислением система s_i как УФО-элемент представляется в виде специального объекта:

$$s_i = [(L_i?, L_i!); f(L_i?)L_i!; (O_i?, O_i!, O_if)]? \text{ где:}$$

$L_i?$ – поле специального объекта для описания множества входящих интерфейсных потоков, соответствующих входящим связям системы s_i,

$L_i!$ – поле специального объекта для описания множества исходящих интерфейсных потоков, соответствующих выходящим связям системы s_i. Причем: $L_i? \subset L$ и $L_i! \subset L$, т.е. относятся к множеству всех связей L.

f – метод специального объекта, описывающий функцию системы s_i, т.е. процесс преобразования входящих интерфейсных потоков (входящих связей системы) $L_i?$ в выходящие $L_i!$. В соответствии с принятой в теории объектов манерой обозначений, метод объекта представляется в следующем виде: $f(L_i?)L_i!$, где **f** – метод объекта (функция/процесс системы s_i) с областью определения $L_i?$ и областью значений $L_i!$, соответственно.

$O_i?$ – множество полей, которое содержит интерфейсные входные характеристики специального объекта (системы s_i), $O_i!$ – множество полей, которое содержит интерфейсные выходные характеристики специального объекта (системы s_i), O_if – множество полей, которое содержит передаточные характеристики специального объекта (системы s_i). При этом множество полей для описания объектных характеристик системы $O_i = O_i? \cup O_i! \cup O_if$,

Графическое представление данного выражения показано на рис. 2.16. Этот непроизводный объект являться элементарным носителем информации в предлагаемом ниже исчислении систем как УФО-элементов.

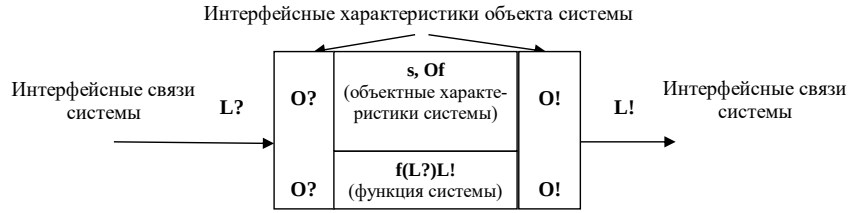


Рис. 2.16. Графический формализм системы как УФО-элемента.

Это определение учитывает не только субстанциальные (объектные) характеристики системы, но ее структурные и функциональные характеристики. Действительно, учтены поля для входных и выходных потоков **L?** и **L!**, которые являются частями надсистемы, т.е. заданы надсистемой, и определяют области определения и значения функции, которую должна выполнять система. Эта функция зафиксирована в методе $f(L?)L!$, который и определяет процесс преобразования входа в выход. Последний компонент $(O_i?, O_i!, O_if)$ определяет поля, задающие субстанциальные интерфейсные и другие характеристики объекта системы, и представляет собою набор статических объектных характеристик – константные поля. Данное формальное определение, кроме того, соответствует системно-объектному пониманию системы как *функционального объекта, функция которого обусловлена функцией*

объекта более высокого яруса [Мельников, 1978]. При этом метод (функциональная зависимость) может быть подробно описан, например, с помощью скрипта.

Данный способ представления системы позволил предложить основные операции с УФО-элементами в рамках исчисления систем как функциональных объектов.

3.2. Исчисление систем как функциональных объектов

3.2.1. Общие понятия и определения

Для создания упомянутого исчисления введем в рассмотрение множество потоковых объектов L, соответствующее множеству связей системы [Zhikharev, 2018].

$$L=\{l_1, l_2,\ldots,l_n\}, \quad (2.1)$$

где n – количество потоковых объектов (связей системы).

Каждый n-й элемент множества L представляет собою специальный потоковый объект (соответствующий конкретной связи системы), который в соответствии с *исчислением объектов* Абади-Кардели, состоит из полей и не включает методы и имеет следующий вид:

$$l_n=[r_1, r_2, \ldots ,r_k], \quad (2.2)$$

где:

$l_n \in L$;

k – количество полей потокового объекта l_n;

r_1, r_2, ..., r_k – поля потокового объекта, представляющие собой пару «идентификатор-значение».

Множество L при этом примет следующий вид:

$$L=\{l_1=[r_1^1, r_1^2,..., r_1^{k1}], l_2=[r_2^1, r_2^2,..., r_2^{k2}], ..., l_n=[r_n^1, r_n^2,..., r_n^{kn}]\} \quad (2.3)$$

Обозначим множество полей потокового объекта R, где:

$$R=\{r_i \mid r_i = [\text{идентификатор, значение}]\}, \; i = 1, 2, \ldots, k. \quad (2.4)$$

Таким образом, множество L потоковых объектов (связей системы) можно определить следующим образом:

$$L=\{l_i \mid l_i=[R_i]\}, \; i = 1, 2, \ldots, n \quad (2.5)$$

Согласно основным положениям методологии «Узел-Функция-Объект» в любой графоаналитической модели имеет место базовая иерархия связей, в нашем случае – потоковых объектов, причем

базовая иерархия связей имеет ряд предопределенных связей следующих видов [Matorin, 2018]:

- класс связей V, по которым «текут» вещественные ресурсы;
- класс связей E, по которым «текут» энергетические ресурсы;
- класс связей D, по которым передаются данные;
- класс связей C, по которым передается управляющая информация.

В соответствии с вышесказанным, введем в множество потоковых объектов системы L предопределенные потоковые объекты – классы, согласно базовой иерархии связей графоаналитического подхода «Узел-Функция-Объект» такие как:

- l^v – потоковый объект родитель, представляющий собой класс вещественных объектов;
- l^e – потоковый объект родитель, представляющий собой класс энергетических объектов;
- l^d – потоковый объект родитель, представляющий собой класс информационных объектов;
- l^c – потоковый объект родитель, представляющий собой класс информационных управляющий объектов.

Представленные выше базовые потоковые объекты не имеют полей, следовательно, имеют вид:

$$l=[R=\emptyset] \tag{2.6}$$

Все остальные элементы множества L не являющиеся базовыми элементами множества являются их наследниками, причем каждый дочерний потоковый объект наследует поля потокового объекта – родителя.

Рассмотрим множество потоковых объектов $L=\{l_1=[R_1], l_2=[R_2]\}$. Обозначим операцию наследования потокового объекта l_2 от потокового объекта l_1 знаком «::», тогда $l_2::l_1$ если $R_2 \subset R_1$ (потоковый объект l_2 является наследником по отношению к потоковому объекту l_1 если любое поле потокового объекта l_2 также является полем потокового объекта l_1). Далее введем обозначение элементов множества L, принадлежащих к определенному классу потоковых объектов как l^k, где верхний индекс k – соответствует классу потоковых объектов и, соответственно может принимать значения: v, e, d, c. Таким образом, для отдельной модели системы в терминах исчисления систем как функциональных объектов, множество потоковых объектов L можно представить в виде иерархии (см. рис. 2.17):

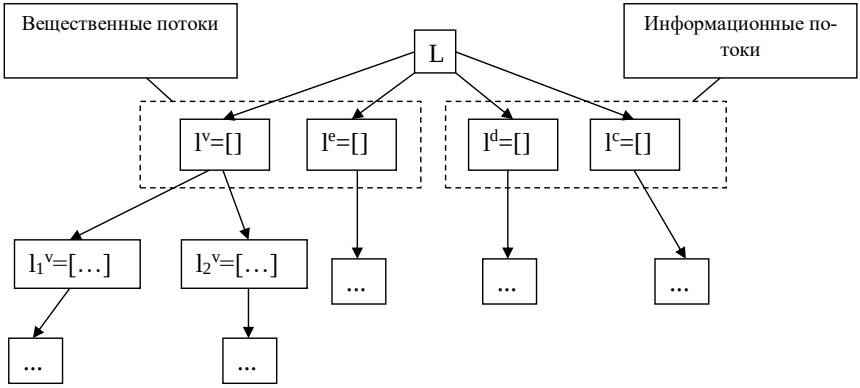


Рис. 2.17. Множество потоковых объектов системы в виде иерархии

Введем далее в рассматриваемое исчисление множество узловых объектов S, которое соответствует множеству систем как УФО-элементов.

$$S=\{s_1, s_2,...,s_m\}, \quad (2.7)$$

где m – количество узловых объектов (систем).

Каждый m-й элемент множества S представляет собой специальный узловой объект (соответствующий конкретной системе/УФО-элементу), который в соответствии с исчислением объектов Абади-Кардели состоит из полей и метода и имеет следующий вид:

$$s_m=[U, f, O], \text{ где:} \quad (2.8)$$

U – представляет собою множество полей для описания интерфейсных потоковых объектов узлового объекта s_m, соответствующих множеству функциональных связей данной системы. Множество $U = L_? \cup L_!$, где $L_?$ – представляет собою множество входящих интерфейсных потоковых объектов, соответствующих входящим связям системы, $L_!$ – представляет собою множество исходящих интерфейсных потоковых объектов, соответствующих выходящим связям системы. Причем: $L_? \subset L$; $L_! \subset L$.

f – представляет собой метод узлового объекта s_m, описывающий функцию преобразования входящих интерфейсных потоковых объектов (входящих связей системы) $L_?$ в выходящие – $L_!$.

Далее метод узлового объекта будем представлять в следующем виде:

$$f(L_?)L_!, \quad (2.9)$$

где f – метод узлового объекта (функция системы) с областью определения $L_?$ и областью значений $L_!$, соответственно.

O – представляет собою множество полей для описания объектных характеристик узлового объекта (системы) s_m, элементы которого имеют следующий формат:

$$O=\{o_i \mid o_i=[\text{идентификатор, значение}]\},\ i = 1, 2, \ldots, p. \quad (2.10)$$

Множество полей для описания объектных характеристик системы состоит из трех подмножеств:

$$O=O_?\cup O_!\cup O_f \quad (2.11)$$

Множество полей $O_?$ содержит интерфейсные входные характеристики узлового объекта. Для каждого поля каждого входного потокового объекта в множестве $O_?$ содержится соответствующий экземпляр вида (2.10). Таким образом, если, например, множество входящих потоковых объектов состоит из одного элемента (потокового объекта l_1), а множество полей входящего потокового объекта состоит из двух элементов следующего вида:

$$L_?=\{l_1=[r_1, r_2]\}. \quad (2.12)$$

Тогда соответствующее множество $O_?$ примет вид:

$$O_?=\{o_1, o_2\}. \quad (2.13)$$

Мощность множества $O_?$ будет зависеть от количества входящих интерфейсных потоковых объектов и количества их полей. Если мощность множества:

$$|L_?|=n, \quad (2.14)$$

а мощности входных потоковых объектов:

$$|l_?^1|= m_1, |l_?^2| = m_2, \ldots, |l_?^n| = m_n, \quad (2.15)$$

тогда мощность соответствующего множества интерфейсных характеристик объекта $O_?$ будет равна:

$$|O_?| = \sum_1^n |l_?^n| \quad (2.16)$$

Мощность множества $O_!$ (соответствует выходных интерфейсным потоковым объектам), по аналогии с выражением (2.16) вычисляется по формуле:

$$|O_!| = \sum_1^n |l_!^n| \quad (2.17)$$

Множество O_f содержит объектные характеристики системы, присущие объекту реализующему функцию, и их количество будет зависеть от конкретной системы.

Таким образом, систему в рамках исчисления систем, описанную выражением (2.8) будем представлять в виде следующего выражения:

$$s_m=[L_?, L_!; f(L_?)L_!; O_?, O_!, O_f]. \quad (2.18)$$

Графическое представление выражения (2.18) показано на рисунке 2.16.

Данное представление будем далее рассматривать как графический формализм, по аналогии с графическим формализмом – образующей – в теории паттернов Гренандера. Этот непроизводный объект будет являться элементарным носителем информации в рассматриваемом исчислении.

3.2.2. Элементарные структурные операции исчисления систем

Далее рассмотрим элементарные операции [Matorin, 2018] над определенными выше множествами потоковых и узловых объектов, с помощью которых в перспективе будет обеспечиваться возможность организации вычислений в рамках разрабатываемой формальной системной теории.

Введем в рассматриваемое исчисление элементарные структурные операции над множествами потоковых объектов L и узловых объектов S. В качестве элементарных операций над множеством потоковых объектов предлагаются операции, представленные ниже.

Операция добавления поля r_0 потоковому объекту l.

Пусть даны – потоковый объект l=[R], где R={r_1, r_2,…, r_k} и поле r_0, для которого справедливо {r_0}= R_0 .Тогда операция добавления нового поля r_0 потоковому объекту l будет соответствовать объединению множеств R и R_0.

$$l.r_0 >> l=[r_1, r_2 ,\ldots, r_k]\rightarrow l^*=[r_0, r_1, r_2,\ldots, r_k] \quad (2.19)$$

Операция удаления из потокового объекта l поля r_0.

Пусть дан потоковый объект l=[R], где R= {r_0, r_1, r_2,…, r_k} и поле r_0, для которого справедливо {r_0}= R_0.Тогда результат операции удаления поля r_0 из потокового объекта l будет соответствовать разности множеств R и R_0. Применяя указанные выше обозначения, получим:

$$l.r_0 << l=[r_0, r_1, r_2,\ldots,r_k] \rightarrow l^* =[r_1, r_2,\ldots, r_k] \quad (2.20)$$

Операция переопределения поля r_0 потокового объекта l.

Пусть дан потоковый объект l=[R], где R= {r_0, r_1, r_2,…, r_k} и поле r_0, для которого справедливо {r_0} = R_0, а также поле r_0*, для которого справедливо {r_0*} = R_0*. Тогда результат операции переопределения

поля r_0 потокового объекта l будет соответствовать объединению множества R_0* с разностью множеств R и R_0:

$$l.r_0 \Leftarrow l = [r_0, r_1, r_2, \ldots, r_k] \rightarrow l^* = [r_0^*, r_1, r_2, \ldots, r_k] \quad (2.21)$$

Операция переопределения метода f узлового объекта s.

Пусть дан узловой объект s=[U, f, O], где $U = L_? \cup L_!$; $f(L_?)L_!$ и метод $f^*(L^*_?)L^*_!$, для которого $L_?^* \subset U$ и $L_!^* \subset U$. Тогда результат операции переопределения метода f в объекте s будет определяться по аналогии с подобной операцией в исчислении объектов Абади-Кардели следующим выражением:

$$s.f \Leftarrow f(L_?)L_! \rightarrow f^*(L^*_?)L^*_! \quad (2.22)$$

Операция соединения узловых объектов (рис 2.18).

Даны два объекта s_i и s_j такие, что: $s_i=[L_{?i}, L_{!i}; f(L_{?i})L_{!i}; O_{?i}, O_{!i}, O_{fi}]$; $s_j=[L_{?j}, L_{!j}; f(L_{?j})L_{!j}; O_{?j}, O_{!j}, O_{fj}]$. Правило переопределения полей и метода объекта j в случае присоединения этого объекта к объекту i (вызов метода объекта j объектом i: $s_j.f \rightarrow L_{!j}\{L_{!i} \rightarrow L_{?j} | s_j\}$), если $L_{?j} \equiv L_{!i}$ и $O_{?j} R O_{!i}$, сведется к следующему выражению:

$$s_j.f \Leftarrow f(L_{?j})L_{!j} \rightarrow f(L_{?i})L_{!j};\ L_{?j} \rightarrow L_{?i};\ O_{?j} \rightarrow O_{?i},\ O_{fij} \quad (2.23)$$

Получаем объект:

$$s_{ij} = [L_{?i}, L_{!j}; f(L_{?i})L_{!j}; (O_{?i}, O_{!j}, O_{fij})] \text{ (см. рис. 2.18)} \quad (2.24)$$

Правило переопределения полей и метода объекта i в случае присоединения этого объекта к объекту j (вызов метода объекта i объектом j: $s_i.f \rightarrow L_{!i}\{L_{!j} \rightarrow L_{?i} | \rightarrow s_i\}$), если $L_{?i} \equiv L_{!j}$ и $O_{?i} R O_{!j}$, сведется к следующему выражению:

$$s_i.f \Leftarrow f(L_{?i})L_{!i} \rightarrow f(L_{?j})L_{!i};\ L_{?i} \rightarrow L_{?j};\ O_{?i} \rightarrow O_{?j},\ O_{fji} \quad (2.25)$$

Получаем объект:

$$s_{ji} = [L_{?j}, L_{!i}; f(L_{?j})L_{!i}; O_{?j}, O_{!i}, O_{fji})] \quad (2.26)$$

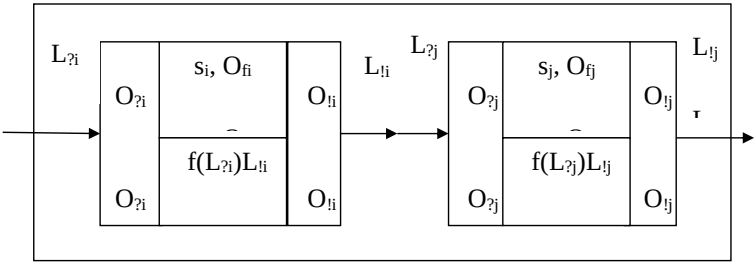


Рис. 2.18. Присоединение объекта s_j к объекту s_i

Также данную операцию можно описать в терминах теоретико-множетсвенного подхода. Пусть даны два узловых объекта (две системы) s_i и s_j такие, что:

$$s_i = [(L?_i, L!_i); f(L?_i)L!_i; (O?_i, O!_i, Of_i)]; s_j = [(L?_j, L!_j); f(L?_j)L!_j; (O?_j, O!_j, Of_j)].$$

Эти объекты можно соединить, если

$$L?_j \equiv L!_i \text{ и } O?_j \text{Ac} O!_i$$

(где Ас – соответствие интерфейсных характеристик систем s_i и s_j в смысле их комплиментарности) При этом s_i должно присоединяться к входу s_j с учетом трех системных аспектов: структурного, функционального и субстанциального.

Структурно система s_{ij}, образующаяся в результате упомянутого присоединения, имеет входы s_i и выходы s_j. Данный результат формально можно получить путем объединения потоков/связей системы s_i и потоков/связей системы s_j с последующим вычитанием из этого объединения результата пересечения потоков/связей s_i и s_j. Это позволит удалить из рассмотрения на уровне нового соединенного узлового объекта (системы s_{ij}) внутренние связи, за счет которых и происходит присоединение s_i к входу s_j. Таким образом, узел

$$U_{ij} = ((L?_i, L!_i) \cup (L?_j, L!_j)) \setminus ((L?_i, L!_i) \cap (L?_j, L!_j)) = (L?_i, L!_j).$$

Функционально система s_{ij} преобразует входной поток $L?_i$ системы s_i в выходной поток $L!_j$ системы s_j. Взаимодействие процессов систем s_i и s_j происходит вследствие того, что процесс s_j использует результаты процесса s_i, так как по условию $L?_j \equiv L!_i$.Таким образом, функция

$$f_{ij} = f_j(f_i(L?_i))L!_j = f_{ij}(L?_i)L!_j.$$

Субстанциально система s_{ij} имеет входной интерфейс $O?_i$ системы s_i и выходной интерфейс $O!_j$ системы s_j. Кроме того, собственно объектные характеристики системы s_{ij} образуются из характеристик Of_i и Of_j. Формально это можно получить путем объединения всех объектных характеристик системы s_i и объектных характеристик системы s_j с последующим вычитанием из этого объединения результата пересечения объектных характеристик s_i и s_j. Это позволит удалить из рассмотрения на уровне нового соединенного узлового объекта (системы s_{ij}) внутренние интерфейсные объектные характеристики, за счет которых происходит присоединение s_i к s_j, по аналогии с потоковыми характеристиками (считая соответствующие друг другу характеристики равными).

Таким образом, объект

$$O_{ij} = ((O?_i, O!_i, Of_i) \cup (O?_j, O!_j, Of_j) \setminus (O?_i, O!_i, Of_i) \cap (O?_j, O!_j, Of_j)) =$$
$$= (O?_i, O!_j, Of_i \cup Of_j) = (O?_i, O!_j, Of_{ij}).$$

Для операции соединения узловых объектов (систем), с учетом манеры обозначения в исчислении процессов Милнера (присоединяемый процесс рассматривается как префикс), введем обозначение: $s_i.s_j$. При этом результат данной операции можно представить следующим образом:

$$s_i.s_j = s_{ij} = [U_{ij}, f_{ij}, O_{ij}] = [(L?_i, L!_j); f_{ij}(L?_i)L!_j; (O?_i, O!_j, Of_{ij})].$$

Таким образом, операция соединения узловых объектов (систем) друг к другу сводится к следующей процедуре:

- узел объекта (системы), получающегося после присоединения, определяется входами присоединяемого объекта и выходами объекта, к которому происходит присоединение;
- функция получающегося после присоединения объекта (системы) определяется как функциональная зависимость выходного потока объекта, к которому происходит присоединение, от входного потока присоединяемого объекта;
- субстанция объекта (системы), получающегося после присоединения, определяется входным интерфейсом присоединяемого объекта, выходным интерфейсом объекта, к которому происходит присоединение, и объединением передаточных характеристик.

Таким образом, например, два узловых объекта (две системы) s_{ij} и s_k такие, что: $s_{ij} = [(L?_i, L!_j); f_{ij}(L?_i)L!_j; (O?_i, O!_j, Of_{ij})]$ и $s_k = [(L?_k, L!_k); f(L?_k)L!_k; (O?_k, O!_k, Of_k)]$, можно соединить, если $L?_k \equiv L!_j$ *и* $O?_k \text{Ac} O!_j$. Результат присоединения s_{ij} к s_k будет выглядеть следующим образом: $s_{ij}.s_k = s_{ijk} = [U_{ijk}, f_{ijk}, O_{ijk}] = [(L?_i, L!_k); f_{ik}(L?_i)L!_k; (O?_i, O!_k, Of_{ik})] = s_{ik}$.

Как видно из рисунка 2.18, операция присоединения одного узлового объекта к другому предполагает создание третьего узлового объекта, однако в практике моделирования часто встречается необходимость коммуникации двух узловых объектов без создания третьего, объединяющего исходные. Поэтому имеет смысл рассмотреть разновидность данной операции.

Связывание двух узловых объектов.

Пусть даны два узловых объекта: s_i и s_j такие, что: $s_i=[L_{?i}=\emptyset, L_{!i}=\emptyset; f(L_{?i})L_{!i}; O_{?i}, O_{!i}, O_{fi}]$; $s_j=[L_{?j}=\emptyset, L_{!j}=\emptyset; f(L_{?j})L_{!j}; O_{?j}, O_{!j}, O_{fj}]$ и потоковый объект $l=[R]$, тогда операция связывания двух узловых объектов

потоковым объектом запишем в следующем виде: $s_i \xrightarrow{l} \rightarrow s_j \leftrightarrow l \gg s_i.L_!; l \gg s_j.L_?$

То есть операция связывания двух узлов потоковым объектом эквивалентна добавлению в качестве интерфейсного потокового объекта l узловому объекту s_i как исходящего и s_j как входящего.

Объединение объектов по входу (рис. 2.19).

Даны два объекта s_i и s_j, у которых $L_{?i} \equiv L_{?j}$ и $O_{?i}=O_{?j}$. Правило переопределения полей и метода в таком случае сводится к следующим двум вариантам:

$$1).\ s_i.f \Leftarrow f(L_{?i})L_{!i} \rightarrow f(L_{?i})L_{!i},L_{!j}];\ L_{!j};\ O_{!j},\ O_{fij} \qquad (2.27)$$

$$2).\ s_j.f \Leftarrow f(L_{?j})L_{!j} \rightarrow f(L_{?i})L_{!i},L_{!j}];\ L_{?j} \rightarrow L_{?i};\ L_{!i};\ O_{?j} \rightarrow O_{?i};\ O_{!i},\ O_{fij} \qquad (2.28)$$

Независимо от варианта получаем объект:

$$s_{ij} = [L_{?i},\ L_{!i},\ L_{!j};\ f(L_{?i})L_{!i},L_{!j};\ O_{?i},\ O_{!i},\ O_{!j},\ O_{fij})] \text{ (см. рис. 2.19)} \qquad (2.29)$$

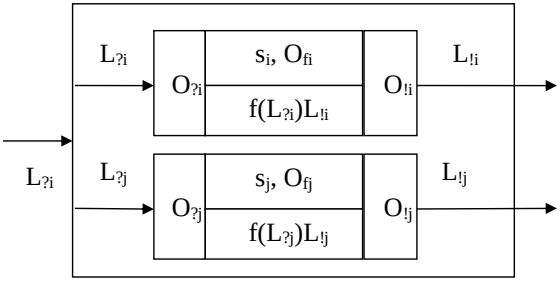


Рис. 2.19. Объединение по входу

Теоретико-множественное описание данной операции сводится к следующим формулировкам. Даны два узловых объекта (две системы) s_i и s_j, у которых $L?_i \equiv L?_j$ и $O?_i = O?_j$. Эти объекты могут быть объединены по входу. Они должны объединяться с учетом трех системных аспектов: структурного, функционального и субстанциального.

Структурно система s_{ij}, образующаяся в результате упомянутого объединения, имеет вход, соответствующий одинаковым входам систем s_i и s_j, и выходы, соответствующие выходам систем s_i и s_j. Формально это можно получить путем объединения потоков/связей системы s_i и потоков/связей системы s_j.

Таким образом, узел

$$U_{ij} = ((L?_i, L!_i) \cup (L?_j, L!_j)) = (L?_i, L!_i, L!_j).$$

Функционально система s_{ij} преобразует входной поток $L?_i$ (он же $L?_j$) в выходные потоки $L!_i$ и $L!_j$ систем s_i и s_j путем параллельного выполнения процессов этих систем. Таким образом, функция

$$f_{ij} = f_i(L?_i)L!_i \wedge f_j(L?_j)L!_j = f_{ij}(L?_i)L!_i,L!_j.$$

Субстанциально система s_{ij} имеет один входной интерфейс $O?_i$ (он же $O?_j$) и два выходных: $O!_i$ и $O!_j$). Кроме того, собственно объектные характеристики системы s_{ij} образуются из характеристик Of_i и Of_j. Формально это можно получить путем объединения всех объектных характеристик системы s_i и объектных характеристик системы s_j.

Таким образом, объект

$$O_{ij} = ((O?_i, O!_i, Of_i) \cup (O?_j, O!_j, Of_j)) = (O?_i, O!_i, O!_j, Of_i \cup Of_j) = (O?_i, O!_i, O!_j, Of_{ij}).$$

Для операции объединения узловых объектов (систем) по входу введем обозначение: $s_i.\cap s_j$. При этом в результате выполнения данной операции получается объект (система) разветвления потоков (обозначим $s^<$), который можно представить следующим образом:

$$s_i.\cap s_j = s^<_{ij} = [U_{ij}, f_{ij}, O_{ij}] = [(L?_i, L!_i, L!_j); f_{ij}(L?_i)L!_i,L!_j; (O?_i, O!_i, O!_j, Of_{ij})].$$

Объединение объектов по выходу (рис. 2.20).

Даны два объекта s_i и s_j, у которых $L_{!i} \equiv L_{!j}$ и $O_{!i} = O_{!j}$. Правило переопределения полей и метода в таком случае сводится к следующим двум вариантам:

$$1).\ s_i.f \Leftarrow f(L?_i)L!_i \rightarrow f(L_{?i},L_{?j})L_{!i}];\ L_{?j};\ O_{?j},\ O_{fij} \qquad (2.30)$$

$$2).s_j.f \Leftarrow f(L?_j)L!_j \rightarrow f(L_{?i},L_{?j})L_{!i}];\ L_{?i};\ L_{!j} \rightarrow L_{!i};\ O_{!j} \rightarrow O_{!i},\ O_{fij} \qquad (2.31)$$

Независимо от варианта получаем объект:

$$s_{ij} = [L_{?i},\ L_{?j},\ L_{!i};\ f(L_{?i},L_{?j})L_{!i};\ O_{?i},\ O_{?j},\ O_{!i},\ O_{fij})] \text{ (см. рис. 2.20)} \qquad (2.32)$$

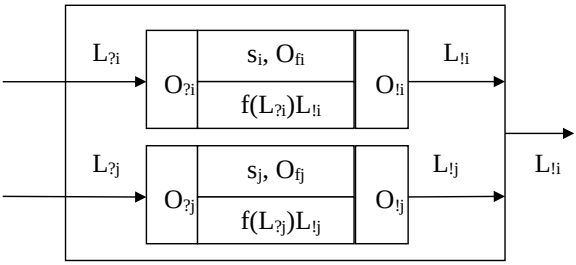


Рис. 2.20. Объединение по выходу

По аналогии с предыдущей описанной операцией, рассмотрим теоретико-множественное представление операции объединения по

выходу. Даны два узловых объекта (две системы) s_i и s_j, у которых $L!_i \equiv L!_j$ и $O!_i = O!_j$. Эти объекты могут быть объединены по выходу. Они должны объединяться также с учетом трех системных аспектов: структурного, функционального и субстанциального.

Структурно система s_{ij}, образующаяся в результате упомянутого объединения, имеет выход, соответствующий одинаковым выходам систем s_i и s_j, и входы, соответствующие входам систем s_i и s_j. Формально это можно получить путем объединения потоков/связей системы s_i и потоков/связей системы s_j.

Таким образом, узел

$$U_{ij} = ((L?_i, L!_i) \cup (L?_j, L!_j)) = (L?_i, L?_j, L!_i).$$

Функционально система s_{ij} преобразует входные потоки $L?_i$ и $L?_j$ систем s_i и s_j в выходной поток $L!_i$ (он же $L!_j$) путем параллельного выполнения процессов этих систем.

Таким образом, функция

$$f_{ij} = f_i(L?_i)L!_i \wedge f_j(L?_j)L!_j = f_{ij}(L?_i, L?_j)L!_i.$$

Субстанциально система s_{ij} имеет один выходной интерфейс $O!_i$ (он же $O!_j$) и два входных: $O?_i$ и $O?_j$. Кроме того, собственно объектные характеристики системы s_{ij} образуются из характеристик Of_i и Of_j. Формально это можно получить путем объединения всех объектных характеристик системы s_i и объектных характеристик системы s_j.

Таким образом, объект

$$O_{ij} = ((O?_i, O!_i, Of_i) \cup (O?_j, O!_j, Of_j)) = (O?_i, O?_j, O!_i, Of_i \cup Of_j) = (O?_i, O?_j, O!_i, Of_{ij}).$$

Для операции объединения узловых объектов (систем) по выходу введем обозначение: $s_i \cap . s_j$. При этом в результате выполнения данной операции получается объект (система) слияния потоков (обозначим $s^{>}$), который можно представить следующим образом:

$$si \cap .sj = s{>}ij = [Uij, fij, Oij] = [(L?i, L?j, L!i); fij(L?i, L?j)L!i; (O?i, O?j, O!i, Ofij)].$$

Описанные три операции (соединение объектов, объединение объектов по входу и объединение объектов по выходу) рассматриваются как базовые структурные операции предлагаемого исчисления. Они соответствуют трем структурным явлениям и трем видам объектов, из которых может быть создана любая структура и система любой сложности: простой поток (простой объект), слияние потоков (объект слияния) и разветвление потока (объект разветвления). По сути дела, эти операции сводятся к описанию изображения, получаемого путем

построения конфигурации из непроизводных объектов (графических формализмов) и описания незамкнутых связей.

Все остальные взаимодействия УФО-элементов как специальных объектов могут быть получены путем комбинирования базовых операций.

3.2.3. Основы моделирования функционирования системы во времени

Рассмотрим возможности формального описания функционирования моделируемой системы в терминах «Узел-Функция-Объект». Как было сказано выше, функциональные особенности системы представлены в виде функции узлового объекта $f(L_?)L_!$, которая описывает процесс преобразования входных потоковых объектов в выходные. Рассмотрим общий случай описания функции системы. Как говорилось ранее, любая моделируемая система представляет собою или будет представлять собою в будущем определенную часть реального мира с учетом ее взаимосвязи с другими системами, с учетом цели ее существования и т.д., а соответственно существует во времени. Учитывая этот факт, представим модель системы в виде уточнения выражения (2.8):

$$s(t)=[U(t), f(t), O(t)] \qquad (2.33)$$

В представленном выражении система как потоковый объект представляет собою функцию по времени t, т.е. выражение (2.33) описывает существование системы во времени и показывает ее изменчивость с течением времени. Компоненты системы: узел – перекресток входящих и исходящих связей, обозначим как U(t), функцию системы обозначим как f(t), объект, реализующий функцию – O(t). Смысл данных обозначений состоит в том, что с течением времени система может изменяться с учетом изменения всех ее компонент, например, могут меняться интерфейсные потоковые объекты системы (их набор и структура), может изменяться функция системы, в зависимости от требований надсистемы, может изменяться и объект, реализующий функцию системы.

Введем в рассматриваемой исчисление понятие «состояние» системы или в случае рассмотрения модели – состояние узлового объекта. Если состояние узлового объекта s – это набор фиксированных значений полей интерфейсных потоковых объектов $L_?$ и $L_!$ в заданный момент времени t, тогда процесс функционирования системы s можно представить как процесс изменения состояния системы в различные

моменты времени t_1,t_2,.. Обозначим множество всевозможных состояний системы как C, получим зависимость:

$$C=f(t), \tag{2.34}$$

где f – функция рассматриваемого узлового объекта.

Выражение (2.34) в дальнейшей работе будем понимать, как закон функционирования узлового объекта, согласно которому в каждый конкретный момент времени t_n, система, модель которой представляет узловой объект, принимает состояние c_n вследствие ее функционирования. Графически, закон функционирования узлового объекта можно представить в виде графика, как показано на рисунке 2.21.

По оси абсцисс находятся временные отрезки, по оси ординат – всевозможные состояния системы, таким образом, график изменения состояний системы во времени и представляет собою функционирование системы. И главной задачей, решение которой позволит строить адекватные модели функционирующих систем – является задача моделирования выражения (2.34), а именно функции f.

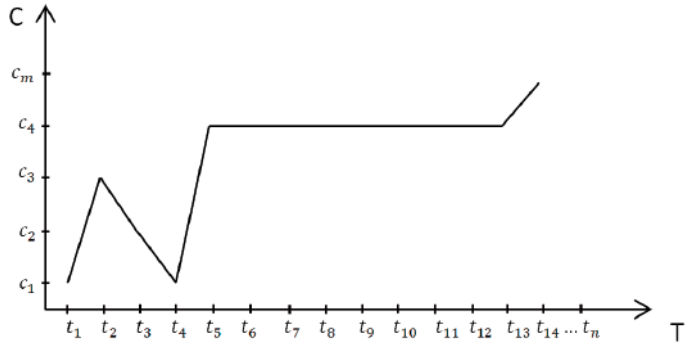


Рис. 2.21. График закона функционирования узлового объекта.

Если набор входных потоковых объектов $L_?=[l_{?1}, l_{?2}, \ldots, l_{?n}]$, а набор исходящих потоковых объектов $L_!=[l_{!1}, l_{!2},\ldots,l_{!m}]$, тогда запишем функцию узлового объекта следующим выражением:

$$f(L_?,t)L_! = \begin{cases} l_{!1}.r^*: l_{!1}.r^* = \varphi_1(L_?);\ \text{delay} = t_1; \\ l_{!2}.r^*: l_{!2}.r^* = \varphi_2(L_?);\ \text{delay} = t_2; \\ \ldots \\ l_{!m}.r^*: l_{!m}.r^* = \varphi_m(L_?);\ \text{delay} = t_m; \end{cases} \tag{2.35}$$

где, оператор delay – моделирует временную задержку на преобразование входа в выход продолжительностью t_m , а функция φ_m представляет

собой однозначную математическую зависимость поля выходного потокового объекта $l_{!1}.r^*$ от набора входных значений $L_?$.

Рассмотрим для примера абстрактную систему, у которой в качестве входного потокового объекта имеется потоковый объект с одним полем $l_?=[x]$ и выходящий потоковый объект с одним полем $l_!=[y]$, в то же время возьмем за условие, что поле у исходящего потокового объекта зависит от входа по следующему закону:

$$l_!.y = a + l_?.x^b, \tag{2.36}$$

и на одну итерационную процедуру тратится время *t=2*a*, причем постоянные параметры a и b представляют собой объектные характеристики системы, тогда метод узлового объекта можно записать в виде:

$$f(l_?)l_! = \{l_!.y : l_!.y = a + l_?.x^b; delay = 2 * a; \tag{2.37}$$

Причем время задержки, как видно из выражений 2.36 и 2.37 также может иметь определенную функциональную зависимость, например от объектных характеристик узла.

Рассмотрим пример моделирования существующей системы. В качестве системы будем рассматривать термоформовочную машину HSC-660A, предназначенную для производства пластиковой тары. Характеристики данного технического устройства следующие: ширина пластикового полотна – 600 мм; толщина – 2.5 мм; производительность – 34 цикла в минуту; длинна потребляемого полотна на один цикл – 250 мм, количество пластиковых стаканчиков за один цикл – 12. Данные характеристики говорят о том, что за одну минуту термоформовочная машина рассматриваемой модели «выдавливает» 34x12=408 пластиковых стаканчиков и потребляет на это 250x34=8500 мм. пластикового полотна. Для данного случая введем в рассмотрение два материальных потоковых объекта:

$$L=[l_1=[r_1=600, r_2=2.5, r_3=100]; l_2=[r_1=0]], \tag{2.38}$$

где: l_1^v – потоковый объект, представляющий собою рулон пленки, шириной – $l_1^v.r_1=600$ миллиметров, толщиной – $l_1^v.r_2=2.5$ миллиметра, длина которого – $l_1^v.r_3=100$ метров; l_2^v – потоковый объект, представляющий собою готовую продукцию – пластиковые стаканчики с одним полем $l_2^v.r_1$ – количество пластиковых стаканчиков, начальное значение равно нулю. Далее введем в модель следующий узловой объект:

$$S=[s_1=[L_?, L_!; f(L_?)L_!; O_?, O_!, O_f]], \tag{2.39}$$

где: $L_?=[l_1]$; $L_!=[l_2]$; $O_?=\varnothing$; $O_! =\varnothing$; $O_f=[o_{f1}=250, o_{f2}=12, o_{f3}=34]$. Множество объектных характеристик O_f содержит технические показатели агрегата – длина пластикового полотна на один цикл, количество готовых стаканчиков на одном цикле работы и производительность (количество в минуту), соответственно. Разберем подробнее метод узлового объекта. В соответствии с выражением (2.32), метод рассматриваемого узлового объекта представим в виде:

$$f(L_?, t)L_! = \{l_2.r_1 : l_2.r_1 = \varphi_1(l_1);\ delay = t; \tag{2.40}$$

В соответствии с техническими характеристиками рассматриваемого агрегата видно, что между входом и выходом имеется следующая зависимость:

$$l_2.r_1 = \frac{o_{f2} * l_1.r_3}{o_{f1}} \tag{2.41}$$

Представленная зависимость действительна при выполнении условий: $l_1.r_1=600$ и $l_1.r_2=2.5$. Условия показывают то, что рулон исходного сырья должен соответствовать техническим характеристикам термоформовочной машины. Время в минутах, затраченное на выдавливание $l_2.r_1$ стаканчиков, тогда будет равно:

$$t = \frac{l_2.r_1}{o_{f2} * o_{f3}} \tag{2.42}$$

Таким образом, метод узлового объекта примет следующий вид:

$$f(L_?, t)L_! = \begin{cases} l_2.r_1 : l_2.r_1 = \frac{o_{f2} * l_1.r_3}{o_{f1}};\ delay = \frac{l_2.r_1}{o_{f2} * o_{f3}} \\ \text{при условии: } l_1.r_1 = 600;\ l_1.r_2 = 2.5 \end{cases} \tag{2.43}$$

Рассмотренный пример показывает применение исчисления функциональных объектов для имитации функционирования технической систем, естественно, в качестве узлового объекта могут быть системы и процессы другого вида, например: организационно-деловые процессы, семантические сети понятий, нейронные сети, физические процессы и т.д. Классы возможных объектов определяются возможностями языка описания функциональных узлов, а так же видам функциональных зависимостей входов от выходов конкретной рассматриваемой системы. В соответствии с основными положениями системно-объектного подхода «Узел-Функция-Объект», можно рассматривать четыре класса функциональных зависимостей входов и выходов системы по количеству параметров:

1. Система *s* представляет собою объект, занимающий узел с одним входом и одним выходом и реализующий функцию преобразования одного переменного. Метод узлового объекта в данном случае имеет вид:

$$f(L_?,t)L_! = \{l_!.r\colon l_!.r = \varphi(l_?);\ delay = t, \tag{2.44}$$

причем $|L_?|=1$, $|L_!|=1$.

2. Система *s* представляет собою объект, занимающий узел с несколькими входами и одним выходом и реализующий функцию преобразования нескольких переменных. Метод узлового объекта в данном случае имеет вид:

$$f(L_?,t)L_! = \{l_!.r\colon l_!.r = \varphi(l_{?1},\dots,l_{?n});\ delay = t, \tag{2.45}$$

причем $|L_?|=n$, $|L_!|=1$.

3. Система *s* представляет собою объект, занимающий узел с одним входом и несколькими выходами. Как правило, функция, которую реализует объект такого вида, состоит из ряда подфункций. Метод узлового объекта в данном случае имеет вид:

$$f(L_?,t)L_! = \begin{cases} l_{!1}.r\colon l_{!1}.r = \varphi_1(l_?);\ delay = t_1; \\ \dots; \\ l_{!n}.r\colon l_{!n}.r = \varphi_n(l_?);\ delay = t_n \end{cases} \tag{2.46}$$

причем $|L_?|=1$, $|L_!|=n$.

4. Система *s* представляет собою объект, занимающий узел с несколькими входами и несколькими выходами. Метод узлового объекта в данном случае имеет вид:

$$f(L_?,t)L_! = \begin{cases} l_{!1}.r\colon l_{!1}.r = \varphi_1(l_{?1},\dots,l_{?m});\ delay = t_1; \\ \dots; \\ l_{!n}.r\colon l_{!n}.r = \varphi_n(l_{?1},\dots,l_{?m});\ delay = t_n \end{cases} \tag{2.47}$$

причем $|L_?|=m$, $|L_!|=n$.

При описании методов узловых объектов, соответствующих второму и третьему виду, как правило, необходимо проводить дополнительную декомпозицию узла, до тех пор, пока мощность множества выходящих потоковых объектов не будет равна единице. В то же время, в ряде случаем этим правилом можно пренебречь, например, в случае, если зависимость выходных показателей от входных не является сложной и поддается описанию с помощью языка описания функциональных узлов, который будет рассмотрен позднее.

3.3. Применение исчисления систем как функциональных объектов для описания их состояния, создания библиотек системных элементов и оптимизации системно-объектных моделей

3.3.1. Состояния потоковых объектов, мера системности

Рассмотрим понятие «системно-объектная модель» системы в терминах упомянутого выше исчисления и представим ее в виде:

$$M = (L, S), \tag{2.48}$$

где: M – модель системы;

L – множество потоковых объектов модели, элементы которого представляют собою объекты, которые не имеет методов и имеют лишь поля:

$$l=[r_1, r_2, \dots, r_k], \tag{2.49}$$

где:

$l \in L$;

k – количество полей потокового объекта l;

$r_1, r_2, ..., r_k$ – поля потокового объекта, представляющие собой пару «идентификатор-значение».

S – множество узловых объектов модели, элементы которого описываются выражением 2.18.

Причем, узловые объекты модели M представляют собой ключевые элементы модели, а множество потоковых объектов – определяет отношения между узловыми объектами модели. Фактически, системно-объектная модель представляет собой граф, для которого в роли вершин выступают узловые объекты, а в роли ребер – потоковые объекты.

Введем в рассматриваемое исчисление понятие «состояние потокового объекта» – представляющее набор определенных значений полей потокового объекта в момент времени t. Состояние потокового объекта l в момент времени t обозначим A_t^l и запишем в следующей виде:

$$A_t^l = [a_1, a_2, \dots, a_k], \tag{2.50}$$

где: $a_1, a_2, \dots, a_k$ значения или промежутки значений соответствующих полей потокового объекта $l.r_1, l.r_2, \dots, l.r_k$ в момент времени t. Множество состояний потокового объекта обозначим, соответственно A.

Тогда переход потокового объекта из одного состояния в другое будет осуществляться при вызове метода узлового объекта, для которого потоковый объект l является интерфейсным. Т.е. потоковый объект l системно-объектной модели M (L,S) является интерфейсным по отношению к узловому объекту s, если l∈s.L? или l∈s.L!, причем l∈L и s∈S. Проще говоря, потоковый объект является интерфейсным если он входит или выходит из узлового объекта.

Рассмотрим пример процедуры изменения состояния потокового объекта. По условию, имеем нагревательный элемент, на вход которого подается поток воды, на выходе, соответственно, поток воды нагретый. Представим данную систему с помощью модификации выражения 2.48:

$$M' = (L, S) \tag{2.51}$$

L={l[temp]} – множество потоковых объектов состоит из одного элемента l – соответствует объекту «вода», который, в свою очередь, содержит одно поле l.temp – соответствует свойству «температура».

S={s[l?, l!; f(l?)l!; o?, o!, of]} – множество узловых объектов состоит из одного элемента s – соответствует объекту «водонагреватель», структура которого следующая:

- l? – интерфейсный потоковый объект «подающий» воду в узловой объект;
- l! – интерфейсный потоковый объект «отводящий» воду из узлового объекта;
- f(l?)l! – метод узлового объекта, описывающий процесс преобразования входа в выход;
- o? – пропускная способность входа системы;
- o! – пропускная способность выхода системы;
- of – время, затрачиваемое на нагрев воды условного объема, на 30 градусов.

Графически, в соответствие с принятыми правилами, рассматриваемая модель будет иметь вид, соответствующий рисунку 2.16.

Введем в рассматриваемую модель два состояния потокового объекта:

A^l_{t1} =[temp=20] – соответствует холодной воде (начальное состояние);

A^l_{t2} =[temp=50], причем t2=t1+of – соответствует горячей воде.

Тогда переход от одного состояния к другому, формально примет следующий вид:

$$A_{t1}^{l} \xrightarrow{\text{s.f(l?)}} A_{t2}^{l} \tag{2.52}$$

Данный переход в дальнейшем будем интерпретировать следующим образом: потоковый объект l перешел из состояния A_{t1}^{l} в состояние A_{t2}^{l} вследствие вызова метода s. f(l?), при условии, что потоковый объект l является интерфейсным по отношению к узловому объекту s как на входе, так и на выходе. В представленном выше примере, соответствующий переход имитирует процесс нагрева воды.

Представим каждое отдельное состояние потокового объекта l в следующем виде:

$$A_1^l = \begin{pmatrix} a_1 \\ a_2 \\ \dots \\ a_k \end{pmatrix} \; A_2^l = \begin{pmatrix} a_1 \\ a_2 \\ \dots \\ a_k \end{pmatrix} \; A_m^l = \begin{pmatrix} a_1 \\ a_2 \\ \dots \\ a_k \end{pmatrix}, \tag{2.53}$$

где A_m^l – состояние потокового объекта l в момент времени m, а элементы матрицы – значения соответствующих свойств потокового объекта l, определяющие данное состояние, тогда состояния потокового объекта l=[r^1, r^2, … ,r^k] можно представить в виде матрицы состояний A_{kxm}, где m – число возможных состояний потокового объекта, k – количество полей потокового объекта, элемент матрицы a_{km} – значение поля потокового объекта, соответствующее определенному состоянию. Общий вид матрицы состояний потокового объекта l представлен ниже:

$$A_l = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{k1} & a_{k2} & \dots & a_{km} \end{pmatrix} \tag{2.54}$$

Таким образом, каждый столбец матрицы A соответствует одному единственному состоянию потокового объекта l, в то же время, каждому элементу множества потоковых объектов L будет соответствовать своя отдельная матрица возможных состояний потокового объекта. Для потокового объекта с одним свойством и двумя состояниями в описанном выше примере матрица состояний примет вид вектора A=[20 50].

Далее рассмотрим подробнее переход от одного состояния потокового объекта к другому, причем рассмотрим простейший вариант описания системы, когда на входе один потоковый объект и на выходе один потоковый объект, которые соответствуют одному элементу множества L. Как было отмечено выше, в соответствии с положениями системно-объектного подхода, переход осуществляется посредством вызова метода узлового объекта.

Рассмотрим абстрактный узловой объект s с соответствующим методом f(l?)l!?, входящий в множество узловых объектов, представленных в выражении 2.51, с учетом того, что l?={l_k} и l!={l_k}, где $l_k \in L$ и, соответственно l?⊂L, l!⊂L.

Область определения метода s.f(l?)l! представляет собой, во-первых, структурный компонент – набор определенных потоковых объектов на входе, которые могут быть обработаны методом узлового объекта, и во-вторых, содержательный компонент – возможные состояния потоковых объектов, из которых может быть осуществлен переход в другие состояния. Если множество интерфейсных потоковых объектов системы принимает вид (2.55), т.е. на входе более одного потокового объекта:

$$l?=\{l_1, l_2,\ldots,l_n\}, \quad (2.55)$$

где $l_1 \in L$, $l_2 \in L$, $l_n \in L$, l?⊂L, для которых заданы матрицы возможных состояний A_1, A_2 и A_n соответственно, тогда область определения метода D(f) представляет собой множество подматриц матриц состояний соответствующих интерфейсным потоковым объектам {l_1, l_2,…,l_n}. $D(f) = \{ A_1', A_2', \ldots, A_n' \}$, причем количество строк подматриц равно количеству строк соответствующих матриц состояний потокового объекта, а количество столбцов (состояний потокового объекта) подматрицы не может быть больше количества столбцов соответствующей матрицы состояний. В тоже время область значений $E(f) = \{ A_1'', A_2'', \ldots, A_n'' \}$ метода узлового объекта также можно представить в виде подматрицы матрицы состояний, которая представляет собою перечень возможных состояний потокового объекта, в которые может быть осуществлен переход при вызове соответствующего метода. Так как мы рассматриваем вариант системы, когда на входе и на выходе один потоковый объект, тогда подматрицы, соответствующие областям определения и значений метода можно представить, как показано на рисунке 2.22. Таким образом, для потокового объекта l с матрицей состояний A_l, в случае, когда он является интерфейсным входным и одновременно выходным потоковым объектом по отношению к узловому объекту s область определения метода $D(f) = A_l'$, а область значений $E(f) = A_l''$.

Рассмотрим понятие области требуемых функциональных состояний системы в соответствии с запросом надсистемы. Для любой системы область требуемых функциональных состояний системы s, в идеальном случае, то есть когда функционал системы отвечает полностью запросу надсистемы, соответствует области возможных состояний – представляющей собой все состояния из области определения A_l' и области значений A_l'', соответственно.

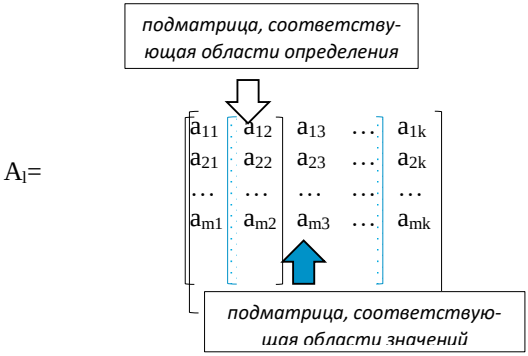


Рис. 2.22. Матрица состояний абстрактного потокового объекта.

Обозначим область возможных функциональных состояний системы s как FPS_s – представляет собой матрицу состоящую из простого объединения по столбцам матриц A'_l и A''_l, причем, если размерность матрицы A'_l равна mxn а матрицы A''_l mxk, тогда размерность матрицы FPS_s – mx(n+k). Обозначим область требуемых функциональных состояний системы s как $FRFS_s$, которая представляет собой матрицу требуемых состояний, являющейся подматрицей матрицы A_l, тогда, мера системности, отражающая степень соответствия системы запросу надсистемы, может быть представлена как коэффициент соответствия матрицы $FRFS_s$ матрице FPS_s. Соответствие определяется наличием одинаковых столбцов в рассматриваемых матрицах.

Рассмотрим алгоритм расчета меры системности. Пусть задана система s с соответствующей областью возможных состояний FPS_s и область требуемых функциональных состояний, тогда алгоритм вычисления меры системности можно представить как показано на рисунке 2.23.

На входе даны две матрицы: $FRFS_s$ – матрица требуемых функциональных состояний размерностью mxk и FPS_s – матрица возможных функциональных состояний размерностью mxn. Переменная MOS – представляет собой искомый коэффициент соответствия области возможных состояний и области требуемых состояний. Вектор A[m] – представляет собой текущее состояние из матрицы требуемых состояний, которое поочередно сравнивается с состояниями из матрицы FPS_s. Таким образом, для каждого вектора-состояния из $FRFS_s$ ищется соответствующий вектор из FPS_s, причем вектора считаются равными друг другу, если все компоненты равны. Если совпадение найдено, к переменной MOS прибавляется единица. После прохождения проверки в переменной MOS хранится количество векторов-состояний из

матрицы $FRFS_s$, которым были найдены соответствующие состояния из FPS_s, после чего переменная MOS делится на общее количество состояний из области требуемых.

Как видно из алгоритма, представленного на рисунке 2.23, он применим для простейшего случая – когда у системы один входной интерфейсный потоковый объект и один выходной интерфейсный потоковый объект, причем одинаковой структуры.

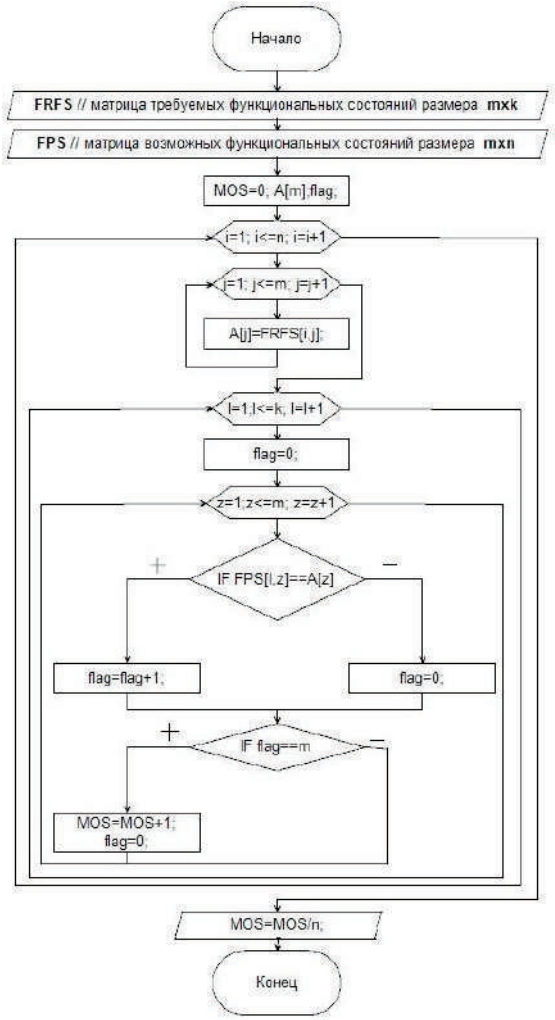


Рис. 2.23. Алгоритм расчета меры системности

Для моделирования реальных систем следует так же рассматривать более сложные ситуации, описанные ниже.

Во-первых, у системы один входной интерфейсный потоковый объект и несколько выходных интерфейсных потоковых объектов, причем структура выходных потоковых объектов отлична от структуры входного потокового объекта (рис. 2.24).

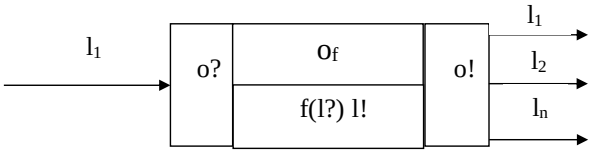


Рис. 2.24. Графический формализм системы «Один вход – несколько выходов»

Во-вторых, у системы несколько входных интерфейсных потоковых объектов и один выходной интерфейсный потоковый объект, причем структура выходного потокового объекта отлична от структуры входных потоковых объектов (рис. 2.25).

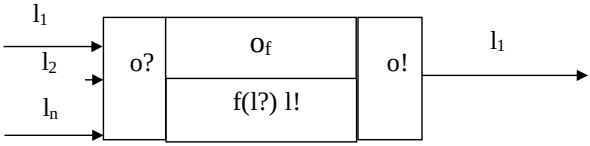


Рис. 2.25. Графический формализм системы «Несколько входов – один выход»

В-третьих, у системы несколько входных интерфейсных потоковых объектов и несколько выходных интерфейсных потоковых объектов, причем структура выходных потоковых объектов отлична от структуры выходных потоковых объектов (рис. 2.26).

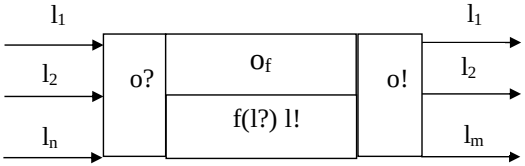


Рис. 2.26. Графический формализм системы «Несколько входов – несколько выходов»

В-четвертых, у системы один входной интерфейсный потоковый объект и один выходной интерфейсный потоковый объект, причем

структура выходного потокового объекта отлична от структуры выходного потокового объекта (рис. 2.27).

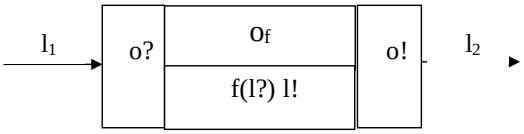


Рис.2.27. Графический формализм системы «Один вход – один выход»

Представленный алгоритм расчета меры системности в рассматриваемом исчислении открывает перспективы формализованного описания процесса адаптации системы к запросу надсистемы, т.е. процесса при котором коэффициент соответствия стремится к единице. Причем числовое выражение данного коэффициента позволяет формально обосновать применимость или неприменимость функционала системы (метода узлового объекта) к реализации запроса надсистемы.

Данные рассуждения показывают, что для определенного класса систем возможна и целесообразна оценка степени их системности и адаптированности к запросу надсистемы путем оценки соответствия текущих характеристик потоковых объектов (связей), «протекающих» через систему, характеристикам требуемым надсистемой. Это обстоятельство обусловлено тем, что требования надсистемы к системе, по сути дела, сводятся к требованию сбалансировать входы и выходы определенного функционального узла. Т.е. надсистеме требуется узел $U = L? \cap L!$, который и представляет собой функциональный запрос надсистемы на систему с определенной функцией или внешнюю детерминанту системы. Таким образом, надсистеме требуется определенное преобразование ее внутренних потоков. Сама преобразующая система (объектная характеристика O) и даже процедура преобразования входов в выходы (f) в требования не включаются. Особенно это актуально для широкого класса систем, которые можно назвать «проточными», т.е. системы с одним входным потоком и с одним выходным. При этом, по сути дела, это содержательно один и тот же поток, но объекты этого потока на входе и выходе имеют разные состояния. К ним можно отнести водонагреватели, насосы, трансформаторы, термопласт автоматы и т.д.

3.3.2. Состояния узловых объектов

Введем в исчисление систем как функциональных объектов понятие состояния узловых объектов, для этого рассмотрим системно-объектную модель вида (2.48), в которой S – множество узловых объектов системы, причем множество состоит из одного единственного элемента – абстрактный узловой объект s с методом f(l?)l!?, как представлено в выражении 2.51.

Состояние узлового объекта s в заданный момент времени t будет определяться его интерфейсными потоковыми объектами, а точнее говоря, значениями полей интерфейсных потоковых объектов в заданный момент времени, или состояниями интерфейсных потоковых объектов. Если состояния интерфейсных потоковых объектов системы рассматривать как выражение (60), где каждому отдельному состоянию потокового объекта l соответствует столбец матрицы, тогда состояние узлового объекта можно представить в виде связной совокупности текущих состояний интерфейсных потоковых объектов.

$$A_l = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{k1} & a_{k2} & \dots & a_{km} \end{pmatrix} \qquad (2.56)$$

Состояния узлового объекта будет представлять собой различные вариации столбцов матриц вида (2.56) соответствующих интерфейсным потоковым объектам. Рассмотрим пример системно-объектной модели вида 2.48, у которой:

$$L=\{l_1[r_{11}, r_{12}, r_{13}], l_2[r_{21}, r_{22}, r_{23}, r_{24}], l_3[r_{31}], l_4[r_{41}, r_{42}]\ \} \qquad (2.57)$$

$$S=\{s[l_?=\{l_3, l_1\}, l_!=\{l_2, l_4\}\ f(l_?)l_!; o_?, o_!, o_f]\} \qquad (2.58)$$

Графически, система, описанная выражениями 2.57 и 2.58, будет иметь вид, как показано на рисунке 2.28:

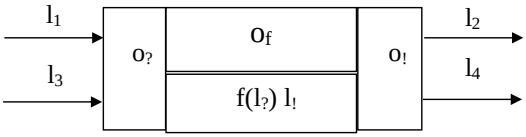


Рис. 2.28. Графический формализм системы

Допустим, что матрицы состояний описанных выше потоковых объектов имеют следующий вид:

$$A_1 = \begin{pmatrix} a_{11}^1 & \dots & a_{1m}^1 \\ a_{21}^1 & \dots & a_{2m}^1 \\ a_{31}^1 & \dots & a_{3m}^1 \end{pmatrix} A_2 = \begin{pmatrix} a_{11}^2 & \dots & a_{1k}^2 \\ a_{21}^2 & \dots & a_{2k}^2 \\ a_{31}^2 & \dots & a_{3k}^2 \\ a_{41}^2 & \dots & a_{4k}^2 \end{pmatrix} A_3 =$$

$$(a_{11}^3 \quad \dots \quad a_{1n}^3)\, A_4 = \begin{pmatrix} a_{11}^4 & \dots & a_{1b}^4 \\ a_{21}^4 & \dots & a_{2b}^4 \end{pmatrix} \leftrightarrow A_1 = (A_1^1, \dots, A_m^1) A_2 =$$

$$(A_1^2, \dots, A_k^2) A_3 = (A_1^3, \dots, A_n^3) A_4 = (A_1^4, \dots, A_b^4) \quad (2.59)$$

Тогда состояние узлового объекта s в момент времени t примет вид:

$$A_t^s = (A_1, A_2, A_3, A_4,) \quad (2.60)$$

Рассмотрим различные варианты узловых объектов в зависимости от структуры интерфейсных потоковых объектов.

Для самого простейшего случая, когда у узлового объекта на входе и на выходе один и тот же потоковый объект, алгоритм расчета меры системности узлового объекта рассмотрен авторами ранее.

Рассмотрим более сложные варианты систем в зависимости от структуры интерфейсных потоковых объектов. У системы один входной интерфейсный потоковый объект и один выходной интерфейсный потоковый объект, причем структура выходного потокового объекта отлична от структуры входного потокового объекта (см. рисунок 2.27). Для такого типа узлового объекта состояние системы можно представить в виде вектора:

$$a_s = [A^{l_1}, A^{l_2}], \quad (2.61)$$

где:

A^{l_1} – состояние входного потокового объекта l_1;

A^{l_2} – состояние исходящего потокового объекта l_2.

В общем случае для узлового объекта состояние в заданный момент времени примет вид:

$$a_s = [A^{l?_1}, \dots, A^{l?_n},\ A^{l!_1}, \dots, A^{l!_m}], \quad (2.62)$$

где:

n – количество входящих интерфейсных потоковых объектов;

m – количество исходящих интерфейсных потоковых объектов.

Для описания алгоритма расчета меры системности для узлового объекта с одним входом и одним выходом, как представлено на рисунке 2.27, введем следующие обозначения: FRFSs – (область) множество

требуемых функциональных состояний узлового объекта и FPSs – (область) множество возможных функциональных состояний. Причем элементы данных множеств имеют вид 2.62. Переменная MOS – представляет собой искомый коэффициент соответствия области возможных состояний и области требуемых функциональных состояний. Алгоритм расчета меры системности будет представлять собою поочередное сравнение элементов множества FRFSs с элементами множества FPSs.

Как видно из рисунка 2.29, элемент из множества требуемых состояний поочередно сравнивается с элементами множества возможных состояний. Если идентичное состояние найдено, тогда переменная MOS увеличивается на единицу и далее осуществляется переход к новому требуемому состоянию, так как далее сравнивать текущее требуемое состояние не имеет смысла, оно уже найдено. По истечении работы внешнего цикла, в переменной MOS будет содержаться количество найденных требуемых функциональных состояний из множества возможных, после чего поделив данное количество на общее количество требуемых состояний, получим значение коэффициента системности в промежутке от нуля до единицы, причем, чем ближе коэффициент к единице, тем система более соответствует запросу надсистемы. Данный алгоритм будет работать для всех видов узловых объектов, главная проблема будет состоять в адекватном формировании множества требуемых функциональных состояний узлового объекта.

Рассмотрим пример расчета меры системности для системно-объектной модели экструдера – устройства, на вход которого подается гранулированный пластик, на выходе – полиэтиленовая пленка определенной длинны и ширины. Для этого подходит модель системы в виде выражения 2.48.

Введем в множество потоковых объектов системы два потоковых объекта с условными характеристиками:

$$L=\{gr[pl, ve, ra]; po[sh, dl]\}, \quad (2.63)$$

где:

gr – потоковый объект «гранулированный пластик»;
pl – плотность гранулированного пластика;
ve – вес гранулированного пластика;
ra – средний диаметр гранул;
po – потоковый объект «полиэтиленовая пленка»;
sh – ширина пленки;
dl – длинна пленки.

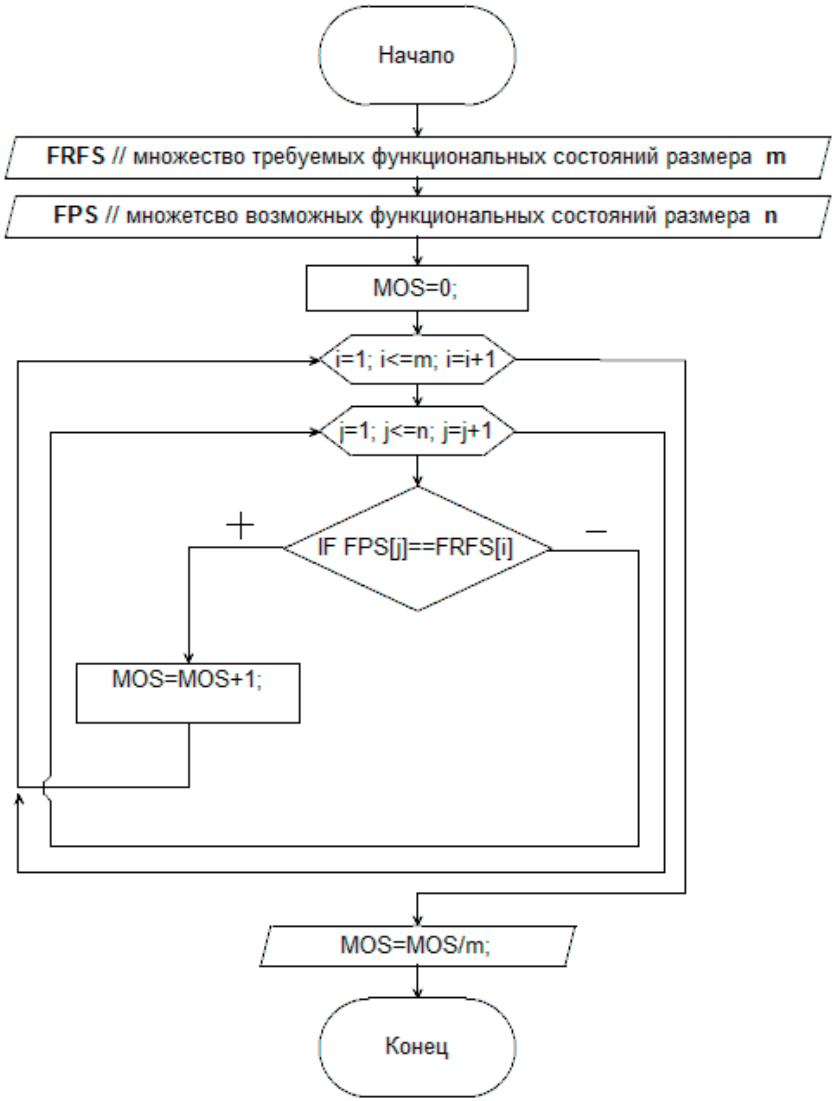


Рис. 2.29. Алгоритм расчета меры системности узлового объекта

Введем области возможных условных состояний для потоковых объектов «гранулированный пластик» и «полиэтиленовая пленка»:

$$A_{gr} = \begin{pmatrix} 2{,}3 & 2{,}3 & 2{,}3 & 2{,}3 \\ 350 & 342 & 267 & 341 \\ 4{,}2 & 4{,}1 & 4{,}21 & 4{,}32 \end{pmatrix} \quad (2.64)$$

$$A_{po} = \begin{pmatrix} 190 & 190 & 190 & 190 & 190 & 190 \\ 23 & 24 & 25 & 26 & 27 & 28 \end{pmatrix} \quad (2.65)$$

Множество узловых объектов будет состоять из одного элемента ek – соответствует объекту «экструдер», структура которого представлена ниже:

$$S=\{ek[gr?, po!; f(gr?)po!; o?, o!, o_f]\}, \quad (2.66)$$

где:

gr? – интерфейсный потоковый объект «гранулированный пластик», подающийся на вход узлового объекта;

po! – интерфейсный потоковый объект «полиэтиленовая пленка», являющийся результатом функционирования экструдера;

f(gr?)po! – метод узлового объекта, описывающий процесс преобразования входа в выход;

o? – пропускная способность входа системы;

o! – пропускная способность выхода системы;

of – время, затрачиваемое на производство одного погонного метра полиэтиленовой пленки.

Графически, в соответствие с принятыми правилами, рассматриваемая модель будет иметь следующий вид (рис. 2.30).

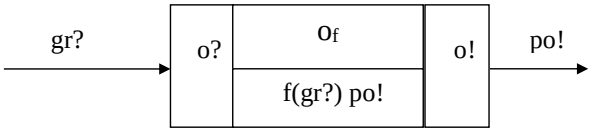


Рис. 2.30. Графический формализм системы «экструдер»

Введем в рассматриваемую модель область возможных условных состояний узлового объекта:

$$A_{ek} = \left(\begin{bmatrix} 2{,}3 & 190 \\ 342 & 24 \\ 4{,}1 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 190 \\ 267 & 28 \\ 4{,}21 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 190 \\ 341 & 23 \\ 4{,}32 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 190 \\ 350 & 27 \\ 4{,}2 & \end{bmatrix} \right) \quad (2.67)$$

Так как компоненты вектора возможных состояний узлового объекта состоят из возможных состояний соответствующих потоковых объектов, нетрудно рассчитать максимальное количество состояний узлового объекта. Если, например, количество возможных состояний первого потокового объекта равно 4, а количество возможных состояний

второго потокового объекта равно 6, тогда количество комбинаций (состояний узлового объекта) будет равно 4*6= 24. Таким образом, для общего случая, максимальное количество возможных состояний узлового объекта можно рассчитать по формуле произведения:

$$K = \prod_{i=1}^{n} k_i \,, \tag{2.68}$$

где:

K – максимальное количество возможных состояний узлового объекта;

n – количество интерфейсных потоковых объектов;

ki – количество возможных состояний i-го потокового объекта.

Для расчета меры системности представленного выше узлового объекта сформулируем область требуемых состояний системы:

$$\mathrm{FRFS}_{ek} = \left(\begin{bmatrix} 2{,}3 & 190 \\ 347 & 24 \\ 4{,}1 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 190 \\ 267 & 28 \\ 4{,}21 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 190 \\ 341 & 24 \\ 4{,}32 & \end{bmatrix}; \begin{bmatrix} 2{,}3 & 192 \\ 356 & 27 \\ 4{,}2 & \end{bmatrix} \right) \tag{2.69}$$

Область возможных состояний системы равна соответственно FPSs = A_{ek}. Тогда в соответствии с представленным алгоритмом, коэффициент системности будет равен ¼. Это говорит о том, что система на 25% соответствует запросу надсистемы.

3.3.3. Библиотеки узловых объектов

Рассмотрим системно-объектную модель вида 2.48, где L – множество потоковых объектов модели, элементы которого представляют собою объект, не имеющий методов, но имеющий поля, представленного ниже вида:

$$l=[r_1, r_2, \ldots, r_k], \tag{2.70}$$

где:

$l \in L$;

k – количество полей потокового объекта l;

r_1, r_2, ..., r_k – поля потокового объекта, представляющие собой пару «идентификатор-значение».

При этом S – множество узловых объектов модели, элементы которого описываются формулой 2.8.

Библиотеку узловых объектов вида 2.48 предлагается создавать с условием, что |L|=0. То есть, библиотека будет иметь лишь узловые объекты, и не будет содержать в своем составе потоковые объекты. Тогда библиотеку системно-объектной модели можно рассматривать как множество узловых объектов следующего вида:

$$S^{`}=[s_1,s_2,\ldots,s_n], \quad (2.71)$$

где n – количество узловых объектов, хранящихся в библиотеке.

Рассмотрим подробнее элемент библиотеки, а точнее его формальное описание. Как было отмечены выше, элемент библиотеки будет представлять собой отдельную смоделированную систему, описываемую выражением 2.8.

Каждый элемент библиотеки представляет собой УФО-элемент с соответствующими интерфейсными связями, по которым и проводится анализ на соответствие текущего элемента заданным характеристикам. Соответственно, библиотека элементов будет представлять собою набор узловых объектов, не связанных между собой.

Рассмотрим абстрактную библиотеку S_M состоящую из следующих элементов, как показано на рисунке 2.31:

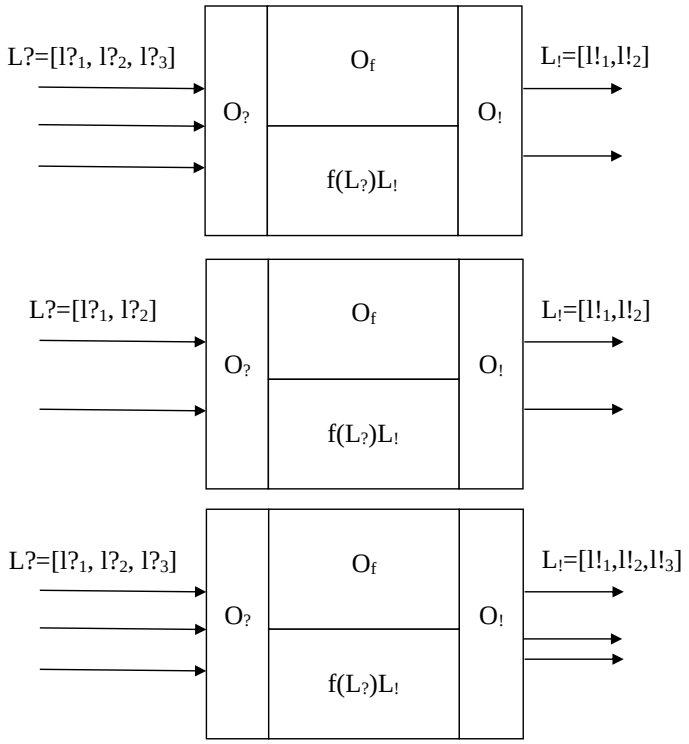


Рис. 2.31. Графический формализм абстрактной библиотеки узловых объектов

Очевидно, что для пополнения библиотеки и ее дальнейшего использования, необходимо рассмотреть как минимум две операций на системно-объектных моделях: добавление узлового объекта в библиотеку – экспорт элемента и импорт узлового объекта из библиотеки. Для описания данных операций, исчерпывающим набором параметров являются интерфейсы импортируемых и экспортируемых узловых объектов. Внутренняя организация таких объектов (функция и объект системы) для рассматриваемых операций не имеет значения. Из рисунка 2.31 видно, что для каждого отдельного узлового объекта интерфейсом является его идентификатор, то есть имя и наборы входных и выходных потоковых объектов с учетом их структур и типов полей.

Рассмотрим в качестве примера узловой объект со структурой интерфейсных связей, как показано у первого объекта на рисунке 2.31.

Формальный вид данного узлового объекта представлен ниже:

$$s_n=[\ L?=\{l?_1, l?_2, l?_3\}, L!=\{l!_1,l!_2\}; f(L?)L!; O?, O!, Of] \qquad (2.72)$$

Интерфейс узлового объекта соответствует структурной характеристике системы U из выражения (2.72), что соответствует основным положениям методологии «Узел-Функция-Объект». Однако, помимо структурной составляющей интерфейса узлового объекта, в случае импорта и экспорта элементов, важную роль имеет типовая структура интерфейсных потоковых объектов, т.е. необходимо учитывать типы данных полей потоковых объектов, составляющих интерфейсную часть узлового объекта. Таким образом, если в узловом объекте интерфейсные потоковые объекты имеют следующую структуру:

$l?_1=(r_1, r_2)$

$l?_2=(r_1)$

$l?_3=(r_1)$

$l!_1=(r_1, r_2, r_3)$

$l!_2=(r_1, r_2)$,

тогда интерфейс узлового объекта запишем в следующем виде:

$$U_s = \begin{cases} L? = \begin{cases} l?_1 = (r_1, r_2) \\ l?_2 = (r_1) \\ l?_3 = (r_1) \end{cases} \\ L! = \begin{cases} l!_1 = (r_1, r_2, r_3) \\ l!_2 = (r_1, r_2) \end{cases} \end{cases} \qquad (2.73)$$

В общем виде выражение (2.73) можно записать следующим образом:

$$U_s = \begin{cases} \text{L?} = \begin{cases} \text{l?}_1 = (\text{r}_1, \dots, \text{r}_{i_1}) \\ \dots \\ \text{l?}_n = (\text{r}_1, \dots, \text{r}_{i_n}) \end{cases} \\ \text{L!} = \begin{cases} \text{l!}_1 = (\text{r}_1, \dots, \text{r}_{j_1}) \\ \dots \\ \text{l!}_m = (\text{r}_1, \dots, \text{r}_{j_m}) \end{cases} \end{cases} \quad (2.74)$$

Рассмотрим подробнее операцию импорта узлового объекта s из системно-объектной модели M в библиотеку S'. Пусть дана иерархия потоковых объектов модели, содержащая три вещественных потоковых объекта со своими полями следующего вида:

$$L_M = [l_1^v = \{r_1^1, r_2^1\}, l_2^v = \{r_1^2, r_2^2, r_3^2\}, l_3^v = \{r_1^3\}] \quad (2.75)$$

Также пусть дана соответствующая системно-объектная модель вида (2.48), причем множество узловых объектов имеет следующий вид:

$$\begin{aligned} &S=[s_1=(L?=\varnothing,\ L!=\{l!_1\};\ f(L?)L!;\ O?,\ O!,\ Of),\\ &s_2=(L?=\{l?_1,\ l?_3\},\ L!=\{l!_2\};\ f(L?)L!;\ O?,\ O!,\ Of),\\ &s_3=(L?=\{l?_2\},\ L!=\{l!_3\};\ f(L?)L!;\ O?,\ O!,\ Of)], \end{aligned} \quad (2.76)$$

а множество потоковых объектов имеет следующий вид:

$$L=[l_1=\{s_1,s_2\},\ l_2=\{s_2,s_3\},\ l_3=\{s_3,s_2\}] \quad (2.77)$$

Графически пример описанный выражениями (2.75), (2.76) и (2.77) имеет следующий вид (рис. 2.32):

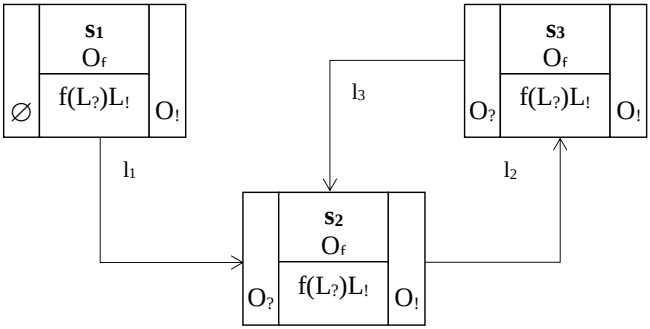


Рис. 2.32. Пример системно-объектной модели

Тогда, для импорта узлового объекта s_2 системно-объектной модели M в библиотеку $L_м$ используем оператор в следующей виде:

L_M^* = import(M,s_2,L_M)→(L^*=∅; S^*=[s_1=(L?=∅, L!={l!$_1$}; f(L?)L!; O?, O!, Of), s_3=(L?={l?$_2$}, L!={l!$_3$}; f(L?)L!; O?, O!, Of)]; L_M^*=[s_2=(L?={l?$_1$, l?$_3$}, L!={l!$_2$}; f(L?)L!; O?, O!, Of)].

Результатом данной операции является библиотека L_M^*, пополненная узловым объектом s_2 и системно-объектная модель $M^*(\varnothing, S^*)$, представленная на рисунке 2.33.

Как было отмечено выше, для работы с узловым объектом, помещаемым в библиотеку, необходимо также учитывать его интерфейс. Для рассматриваемого примера, интерфейс объекта s_2 примет следующий вид:

$$U_{s_2} = \begin{cases} \text{L?} = \begin{cases} l?_1 = (r_1^1, r_2^1) \\ l?_3 = (r_1^3) \end{cases} \\ \text{L!} = \{l!_2 = (r_1^2, r_2^2, r_3^2)\} \end{cases} \quad (2.78)$$

Рис. 2.33. Результат импорта узлового объекта в библиотеку

Следует отметить, что описанная операция импорта узлового объекта в примере предполагает извлечение вышеупомянутого элемента из модели в библиотеку. Вместе с тем, в процессе моделирования пользователь может сохранить узловой объект в библиотеку путем копирования, то есть без удаления первого из исходной модели и освобождения соответствующих потоковых объектов. В таком случае исходная модель, из которой экспортируется элемент в библиотеку, остается в неизменном виде, как показано на рисунке 2.32.

Далее рассмотрим операцию экспорта узлового объекта s_2 из библиотеки L_M^* в системно-объектную модель $M^*(\varnothing, S^*)$. Оператор экспорта имеет следующий вид:

M^* = export(M,s_2,L_M^*)→(L^*=∅; S^*=[s_1=(L?=∅, L!={l!$_1$}; f(L?)L!; O?, O!, Of),s_2=(L?={l?$_1$, l?$_3$}, L!={l!$_2$}; f(L?)L!; O?, O!, Of),s_3=(L?={l?$_2$}, L!={l!$_3$}; f(L?)L!; O?, O!, Of)]; L_M^*=[s_2=(L?={l?$_1$, l?$_3$}, L!={l!$_2$}; f(L?)L!;O?, O!, Of)].

Как видно из описания операции экспорта, соответствующий узловой объект был добавлен в модель, которая принимает исходный

вид. Наряду с этим, для подбора элементов из библиотеки имеется возможность анализировать специальный количественный показатель «мера системности», который был описан ранее.

3.3.4. Оптимизация системно-объектной модели

Рассмотрим формальные основы оптимизации системно-объектных моделей систем, представляемых в терминах рассматриваемого исчисления. Здесь в качестве критериев оптимальности, предлагается использовать, в первую очередь, некоторые общесистемные принципы и закономерности. Рассмотрим подробнее принцип коммуникативности, согласно которому любая система должна быть связана множеством коммуникаций с окружающей средой (другими системами), иначе существование системы не имеет никакого смысла. Если рассматривать системно-объектную модель вида (2.48), то в терминах исчисления систем, данный принцип можно формально представить в следующем виде:

$$\forall s \in S: \nexists s.U = \emptyset \tag{2.79}$$

Так как, в терминах исчисления систем, связи узлового объекта с внешней средой представлены в форме его интерфейсных потоковых объектов U=($L_?$, $L_!$), тогда соответствие моделируемой системы принципу коммуникативности определяется отсутствием в модели узловых объектов, для которых множество интерфейсных потоковых объектов является пустым. Однако, может быть ситуация, когда, например, у узлового объекта имеются входные потоковые объекты и не имеется выходных потоковых объектов и наоборот, такая ситуация также противоречит принципу коммуникативности, поэтому данный принцип можно уточнить в следующей форме:

$$\forall s \in S: \nexists s.L_? = \emptyset \;||s.L_! = \emptyset \tag{2.80}$$

Таким образом, процесс оптимизации модели по принципу коммуникативности будет заключаться в связывании имеющихся узловых объектов с другими.

Рассмотрим оптимизацию модели абстрактной системы. Пусть дана модель вида (2.48), причем:

$$L=\{l_1[], l_2[], l_3[], l_4[] \} \tag{2.81}$$

$$\begin{aligned} &S=\{s_1[l_?=\emptyset, l_!= \emptyset; f(l_?)l_!; o], \\ &s_2[l_?=\{l_2\}, l_!=\{l_1\}; f(l_?)l_!; o], \\ &s_3[l_?=\{l_1\}, l_!=\{l_2\}; f(l_?)l_!; o]\} \end{aligned} \tag{2.82}$$

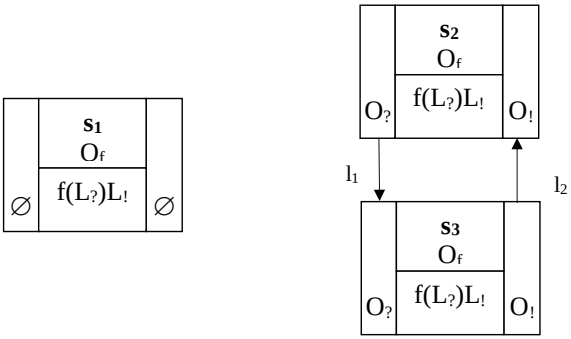


Рис. 2.34. Модель до оптимизации по принципу коммуникативности

Как видно из рисунка 2.34, рассматриваемая модель не отвечает принципу коммуникативности, так как содержатся узловые объекты с пустыми интерфейсами. Применим к рассматриваемой модели операторы присоединения в следующем формате:

$$s_1 \xrightarrow{l_3} \rightarrow s_2;\ s_3 \xrightarrow{l_4} \rightarrow s_1 \qquad (2.83)$$

После применения операторов присоединения множество потоковых объектов остается неизменным, а множество узловых объектов примет вид (2.84), модель вид см. рисунок 2.35:

$$S=\{s_1[l_?=l_4,\ l_!=l_3;f(l_?)l_!;o],\ s_2[l_?=\{l_2, l_3\},\ l_!=\{l_1\};f(l_?)l_!;o],\ s_3[l_?=\{l_1\},\ l_!=\{l_2, l_4\};f(l_?)l_!;o]\} \qquad (2.84)$$

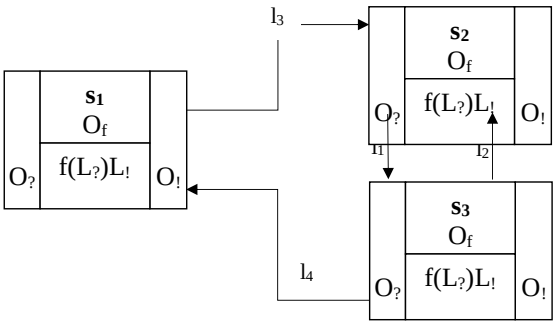


Рис. 2.35. Модель после оптимизации по принципу коммуникативности

Далее рассмотрим процесс оптимизации модели по принципу обратной связи, согласно которому устойчивость в сложных динамических

системах достигается за счет замыкания петель обратных связей. Формально, данный принцип можно представить в виде следующего выражения:

$$\exists s_i \in S,\ s_j \in S: l_m \in s_i.L_?, l_m \in s_j.L_!, l_n \in s_j.L_?, l_n \in s_i.L_! \qquad (2.85)$$

Выражение (2.83) гарантирует наличие двух узловых объектов, соединенных обратными связями, как показано на рисунке 2.36.

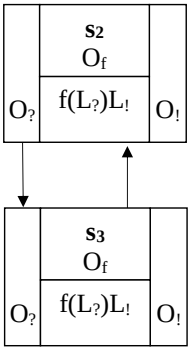


Рис. 2.36. Пример замкнутых связей системы

Соответственно, оптимизация модели по принципу обратной связи заключается в присоединения соответствующими видами связей узловых объектов, использую операцию присоединения, аналогично процессу оптимизации модели по принципу коммуникативности.

4. Приложения теории систем, основанной на системно-объектном подходе

4.1. Создание онтологии на основе системно-объектной уфо-модели

В конкурентной борьбе современных предприятий все большую роль приобретает организационное знание. Соответственно, исследователи и практические специалисты прикладывают значительные усилия к развитию такой области, как управление знаниями. В частности, разрабатываются и совершенствуются методы обработки, хранения и совместного использования накопленных знаний. Эффективные решения подобных задач предоставляет онтологический инжиниринг – теория и технологии разработки онтологий. Под онтологией в данном

случае имеется в виду подробная спецификация структуры определенной проблемной области.

Онтологии предметных областей широко применяются при построении прикладных информационных систем, в том числе для поддержки принятия решений в сложных и слабоструктурированных областях. Несмотря на наличие общедоступных хранилищ онтологий и соответствующих поисковых систем, их практическое применение сопряжено с рядом трудностей. В первую очередь, это избыточность опубликованных онтологий по отношению к конкретной решаемой задаче. К тому же, существующие онтологии с трудом поддаются оценке на предмет актуальности, достоверности и полноты. Соответственно, разработчики информационных систем и эксперты зачастую прибегают к созданию собственных онтологий предметных областей.

В настоящее время разработан целый ряд подходов и методологий построения онтологий: «скелетная» методология Ушолда и Кинга, методология Грюнингера и Фокса, METHONTOLOGY, On-To-Knowledge (OTKM), SENSUS, KACTUS, DILIGENT, группа методов построения онтологий на основе лингвистического анализа текстов. Однако большинство из них предъявляет высокие требования к разработчикам онтологии, слабо детализированы, плохо поддаются формализации и автоматизации. Альтернативой является интеграция методов онтологического инжиниринга и средств системно-объектного подхода «Узел-Функция-Объект» (УФО-подхода), в частности, применение уже существующих или разрабатываемых компьютерных визуальных моделей предметной области (УФО-моделей) для отбора понятий и построения прототипа онтологии [Кондратенко, 2016, 2].

Такой подход к построению онтологий позволяет уменьшить долю непосредственного («ручного») участия экспертов в процессе создания онтологии, формализовать (благодаря широким возможностям формализации УФО-подхода) и автоматизировать наиболее трудоемкие и длительные операций. Кроме того, УФО-модель, которая в данном случае становится основой для построения онтологии, достаточно полно отражает предметную область и при этом не содержит избыточных, не связанных с рассматриваемой предметной областью сущностей. Указанная особенность, а также простота актуализации УФО-модели, и, соответственно, построенной на ее основе онтологии, позволяют говорить об эффективности такого подхода.

Рассмотрим суть метода построения онтологий на основе системно-объектного подхода. Знания о предметной области, содержащиеся в УФО-модели, могут быть представлены в виде набора фактов (аксиом) об отдельных концептах. Такие аксиомы можно систематизировать с точки зрения уровня рассмотрения фактов [Слободюк, 2014, 1], приняв следующие обозначения: k? – количество входящих связей узла, k! – количество исходящих связей, t? – количество различных типов входящих связей, t! – количество различных типов исходящих связей, q – количество признаков (атрибутов) объекта, h – количество связей между УФО-элементами УФО^1 и УФО^2:

Группа I. На уровне УФО-элемента

1. Узел балансируется функцией.
2. Функция реализуется с помощью объекта.
3. Узел занят объектом.

Группа I(inv). На уровне УФО-элемента

1. Функция балансирует узел.
2. Объект реализует функцию.
3. Объект занимает узел.

Группа II. На уровне Узла

1. (Для каждого m=1..t?) Узел имеет входящую связь (порт) класса β_m
2. (Для каждого m=1..t!) Узел имеет исходящую связь (порт) класса β_m.
3. (Для каждого m=1..k?) Узел имеет входящую связь L_m?
4. (Для каждого m=1..k!) Узел имеет исходящую связь L_m!
5. (Для каждого m=1..k?) Связь L_m? принадлежит классу β_j, причем $j \in (1..t?)$
6. (Для каждого m=1..k!) Связь L_m! принадлежит классу β_j, причем $j \in (1..t!)$

Группа III. На уровне Функции

1. (Для каждого m=1..k?) Функция преобразует вход L_m?
2. (Для каждого m=1..k!) Функция выдает выход L_m!

Группа IV. На уровне Объекта

1. (Для каждого m=1..q) Объект обладает признаком α_m.

Группа V. На уровне связей между двумя УФО-элементами

1. (Для каждого m=1..h) УФО^1 передает L УФО^2

Группа V(inv). На уровне связей между двумя УФО-элементами

1. (Для каждого m=1..h) УФО^2 получает L от УФО^1

Группа VI. На уровне декомпозиции УФО-элемента

1. УФО^{1}имеет в качестве части УФО^n

Группа VII. На уровне агрегации УФО-элементов

1. УФО^{n}является частью УФО^1.

Для включения подобных фактов в создаваемую онтологию требуется их формальное описание. Базовым формализованным языком представления онтологий является RDF (Resource Description Framework), предполагающий представление сведений о предметной области в виде триплетов «Субъект–предикат–объект». Проведенные исследования [Слободюк, 2014, 1; Кондратенко, 2015; 2016, 1] показали, что факты о предметной области, содержащиеся в УФО-модели, возможно описать средствами языка RDF. Для этого, с одной стороны, необходимо использовать адаптированные средства алгебраического описания УФО-моделей, а с другой стороны, несколько модифицировать формализованные средства представления создаваемой онтологии.

Так, для представления извлекаемых из УФО-модели фактов о предметной области на языке описания онтологий, в частности, RDF, требуется набор предикатов – сущностей, которые обозначают отношения между субъектами и объектами RDF-триплетов [Кондратенко, 2016, 2]. В онтологическом инжиниринге в целях повышения универсальности разрабатываемой онтологии принято пользоваться предикатами из общепринятых RDF-словарей, но для описания УФО-моделей их недостаточно. Поэтому вводится дополнительный RDF-словарь, предикаты которого позволяют представить все специфические сведения, извлеченные из УФО-модели согласно классификации фактов. В таблице 2.9 приведен небольшой фрагмент указанного словаря. Полный перечень введенных предикатов можно найти в работе [Кондратенко, 2016, 3].

Чтобы упорядочить введенные предикаты, применяют расширение языка RDF – RDF Schema (сокращенно RDFS). Его средствами создаются новые классы пространства имен ufo для каждого следующего понятия УФО-модели: УФО-элемент (ufo:UFO), Узел (ufo:UFONode), Функция (ufo:UFOFunction), Объект (ufo:UFOObject), Класс связи (ufo:LinkClass), Связь (ufo:Link), Порт (ufo:Port). Предикаты из введенного ранее словаря становятся свойствами данных классов. Применение подобных конструкций предоставляет возможность ввода собственных предикатов без нарушения синтаксической и семантической непротиворечивости кода.

Таблица 2.9

Фрагмент специализированного словаря для представления УФО-моделей на языке RDF

Группа фактов	Наименование отношения	Предикат	Полное представление предиката в нотации N-Triples языка RDF
I	Балансируется	isBalancedBy	<http://www.ufo-toolkit.ru/#isBalancedBy>
	Реализуется с помощью	isRealizedBy	<http://www.ufo-toolkit.ru/#isRealizedBy>
	Занят	isRepresentedBy	<http://www.ufo-toolkit.ru/#isRepresentedBy>
II	Имеет входящий порт	hasInputPort	<http://www.ufo-toolkit.ru/#hasInputPort>
	Имеет исходящую связь	hasInputRelation	<http://www.ufo-toolkit.ru/#hasInputRelation>
III	Преобразует вход	translateInput	<http://www.ufo-toolkit.ru/#translateInput>
	Выдает выход	giveOutput	<http://www.ufo-toolkit.ru/#giveOutput>
IV	Обладает признаком	hasAttribute	<http://www.ufo-toolkit.ru/#hasAttribute>
V и V (inv)	Эквивалентна	equivalent	<http://www.ufo-toolkit.ru/#equivalent>
	Имеет идентификатор	hasID	<http://www.ufo-toolkit.ru/#hasID>
	Предполагает прямое отношение	hasDirectRelation	<http://www.ufo-toolkit.ru/#hasDirectRelation>
	Предполагает обратное отношение	hasReverseRelation	<http://www.ufo-toolkit.ru/#hasReverseRelation>

Таким образом, метод создания онтологии предметной области на основе УФО-модели предметной области можно представить с помощью следующей схемы (рисунок 2.37).

Рис. 2.37. Основные составляющие метода построения онтологий на основе УФО-моделей предметных областей

Особое внимание при построении онтологии по данному методу уделяется валидации результата. Здесь под валидацией имеется в виду оценка непротиворечивости онтологии и соответствие ее формального представления регламентированным правилам соответствующего языка (формата) [Кондратенко, 2016, 3]. Для описания критериев (правил) корректности разрабатываемой онтологии воспользуемся средствами языка SWRL–Semantic Web Rule Language [SWRL]. Для онтологии, построенной на основе УФО-модели предметной области, должны выполняться следующие правила:

- Для каждой связи должен быть указан класс

$$hasClass(?L, null) \Rightarrow hasNoLinkCLass(?L);$$

- Для каждой связи должен быть указан только один класс

$$hasClass(?L, ?LC1) \wedge hasClass(?L, ?LC2) \Rightarrow hasManyLinkClasses(?L);$$

- В онтологии не должно быть нескольких сущностей с одинаковым значением идентификатора (ID)

$$rdfs:hasID(?x, ?id) \wedge rdfs:hasID(?y, ?id) \Rightarrow hasSameID(?x, ?y);$$

- Одна и та же функция не должна балансировать более одного узла

$$balances(?f, ?u1) \wedge balances(?f, ?u2) \Rightarrow balancesManyNodes(?f);$$

- Один и тот же объект не должен занимать более одного узла

$$represents(?o, ?u1) \wedge represents(?o, ?u2) \Rightarrow representsManyNodes(?o);$$

- В онтологии не должно быть нескольких объектов с одинаковым именем (значением свойства Name)

$$Name(?o1, ?N) \wedge Name(?o2, ?N) \Rightarrow sameNameObjects(?o1, ?o2);$$

- В онтологии не должно быть нескольких функций с одинаковым именем (значением свойства Name)

$$Name(?f1, ?N) \wedge Name(?f2, ?N) \Rightarrow sameNameFunctions(?f1, ?f2);$$

- В онтологии не должно быть нескольких узлов с одинаковой нотацией (значением свойства Notation)

$$Notation(?u1, ?N) \wedge Notation(?u2, ?N) \Rightarrow sameNotationNodes(?u1, ?u2).$$

Проверка соответствия данным критериям позволяет выявить и исправить ряд логических ошибок в построенной онтологии.

Метод построения онтологий на основе системно-объектных моделей предметной области позволяет применить средства обработки онтологий к лежащим в их основе УФО-моделям. Например, известен метод решения задач логического вывода знаний на УФО-моделях [Кондратенко, 2016, 3], основными этапами которого являются:

- извлечение знаний о предметной области из компьютерной графоаналитической системно-объектной модели (УФО-модели);
- представление извлеченных знаний на формальном языке записи онтологий, то есть в виде, пригодном для машинной обработки;
- применение инструментов логического вывода (языка запросов или решателя) к полученному формальному представлению онтологии, построенной на основе УФО-модели.

Более подробно суть данного метода отражает алгоритм логического вывода на УФО-моделях, блок-схема которого приведена на рисунке 2.38.

При решении задач поиска требуемых знаний в онтологии используется язык запросов SPARQL. Для вывода новых знаний на базе уже включенных в онтологию сведений, а также добавления результатов такого вывода непосредственно в формальное описание онтологии, используется ранее упомянутый язык SWRL. Это более сложный класс задач, примеры решения которых приведены ниже (формальная запись представлена в виде SWRL-правил):

- включение в онтологию фактов вида «Функция преобразует вход…»

$$balances(?f, ?u) \wedge hasInput\,\mathrm{Re}\,lation(?u, ?Li) \Rightarrow translateInput(?f, ?Li),$$

$$isBalancedBy(?u, ?f) \wedge hasInput\,\mathrm{Re}\,lation(?u, ?Li) \Rightarrow translateInput(?f, ?Li);$$

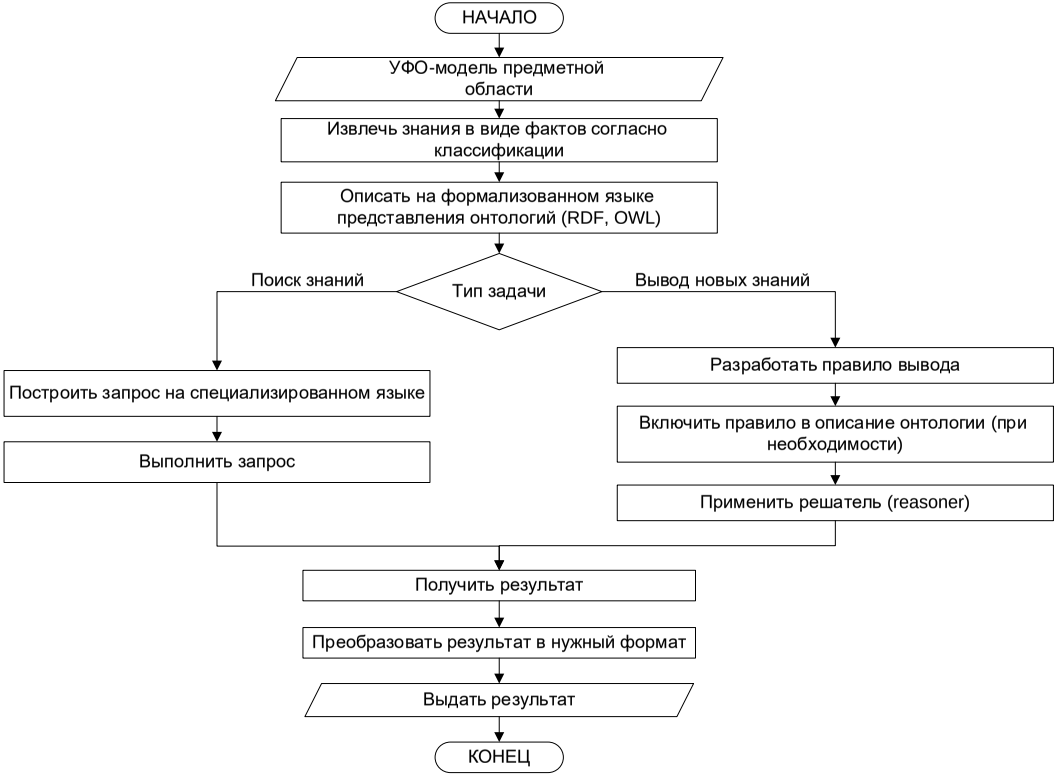


Рис. 2.38. Блок-схема алгоритма логического вывода на УФО-моделях с помощью средств логического вывода на онтологиях

- включение в онтологию фактов вида «Функция выдает выход…»

$$balances(?f, ?u) \wedge hasOutput\,\mathrm{Re}lation(?u, ?Lj) \Rightarrow giveOutput(?f, ?Lj),$$

$$isBalancedBy(?u, ?f) \wedge hasOutput\,\mathrm{Re}lation(?u, ?Lj) \Rightarrow giveOutput(?f, ?Lj);$$

- включение в онтологию фактов вида «Объект занимает узел»

$$realizes(?o, ?f) \wedge balances(?f, ?u) \Rightarrow represents(?o, ?u),$$

$$realizes(?o, ?f) \wedge isBalanceddBy(?u, ?f) \Rightarrow represents(?o, ?u),$$

$$is\,\mathrm{Re}alizedBy(?f, ?o) \wedge balances(?f, ?u) \Rightarrow represents(?o, ?u),$$

$$is\,\mathrm{Re}alizedBy(?f, ?o) \wedge isBalancedBy(?u, ?f) \Rightarrow represents(?o, ?u);$$

- учет нескольких уровней декомпозиции (УФО-элемент является частью другого элемента на несколько уровней выше)

$$isPartOf(?u_i^{n-2}, ?u_j^{n-1}) \wedge isPartOf(?u_j^{n-1}, ?u^n) \Rightarrow isPartOf(?u_i^{n-2}, ?u^n);$$

- определение узлов, имеющих сходные порты с заданным

$$hasInputPort(?u1, ?L_i) \wedge hasInputPort(?u2, ?L_i) \Rightarrow hasSamePort(?u1, ?u2),$$

$$hasOutputPort(?u1, ?L_i) \wedge hasOutputPort(?u2, ?L_i) \Rightarrow hasSamePort(?u1, ?u2),$$

$$hasInputPort(?u1, ?L_i) \wedge hasOutputPort(?u2, ?L_i) \Rightarrow hasSamePort(?u1, ?u2),$$

$$hasOutputPort(?u1, ?L_i) \wedge hasInputPort(?u2, ?L_i) \Rightarrow hasSamePort(?u1, ?u2).$$

Таким образом, интеграция средств УФО-подхода и онтологического инжиниринга способствует упрощению и повышению эффективности разработки онтологий предметных областей, а именно:

- позволяет снизить долю участия экспертов в процессе отбора концептов;
- обеспечивает формализацию процесса создания онтологии и предполагает автоматизацию отдельных его этапов;
- открывает возможности использования результатов ранее проведенных исследований предметной области;
- облегчает процесс актуализации онтологии, поскольку предполагает формализованную трансляцию изменений из базисной УФО-модели предметной области.

В дальнейшем средствами системно-объектного подхода могут быть решены и другие проблемы онтологического инжиниринга, в частности, задачи сравнения и интеграции (слияния) онтологий. Об этом свидетельствует возможность «обратного» преобразования описания онтологии на специализированном языке в формальное описание УФО-модели, что может стать основой нового метода интеграции онтологий.

4.2. Моделирование административных процедур

4.2.1. Системно-объектный подход к моделированию административных процедур

С развитием информационно-коммуникационных технологий во многих странах мира сформулирована и решается задача оказания государственных и муниципальных услуг населению в электронном виде. В Российской Федерации эта задача решается в рамках государственной программы «Электронная Россия».

Обзор публикаций по разработкам в рамках упомянутой программы показывает, что специалисты, работающие над ее реализацией, уделяют большое внимание задаче компьютерного моделирования регламентов административных процедур (АП), в соответствии с которыми и оказываются государственные и муниципальные услуги населению. Ведущую роль в этом вопросе занимает перспективное научное направление по созданию технологий, повышающих «прозрачность» и управляемость организационно-деловых и производственно-технологических процессов (бизнес-процессов) посредством разработки и использования типовых формализованных электронных моделей, обеспечивающих анализ и реинжиниринг этих процессов. Практически это означает, что методы и средства функционального бизнес-моделирования начинают реально применяться для анализа и рациональной организации не только бизнес-процессов, но и АП, окруженных ранее мифическим ореолом святости и неприкосновенности.

Компьютерные модели АП должны учитывать множество структурных, динамических и субстанциальных характеристик, присущих АП. Кроме того, они должны:

- использовать язык описания, не требующий специального обучения;
- использовать формализованные правила контроля содержания и согласованности (непротиворечивости) описания;
- использовать для интерфейсов пользователей программные средства общего назначения (например, офисные продукты).

Следовательно, компьютерные модели АП должны быть системными графоаналитическими моделями. Рассмотрим возможность создания моделей, удовлетворяющих упомянутым выше требованиям, средствами системно-объектного подхода. Одним из способов построения системных графоаналитических моделей АП является

использование для моделирования АП так называемых блок схем алгоритмов с дорожками или, как они именуются в пакете Microsoft Office Visio – «*Basic Flowchart Shapes*» (BFSh), модифицированные с использованием системно-объектного подхода «Узел-Функция-Объект», т.е. УФО-подхода.

Диаграммы BFSh это обычные блок схемы алгоритмов, выполняемые в соответствии со стандартом ИСО 5807-85 (ГОСТ 19.701-90), блоки в которых размещаются на двух и более параллельных дорожках. Дорожки используются для моделирования взаимодействующих между собой процессов, устройств, подразделений и т.д. и т.п.

Если использовать в диаграммах BFSh три дорожки для отображения соответственно «Узлов», «Функций» и «Объектов», то эти диаграммы становятся особенно удобными для моделирования АП. Это обусловлено тем, что процедуры АП, являясь бюрократическими, во всех смыслах этого слова, представляют собой совокупности процессов, связанных между собой потоками документов. При этом в ходе выполнения процессов осуществляются функции обслуживания документов на всех этапах их жизненного цикла. Таким образом, АП предлагается рассматривать как УФО-элемент, «Узел» которого есть перекресток входных и выходных документов, «Функция» – процесс преобразования входных документов в выходные, а «Объект» – сущность, ответственная за выполнение этого процесса. Для графоаналитического моделирования АП, следовательно, на первой дорожке размещаем документы и (или) их состояния на различных этапах (дорожка «Узлы»). На второй дорожке размещаем процессы создания, согласования, утверждения, передачи и т.д. документов, входами которых являются документы или их состояния, а выходами другие документы или другие состояния документов (дорожка «Функции»). На третьей дорожке размещаем наименования структурных подразделений или должностей администраций различного уровня, ответственных за исполнение процессов и документов (дорожка «Объекты»).

Такое представление АП позволяет моделировать иерархию процессов и объектов, что соответствует требованию системности моделей. Подробнее см. работы [Зимовец, 2012, 1–4; 2014 и Белов, 2014].

Например, на рисунках 2.39, 2.40 и 2.41 показаны три уровня представления муниципальной услуги по приватизации жилья.

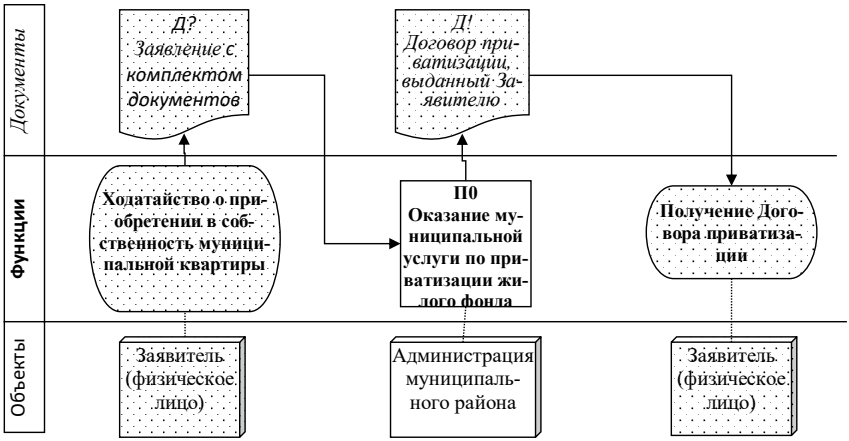


Рис. 2.39. Пример диаграммы BFSh с учетом УФО-подхода. Контекст

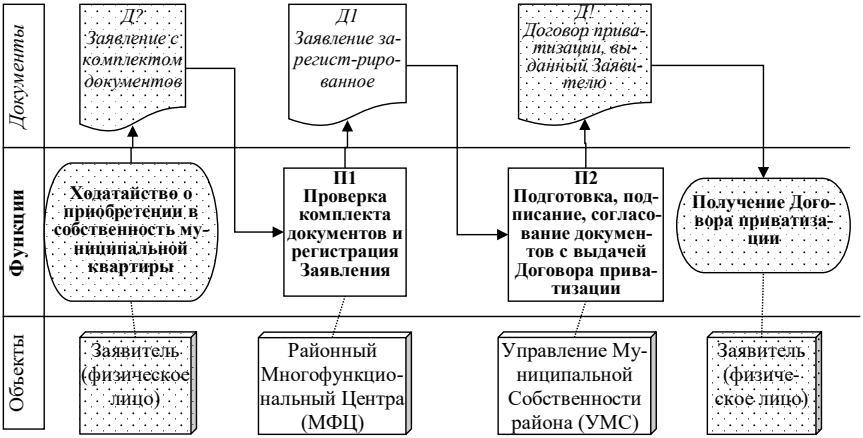


Рис. 2.40. Пример диаграммы BFSh с учетом УФО-подхода. Первая декомпозиция

Контекстные по отношению к данной услуге процессы и документы выделены точечным узором. При необходимости далее могут быть аналогичным образом декомпозированы все процессы, что позволит в явном виде проследить этапы жизненного цикла соответствующих документов. Предложенная модификация BFSh-диаграмм с использованием системного УФО-подхода (BF-UFOSh-диаграммы) позволяет моделировать АП целостно с учетом требования системности моделей и создавать диаграммы представленного вида.

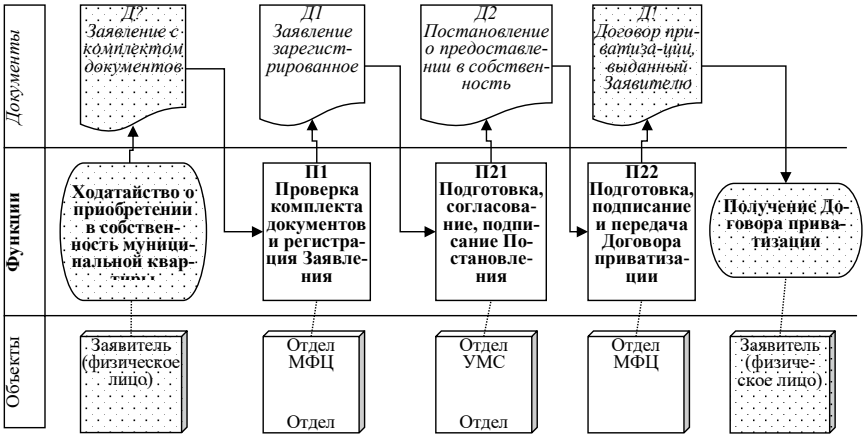


Рис.2.41. Пример диаграммы BFSh с учетом УФО-подхода. Вторая декомпозиция

При этом сформулированы новые понятия, необходимые для моделирования специфических особенностей АП:

Декомпозиция путем развертывания – это такая декомпозиция, при которой все подсистемы любого нижнего уровня иерархии данной системы изображаются на одной единственной диаграмме.

Интерфейсная декомпозиция – это такая декомпозиция, при которой система разбивается на две подсистемы: принимающую входные потоки системы и вырабатывающую ее выходные потоки.

Возможность представления иерархии процессов (и объектов), а также жизненного цикла документов чрезвычайно полезна на современном этапе анализа и организации АП. Опыт показывает, что в настоящее время описание административных регламентов носит, пока еще, итерационный характер, что связано с многочисленными уточнениями подробностей и деталей как в описаниях самих административных процессов, так и в документах и их состояниях. Кроме того, этому способствует постоянная изменчивость отечественного законодательства.

В представленных диаграммах каждый декомпозируемый процесс разбивается на два интерфейсных подпроцесса, обеспечивающих вход и выход процесса соответственно. Это обусловлено реальным способом функционирования бюрократической машины, который заключается в том, что обязательно что-то делается с входным документом, а затем что-то делается для получения выходного

документа. Это приводит к пониманию того, что любой административный процесс естественным образом можно представить как совокупность двух взаимодействующих подпроцессов: входного (работа с входным документом) и выходного (работа по созданию выходного документа).

Представленные диаграммы, кроме того, хорошо подходят для выполнения проектов по разработке и (или) настройке информационной системы, отслеживающей поток работ и автоматизирующей формирование документов, так как, по сути дела, представляют собой общепринятые стандартные схемы алгоритмов.

4.2.2. Формализованное описание системно-объектных моделей административных процедур

Для формального описания рассмотренных выше моделей проведено сравнительное исследование теории паттернов Гренандера [Гренандер, 1979] и исчисления процессов Милнера [Milner, 1989]. Результаты сравнения представлены в таблице 2.10. Сравнительное исследование позволило обосновать целесообразность и возможность использования PT и CCS для формализации системного УФО-подхода, а также тот факт, что множество конфигураций представляет собой подмножество процессных графов: $(G, \sigma) \subset (S, R)$.

Таблица 2.10

Сравнение PT и CCS

Конфигурация c в теории паттернов (PT)	**Процесс P в исчислении процессов (CCS)**
$c = (G, \sigma)$, где **G** – множество «*образующих*» (вершин графа конфигурации); **σ** – множество *соединений* связей образующих (определяющих структуру графа конфигурации)	**$P = (S, R)$, $P = (S, s^0, R)$,** где **S** – множество «*состояний*» процесса (вершин процессного графа); **R** – множество *переходов* между состояниями процесса (определяющих структуру процессного графа)
Образующая **g** (множество которых составляет множество **G**, состоящее из непересекающихся классов) – это именованный объект со связями, который характеризуется признаком **α** и показателями входных и выходных связей **β**. Рассматривается как графический формализм. *Преобразование подобия* **S** – это отображение множества **G** в себя, не выводящее образующую из своего класса	Из множества *состояний* **S** выделено *начальное состояние* **$s^0 \in S$**

Конфигурация с в теории паттернов (PT)	**Процесс P в исчислении процессов (CCS)**
Тип соединения **Σ** – это множество всех допустимых множеств соединений **σ**. *Отношение согласования* (или *связи*) **ρ** – это показатель взаимного соответствия связей (**βρβ***). *Регулярная* или *допустимая* конфигурация – это конфигурация, у которой для любого соединения **(β,β*)∈σ∈Σ** выполняется (**βρβ***). *Внутренние связи* конфигурации – связи, участвующие в соединениях, предусмотренных структурой **σ**. *Внешние связи* конфигурации **ext(c)** – связи, не участвующие в соединениях, предусмотренных структурой **σ**	Предусматривается размеченная система переходов **(S, R)** над множеством действий **Act(P)**, разбиваемым на классы, именуемые входными действиями (**α?**), выходными действиями (**α!**) и внутренними действиями (**ατ**). Действия интерпретируются как ввод, вывод или передача элемента с именем действия

В соответствии с этим сравнением система $\mathbf{e_i}$, как элемент «Узел-Функция-Объект» (УФО-элемент), может быть представлена следующим выражением:

$$\mathbf{e_i = <(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>}, \text{ где}$$

$\mathbf{(L?_i, L!_i)}$ – «Узел» УФО-элемента, причем $\mathbf{L?_i \subset L}$ – множество входных связей, $\mathbf{L!_i \subset L}$ – множество выходных связей;

$\mathbf{(P_i, P^0_i, L\tau_i)}$ – «Функция» УФО-элемента, где $\mathbf{P_i}$ – множество подпроцессов процесса, соответствующего «Функции», которые реализуются УФО-элементами нижнего яруса иерархии; $\mathbf{P^0_i \subset P_i}$ – множество интерфейсных подпроцессов (входных $\mathbf{P?_i}$ и выходных $\mathbf{P!_i}$, причем $\mathbf{P^0_i = P?_i \cup P!_i}$; в число входных связей $\mathbf{P?_i}$ входит $\mathbf{L?_i}$, в число выходных связей $\mathbf{P!_i}$ входит $\mathbf{L!_i}$); $\mathbf{L\tau_i}$ – множество внутренних связей/переходов в $\mathbf{P_i}$, осуществляемых путем передачи, ввода и вывода элементов глубинного яруса связанных подпроцессов;

$\mathbf{(n_i, \alpha_i, \beta?_i, \beta!_i)}$ – «Объект» УФО-элемента, где $\mathbf{n_i}$ – имя «Объекта» $\mathbf{(n_i \in N)}$; $\mathbf{\alpha_i}$ – множество признаков «Объекта» $\mathbf{n_i}$; $\mathbf{\beta?_i}$ – множество показателей $\mathbf{L?_i}$; $\mathbf{\beta!_i}$ – множество показателей $\mathbf{L!_i}$.

При формализации системы как УФО-элемента в наибольшей степени оказывается формализованным именно функциональный компонент этой конструкции, что особенно ценно в связи с важностью процессного подхода для моделирования АП. Однако, для обеспечения полноценной формализации графоаналитических (визуальных) диаграмм необходимо формальное описание процедур декомпозиции и агрегации элементов этих диаграмм.

Для формального описания этих процедур определены алгебраические операции на функциях. Эти операции на функциях сформулированы (таблица 2.11) по аналогии с операциями на процессах в CCS.

Таблица 2.11

Операции на процессах и функциях

Операции на процессах (CCS)	**Операции на функциях (УФО-подход)**
Процесс: $\mathbf{P = (S, s^0, R)}$	Функция: $\mathbf{F = (P, P^0, L\tau)}$
Префиксное действие (процесс $\alpha.P$ совершает действие α и продолжается как процесс **P**): $\mathbf{\alpha.P = (S \cup \{s^{0*} \notin S\}, s^{0*}, R \cup \{s^{0*}, \alpha, s^0\})}$, где запись «$s^{0*}$, α, s^0» обозначает связь/переход α между состояниями s^{0*} и s^0	Префиксное действие (функция **p?.F** совершает процесс **p?** и продолжается как функция **F**): $\mathbf{p?.F = (P \cup \{p? \notin P\}, P^0 \cup \{p?\}, L\tau \cup \{p?, l\tau_{?i}, \{p_i \in P?\}\})}$ Постфиксное действие (функция p!.F выполняет функцию F и продолжается как процесс p!): $\mathbf{p!.F = (P \cup \{p! \notin P\}, P^0 \cup \{p!\}, L\tau \cup \{\{p_i \in P!\}, l\tau_{i!}, p!\})}$
Альтернативная композиция (процесс $\mathbf{P_1+P_2}$ продолжается либо как процесс $\mathbf{P_1}$, либо как процесс $\mathbf{P_2}$): $\mathbf{P_1+P_2 = (S_1 \cup S_2 \cup \{s^{0*} \notin S_1 \cup S_2\}, s^{0*}, R_1 \cup R_2 \cup \{(s^{0*}, \alpha, s_1) \mid (s^0{}_1, \alpha, s_1) \in R_1\} \cup \{(s^{0*}, \alpha, s_2) \mid (s^0{}_2, \alpha, s_2) \in R_2\})}$	Альтернативная композиция по входу (функция $\mathbf{F_1+F_2}$ совершает процесс **p?** и продолжается либо как функция $\mathbf{F_1}$, либо как функция $\mathbf{F_2}$): $\mathbf{p?.(F_1+F_2) = (P_1 \cup P_2 \cup \{p? \notin P_1 \cup P_2\}, P^0{}_1 \cup P^0{}_2 \cup \{p?\}, L\tau_1 \cup L\tau_2 \cup \{p?, l\tau_{?1}, \{p_1 \in P?_1\}\} \cup \{p?, l\tau_{?2}, \{p_2 \in P?_2\}\})}$. Альтернативная композиция по выходу (функция $\mathbf{F_1+F_2}$ выполняется либо как функция $\mathbf{F_1}$, либо как функция $\mathbf{F_2}$, и продолжается как процесс **p!**): $\mathbf{p!.(F_1+F_2) = (P_1 \cup P_2 \cup \{p! \notin P_1 \cup P_2\}, P^0{}_1 \cup P^0{}_2 \cup \{p!\}, L\tau_1 \cup L\tau_2 \cup \{\{p_1 \in P!_1\}, l\tau_{1!}, p!\} \cup \{\{p_2 \in P!_2\}, l\tau_{2!}, p!\})}$
Параллельная композиция (процессы $\mathbf{P_1}$ и $\mathbf{P_2}$ выполняются одновременно и взаимодействуют между собой): $\mathbf{(P1 \mid P2) = ((S1, S2), (s01, s02), (((s1, \alpha, s1*) \in R1, s2 \in S2 \Rightarrow ((s1, s2), \alpha, (s1*, s2)) \in R) \wedge ((s2, \alpha, s2*) \in R2, s1 \in S1 \Rightarrow ((s1, s2), \alpha, (s1, s2*)) \in R) \wedge ((s1, \alpha, s1*) \in R1, (s2, \alpha, s2*) \in R2, \alpha \neq \tau \Rightarrow ((s1, s2), \tau, (s1*, s2*)) \in R)))}$	Параллельная композиция (функции $\mathbf{F_1}$ и $\mathbf{F_2}$ выполняются одновременно и взаимодействуют между собой): $\mathbf{(F_1 \mid F_2) = (P_1 \cup P_2, P^0{}_1 \cup P^0{}_2, L\tau_1 \cup L\tau_2 \cup \{\{p!_1 \in P!_1\}, l\tau_{12}, \{p?_2 \in P?_2\}\} \wedge \{\{p!_2 \in P!_2\}, l\tau_{21}, \{p?_1 \in P?_1\}\})}$

При этом и в исчислении процессов (CCS), и в предлагаемом «исчислении функций» речь идет об одних и тех же процессах, описываемых

с разных точек зрения. В CCS процесс **P** описывается как целое, имеющее некоторую структуру состояний **S**, а в исчислении функций **F** УФО-элементов процесс **P** описывается и целостно, и как иерархическая структура его подпроцессов p_i различного уровня.

При этом показано:

1). $p?.p!.F = p?.F \cup p!.F$.

2). $p?.(F_1+F_2) = p?.F_1 \cup p?.F_2$.

3). $p!.(F_1+F_2) = p!.F_1 \cup p!.F_2$.

4). $(F_1|F_2) = p!_2.F_1 \cup p?_1.F_2 \wedge p!_1.F_2 \cup p?_2.F_1$.

В связи с введением понятия «интерфейсная декомпозиция» особый интерес представляют определения операций на функциях, рассматриваемых на контекстном уровне, т.е. для случая $F = (\{p^0 \in P\}, \{p^0 \in P^0\}, L\tau = \varnothing) = p^0$. В этом случае представленные в таблице 2.10 определения примут вид:

Префиксное действие: $p?.p^0 = \{p?, l\tau_{?0}, p^0\}$.

Постфиксное действие: $p!.p^0 = \{p^0, l\tau_{0!}, p!\}$.

При этом:

1). $p?.p!.p^0 = p?.p^0 \cup p!.p^0$.

2). $p?_1.p?_2.p^0 = p?_1.p?_2 \cup p?_2.p^0$.

3). $p!_1.p!_2.p^0 = p!_2.p^0 \cup p!_1.p!_2$.

4). $p^0?_1.p^0_2 = p^0!_2.p^0_1$ и $p^0!_1.p^0_2 = p^0?_2.p^0_1$.

Альтернативная композиция по входу: $p?.(p^0_1+p^0_2) = p?.p^0_1 \cup p?.p^0_2$.

Альтернативная композиция по выходу: $p!.(p^0_1+p^0_2) = p!.p^0_1 \cup p!.p^0_2$.

Кроме того, $p?.p!.(p^0_1+p^0_2) = p?.p!.p^0_1 \cup p?.p!.p^0_2$.

Параллельная композиция: $(p^0_1|p^0_2) = p?_1.p^0_2 \wedge p!_2.p^0_1$.

Дано формальное определение специфического для административных процессов понятия «**интерфейсная декомпозиция**». Декомпозиция системы называется **интерфейсной**, если из диаграммы декомпозиции для процессов, составляющих функцию системы, следует равенство: $P = P^0$, т.е. в системе имеются только входные и выходные процессы. Если диаграмма соответствует данному определению, то выражение для «Функции» F_i УФО-элемента с учетом внутренней структуры будет иметь вид:

$$F_i = (P^0_i, P^0_i, L\tau_{?!i}).$$

Формальное определение интерфейсной декомпозиции позволяет высказать и обосновать следующее утверждение. Если на уровне декомпозиции внутренняя, функциональная структура УФО-элемента характеризуется $L\tau_{?!} = \{l\tau_{?!}\}$ (т.е. является одноэлементным множеством), то ее **Σ** есть «линейный порядок», а декомпозиция является интерфейсной.

Это можно обосновать следующим образом. Из того факта, что $L\tau_{?!} = \{l\tau_{?!}\}$ напрямую следует, что $P \backslash P^0 = \varnothing$. Из последнего, в свою очередь, следует, что $P = P^0$, т.е. имеет место интерфейсная декомпозиция, по определению. Если $L\tau_{?!} = \{l\tau_{?!}\}$, то, естественно, выходная связь первого подпроцесса соединена с входной связью последнего, что соответствует определению **Σ** «линейный порядок» в РТ.

В случае интерфейсной декомпозиции с линейным порядком последнее выражение для УФО-элемента на первом шаге декомпозиции системы будет иметь вид:

$$e_i = <(L?_i, L!_i), (\{p?_i\}, \{l\tau_{?!}\}, \{p!_i\}), (n_i, \alpha_i, \beta?_i, \beta!_i)>.$$

Далее рассмотрен вариант формального описания агрегирования систем как УФО-элементов на примере двух бинарных УФО-элементов e_i и e_j, представляемых на контекстном уровне с помощью следующих выражений:

$$e_i = <(\{l?_i\}, \{l!_i\}), (\{p^0_i\}), (\{\beta?_i, \beta!_i\})>;$$
$$e_j = <(\{l?_j\}, \{l!_j\}), (\{p^0_j\}), (\{\beta?_j, \beta!_j\})>.$$

Для решения данной задачи параметры «Объекта» **n** и **α** не существенны, поэтому для сокращения записи здесь и далее не учитываются.

При данном способе формализации BF-UFOSh-диаграмм **условия агрегации УФО-элементов** уточняются следующим образом. Две системы $\mathbf{e_i}$ и $\mathbf{e_j}$, представляемые в виде УФО-элементов, могут быть агрегированы в одну систему (в один УФО-элемент e_{ij}), если выполняется хотя бы одна пара условий:

$$l!_i = l?_j \text{ и } \beta!_i \subseteq \beta?_j;\ l?_i = l!_j \text{ и } \beta?_i \supseteq \beta!_j.$$

Т.е. УФО-элементы агрегируются в соответствии с правилами выполнения операции «*присоединения*» *алгебры изображений* в РТ.

Если $l!_i = l?_j$ и $\beta!_i \subseteq \beta?_j$ для упомянутых выше элементов e_i и e_j, то их соединение дает систему $\mathbf{e_{ij}}$, представляемую выражением:

$$e_{ij} = <(\{l?_i\}, \{l!_j\}), (\{p^0_{ij}\}), (\{\beta?_i, \beta!_j\})>.$$

В соответствии с операциями «Префиксное действие» и «Постфиксное действие» функциональность $\mathbf{e_{ij}}$ может быть представлена следующим образом:

$$\mathbf{p^0_{ij} = p^0?_i.p^0_j = p^0!_j.p^0_i = (\{p^0?_i\}, \{l\tau_{ij}\}\ \{p^0!_j\}).}$$

Если $l?_i = l!_j$ и $\beta?_i \supseteq \beta!_j$, то:

$\mathbf{e_{ji} = <(\{l?_j\}, \{l!_i\}), (\{p^0_{ji}\}), (\{\beta?_j, \beta!_i\})>}$ и $\mathbf{p^0_{ji} = p^0!_i.p^0_j = p^0?_j.p^0_i = (\{p?_j\}, \{l\tau_{ji}\}\ \{p!_i\})}$.

Пусть $\mathbf{e_i}$ и $\mathbf{e_j}$ представляют собой элементы, соответствующие двум альтернативным потокам работ. Элемент $\mathbf{e^R_k = <(\{l?_k\}, \{l!_{k1}, l!_{k2}\}), (\{p^{0R}_k\}), (\{\beta?_k, \beta!_{k1}, \beta!_{k2}\})>}$ – элемент разветвления (Ramification), после которого начинаются два альтернативных потока. Если $\mathbf{l!_{k1} = l?_i,\ \beta!_{k1} \subseteq \beta?_i;\ l!_{k2} = l?_j,\ \beta!_{k2} \subseteq \beta?_j}$, то подсоединение $\mathbf{e_i}$ и $\mathbf{e_j}$ к $\mathbf{e^R_k}$ дает систему $\mathbf{e^R_{kij}}$ с разветвлением потоков работ, представляемую выражением:

$$\mathbf{e^R_{kij} = <(\{l?_k\}, \{l!_i, l!_j\}), (\{p^{0R}_{kij}\}), (\{\beta?_k, \beta!_i, \beta!_j\})>.}$$

В соответствии с операцией «Альтернативная композиция по входу» функциональность $\mathbf{e^R_{kij}}$ может быть представлена следующим образом:

$$\mathbf{p^{0R}_{kij} = p^{0R}?_k.(p^0_i + p^0_j) = (\{p^{0R}?_k, p^0_i, p^0_j\}, \{p^{0R}?_k, p^0!_i, p^0!_j\}, \{l\tau_{ki}, l\tau_{kj}\}) =}$$
$$\mathbf{= p^{0R}?_k.p^0_i \cup p^{0R}?_k.p^0_j.}$$

Пусть $\mathbf{e_i}$ и $\mathbf{e_j}$ представляют собой элементы, соответствующие двум альтернативным потокам работ. Элемент $\mathbf{e^M_k = <(\{l?_{k1}, l?_{k2}\}, \{l!_k\}), (\{p^{0M}_k\}), (\{\beta?_{k1}, \beta?_{k2}, \beta!_k\})>}$ – элемент слияния (Merger) двух потоков работ в один. Если $\mathbf{l!_i = l?_{k1},\ \beta!_i \subseteq \beta?_{k1};\ l!_j = l?_{k2},}$

$\mathbf{\beta!_j \subseteq \beta?_{k2}}$, то подсоединение $\mathbf{e^M_k}$ к $\mathbf{e_i}$ и $\mathbf{e_j}$ дает систему $\mathbf{e^M_{ijk}}$ со слиянием потоков работ, представляемую выражением:

$$\mathbf{e^M_{ijk} = <(\{l?_i, l?_j\}, \{l!_k\},), (\{p^{0M}_{ijk}\}), (\{\beta?_i, \beta?_j, \beta!_k\})>.}$$

В соответствии с операцией «Альтернативная композиция по выходу» функциональность $\mathbf{e^M_{ijk}}$ может быть представлена следующим образом:

$$\mathbf{p^{0M}_{ijk} = p^{0M}!_k.(p^0_i + p^0_j) = (\{p^0_i, p^0_j, p^{0M}!_k\}, \{p^0?_i, p^0?_j, p^{0M}!_k\}, \{l\tau_{ik}, l\tau_{jk}\}) =}$$
$$\mathbf{= p^{0M}!_k.p^0_i \cup p^{0M}!_k.p^0_j.}$$

Пусть $\mathbf{e_i}$ и $\mathbf{e_j}$ представляют собой элементы, соответствующие двум альтернативным потокам работ. Элементы $\mathbf{e^R_k}$ и $\mathbf{e^M_k}$ – соответственно

элемент разветвления и элемент слияния одних и тех же двух альтернативных потоков. Если $I!_{k1} = I?_i$, $\beta!_{k1} \subseteq \beta?_i$;

$I!_{k2} = I?_j$, $\beta!_{k2} \subseteq \beta?_j$; $I!_i = I?_{k1}$, $\beta!_i \subseteq \beta?_{k1}$; $I!_j = I?_{k2}$, $\beta!_j \subseteq \beta?_{k2}$, то подсоединение e_i и e_j к e^R_k и далее – e^M_k дает систему e^{RM}_{kijk} с разветвлением и слиянием потоков работ, представляемую выражением:

$$e^{RM}_{kijk} = <(\{I?_k\}, \{I!_k\}), (\{p^{0RM}_{kijk}\}), (\{\beta?_k, \beta!_k\})>.$$

В соответствии с объединением операций «Альтернативная композиция по входу» и «Альтернативная композиция по выходу» функциональность e^{RM}_{kijk} может быть представлена следующим образом:

$$p^{0RM}_{kijk} = p^{0R}?_k.p^{0M}!_k.(p^0_i+p^0_j) = (\{p^{0R}?_k, p^{0M}!_k, p^0_i, p^0_j\}, \{p^{0R}?_k, p^{0M}!_k\}, \{I\tau_{ki}, I\tau_{kj}, I\tau_{ik}, I\tau_{jk}\}) = = p^{0R}?_k.p^{0M}!_k.p^0_i \cup p^{0R}?_k.p^{0M}!_k.p^0_j = p^{0R}?_k.p^0_i \cup p^{0R}?_k.p^0_j \cup p^{0M}!_k.p^0_i \cup p^{0M}!_k.p^0_j.$$

На практике часто встречается ситуация, когда и разветвление потоков работ, и их слияние происходят в рамках одних и тех же трех элементов: e^R_k, e_i, e^M_k. Если $I!_{k1} = I?_{k1}$, $\beta!_{k1} \subseteq \beta?_{k1}$; $I!_{k2} = I?_i$, $\beta!_{k2} \subseteq \beta?_i$; $I!_i = I?_{k2}$, $\beta!_i \subseteq \beta?_{k2}$, то соединение e^R_k, e_i, e^M_k дает систему e^{RM}_{kik}, представляемую выражением:

$$e^{RM}_{kik} = <(\{I?_k\}, \{I!_k\}), (\{p^{0RM}_{kik}\}), (\{\beta?_k, \beta!_k\})>.$$

В соответствии с операциями «Альтернативная композиция по входу» и «Альтернативная композиция по выходу» функциональность e^{RM}_{kik} может быть представлена следующим образом:

$$p^{0RM}_{kik} = p^{0R}?_k.(p^0_i+p^{0M}_k) \cup p^{0M}!_k.(p^0_i+p^{0R}_k) = =p^{0R}?_k.p^0_i \cup p^{0R}?_k.p^{0M}_k \cup p^{0M}!_k.p^0_i.$$

Исследовано понятие «эквивалентность функций» УФО-элементов по аналогии с понятием эквивалентности процессов в CCS, а также понятие «преобразование подобия» УФО-элементов по аналогии с преобразованием подобия в РТ для обеспечения построения графоаналитических моделей с минимальным числом элементов, определяющих, тем не менее, некоторое заданное поведение.

В CCS введено понятие «*сильная эквивалентность*», которое основано на следующем понимании эквивалентности процессов P_1 и P_2. Во-первых, P_1 выполняет действие $\alpha \in Act(P)$ и далее ведет себя как процесс P_1^*. Во-вторых, P_2 выполняет то же действие $\alpha \in Act(P)$ и далее ведет себя как процесс P_2^*. В-третьих, P_1^* и P_2^* эквивалентны.

Таким образом, в исчислении процессов эквивалентными считаются процессы, выполняющие одни и те же действия. Состояние

процессов при этом не учитывается. Это связано с рациональным пониманием того, что от процесса требуется получение некоторого заданного результата, зависящего от тех действий, которые выполняет процесс, а не от его состояний. В свою очередь это обусловлено тем, что действия в CCS интерпретируются как ввод, вывод или передача элемента с именем действия.

В предлагаемом исчислении функций связи между процессами также интерпретируются, как и действия в CCS, как ввод, вывод или передача элемента (с именем потока). Следовательно, можно ввести понятие эквивалентности для функций, по аналогии с пониманием эквивалентности процессов в CCS. Функции F_1 и F_2 эквивалентны $(F_1 \sim F_2)$ если:

- F_1 начинается с потока $l? \in Act(F)$, имеющего показатель $\beta?$, и далее ведет себя как функция F_1^*; F_2 начинается с того же потока $l? \in Act(F)$, имеющего тот же показатель, и далее ведет себя как функция F_2^*,
- F_1^* и F_2^* эквивалентны $(F_1^* \sim F_2^*)$.

Кроме того, для решения задачи минимизации графоаналитических УФО-моделей рассмотрено понятие «**преобразование подобия на множестве УФО-элементов**» по аналогии с понятием «*преобразование подобия*» на множестве образующих в рамках РТ. Преобразование подобия на множестве УФО-элементов E, которое представляет собой отображение множества E в себя, не выводящее элемент из своего класса, имеет вид: $f: E \to E;\ f(e_i) = e_j$. С учетом представления системы как УФО-элемента следует говорить о трех видах преобразования подобия, которые на УФО-элементах могут быть определены представленным ниже способом.

Во-первых, $f(e_i) = e_j$, где e_i и e_j такие, что $(L?_i, L!_i) = (L?_j, L!_j)$. Преобразование подобия f_y (**преобразование относительно узла – модернизация**) – это такое преобразование, при котором не меняются узловые (структурные) характеристики УФО-элемента, но изменяются его функциональные и объектные характеристики, т.е. справедливы неравенства: $(P_i, P^0_i, L\tau_i) \neq (P_j, P^0_j, L\tau_j)$; $(n_i, \alpha_i, \beta?_i, \beta!_i) \neq (n_j, \alpha_j, \beta?_j, \beta!_j)$. Т.е. модернизация есть преобразование подобия относительно узла:

$$f_y\ (<(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>) = <(L?_i, L!_i), (P_j, P^0_j, L\tau_j), (n_j, \alpha_j, \beta?_j, \beta!_j)>.$$

Во-вторых, $f(e_i) = e_j$, где e_i и e_j такие, что $(P_i, P^0_i, L\tau_i) = (P_j, P^0_j, L\tau_j)$. Преобразование подобия f_ϕ (**преобразование относительно**

функции – усовершенствование) – это такое преобразование, при котором не меняются функциональные и, естественно, узловые (**$(L?_i, L!_i) = (L?_j, L!_j)$**) характеристики УФО-элемента, но изменяются его объектные характеристики, т.е. справедливо неравенство: **$(n_i, \alpha_i, \beta?_i, \beta!_i) \neq (n_j, \alpha_j, \beta?_j, \beta!_j)$**. Т.е. усовершенствование есть преобразование подобия относительно функции:

f_ϕ $(<(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>) = <(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_j, \alpha_j, \beta?_j, \beta!_j)>$.

При этом **$f_\phi \subset f_y$**.

В-третьих, **$f(e_i) = e_j$**, где **e_i** и **e_j** такие, что **$(n_i, \alpha_i, \beta?_i, \beta!_i) = (n_j, \alpha_j, \beta?_j, \beta!_j)$**. Преобразование подобия **f_o** (**преобразование относительно объекта – восстановление**) – это такое преобразование, при котором не меняются объектные характеристики УФО-элемента, а также функциональные (**$(P_i, P^0_i, L\tau_i) = (P_j, P^0_j, L\tau_j)$**) и узловые (**$(L?_i, L!_i) = (L?_j, L!_j)$**) характеристики, но меняется экземпляр объекта, который реализует функциональность, балансирующую данный узел. Т.е. восстановление есть преобразование подобия относительно объекта:

f_o $(<(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>) = <(L?_i, L!_i), (P_i, P^0_i, L\tau_i), (n_i, \alpha_i, \beta?_i, \beta!_i)>$.

При этом **$f_o \subset f_\phi \subset f_y$**.

Данное преобразование позволяет формализовать понятия сходства УФО-элементов по аналогии с понятием «*сходство*» образующих в РТ.

Методика трансформации диаграммы BF-UFOSh в алгебраические выражения включает следующие шаги:

1. Анализ диаграммы и выявление элементов с линейным порядком соединения. Описание соединения этих элементов с помощью операций «Префиксное действие» и «Постфиксное действие». Формирование процессов верхнего яруса.
2. Анализ диаграммы с учетом процессов верхнего яруса (без элементов с линейным порядком соединения) и выявление элементов с порядком соединения типа «дерево». Описание элементов с помощью операций «Альтернативная композиция по входу» и «Альтернативная композиция по выходу».
3. Анализ диаграммы, выявление параллельных потоков работ. Описание параллельных потоков с помощью операции «Параллельная композиция».

Согласно предложенной методике, например, контекстную диаграмму, представленную на рисунке 2.39, с учетом обозначений, принятых в графической модели, можно представить следующим образом:

$$\mathbf{АП_{приват} = <(Д?, Д!), (П0),} \text{ (Админ\_Района)}\mathbf{>}.$$

Диаграмму декомпозиции на рисунке 2.40 – следующим образом:

$$\mathbf{АП_{приват} = <(Д?, Д!), (П1?.П21?.П22), (О\_МФЦ, О\_УМС)> =}$$

$$\mathbf{= <(Д?, Д!), ((П1, Д1, П21) \cup (П21, Д2, П22)), (О\_МФЦ, О\_УМС)> =}$$

$$\mathbf{= <(Д?, Д!), (П1, Д1, П21, Д2, П22),} \text{ (О\_МФЦ, О\_УМС)}\mathbf{>}.$$

Предложенная методика формализации BF-UFOSh-диаграмм позволяет минимизировать графоаналитические УФО-модели путем анализа их алгебраического описания. Методика минимизации BF-UFOSh-диаграмм сводится к удалению из модели перечисленных ниже элементов.

1. Процессов, у которых входные и выходные потоки одинаковы ($\mathbf{l\tau_{?0} = l\tau_{0!}}$):

$$\mathbf{p?.p!.p^0 = p?.p^0 \cup p!.p^0 = \{p?, l\tau_{?0}, p^0\} \cup \{p^0, l\tau_{0!}, p!\}) = \{p?, l\tau_{?0}, p^0, p!\}}.$$

2. Процессов, у которых нет выходов ($\mathbf{l!_i = 0}$) и которые могут встречаться в моделях анализа АП «как есть»:

$$\mathbf{p?.p!.p^0 = p?.p^0 \cup p!.p^0 = \{p?, l\tau_{?0}, p^0\} \cup \{p^0, \varnothing, p!\} = \{p?, l\tau_{?0}, p^0, p!\}}.$$

3. Альтернативных или параллельных потоков, которые не участвуют в формировании выходного потока, зафиксированного на уровне контекстной модели, что соответствует ситуации, когда:
   - для группы операций

$$\mathbf{p^{0R}?_k.(p^0_i+p^0_j) = p^{0R}?_k.p^0_i \cup p^{0R}?_k.p^0_j}$$

отсутствует группа операций

$$\mathbf{p^{0M}!_{k+1}.(p^0_{i+n}+p^0_{j+m}) = p^{0M}!_{k+1}.p^0_{i+n} \cup p^{0M}!_{k+1}.p^0_{j+m};}$$

   - операция $(p^0_1|p^0_2)$ определена как группа операций

$$\mathbf{p?_1.p^0_2 \oplus p!_2.p^0_1.}$$

На рисунке 2.42 представлен алгоритм минимизации графоаналитических BF-UFOSh-диаграмм путем анализа их алгебраического описания.

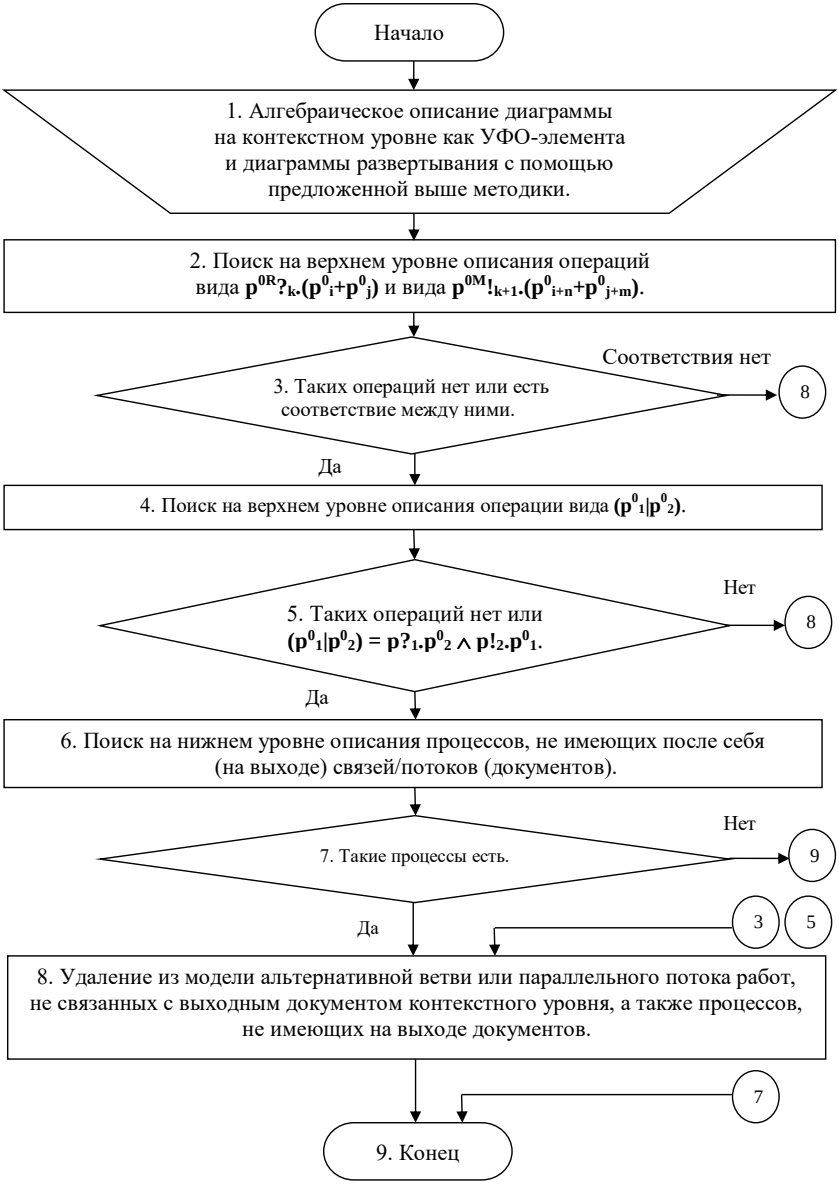


Рис. 2.42. Алгоритм минимизации диаграммы BF-UFOSh путем анализа ее алгебраического описания

4.2.3. Методика преобразования системно-объектных моделей административных процедур в описания на языке исполнения бизнес-процессов

XPDL (*XML Process Definition Language*) – это язык, предназначенный для описания определений рабочих процессов и их реализаций. XPDL предложен в качестве стандарта для импорта/экспорта описаний бизнес-процессов. На его основе решается задача интеграции программных средств разных производителей. Разработчики графических средств для моделирования и реинжиниринга бизнес-процессов встраивают в свои продукты возможность экспорта в формате XPDL, а разработчики BPM-систем – возможность импорта.

XPDL реализует граф-ориентированный подход к описанию бизнес-процессов. Граф представляет собой набор узлов, соединенных переходами. Изменение состояния бизнес-процесса соответствует переходу точки управления из одного узла графа в другой.

В XPDL нет жесткой привязки к веб-сервисам, в нем используется абстрактное понятие внешнего приложения. XPDL может описывать как работу автоматических процессов, так и человеко-машинное взаимодействие путем явного описания пользователей и ролей.

Фрагмент методики трансформации элементов диаграмм BF-UFOSh на язык XPDL представлен в таблице 2.12. На рисунке 2.43 представлен алгоритм трансформации элементов диаграмм BF-UFOSh на язык XPDL.

Идея, на основании которой предложена методика учета содержания АП, с точки зрения состава документов и этапов их обработки, в процессе их системного формализованного графоаналитического моделирования, состоит в следующем.

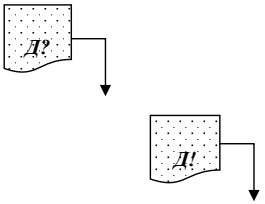
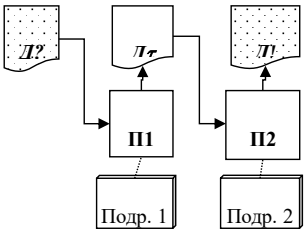
Использование классификации для придания знакам формальной системы уникального предметно-ориентированного содержания превращает формальную систему в систему формально-семантическую. Автоматизация выполнения АП в рамках государственной программы «Электронная Россия» происходит за счет автоматизации процессов формирования (создания, корректировки, согласования и т.п.) документов.

Поэтому, с точки зрения процессов и объектов, вполне достаточно, чтобы их знаки различались исключительно в рамках описываемой данным выражением услуги, т.е. имели только формальное значение. Если иметь классификацию всех документов (и их состояний), использующихся при оказании услуг, то использование знаков для документов из классификации при формальном описании услуги позволит получать выражения, которые будут отражать не только структуру

соответствующей АП, но и ее содержание, с точки зрения формируемых документов.

Таблица 2.12

Фрагмент методики трансформации элементов BF-UFOSh на язык XPDL

Графические элементы диаграмм BF-UFOSh	**Формальное описание элементов диаграмм BF-UFOSh**	**Описание на XPDL**
	**(*Д?*, *Д!*), (β?, β!)**	<Transition id=«***Д?***» from=«**П?**» to=«**П0**» sign=«**β?**» /> <Transition id=«***Д!***» from=«**П0**» to=«**П!**» sign=«**β!**» />
	**(П0)**	<Activity id=«**П0**»> <BlockActivity /> … </Activity>
	**(n, α)**	<Participant type=«OrganisationUnit» />
	**<(*Д?*, *Д!*), (П1, Дτ, П2), (**Подр_1, Подр_2**)>**	<Activity id=«**П1**»> <Implementation> <SubFlow /> … </Implementation> </Activity> <Activity id=«**П2**»> <Implementation> <SubFlow /> … </Implementation> </Activity>

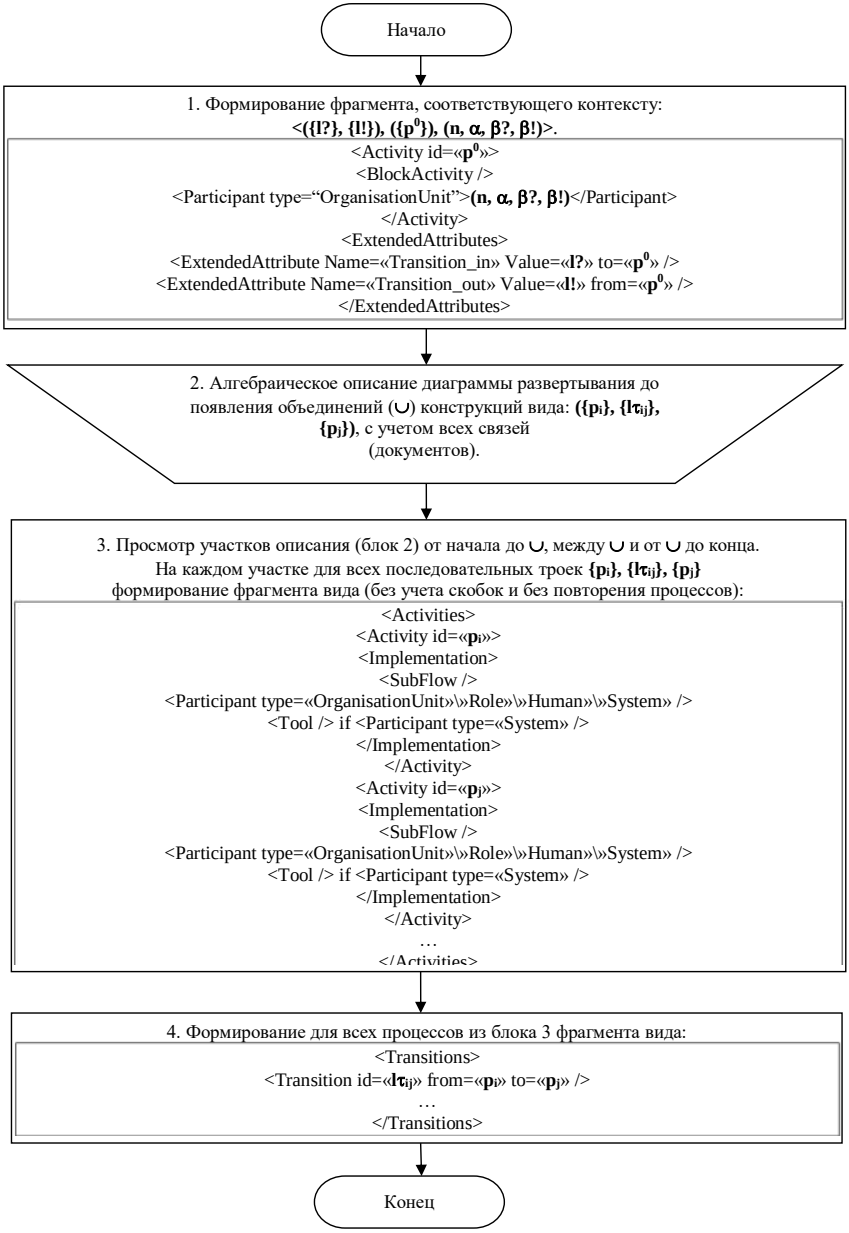


Рис. 2.43. Алгоритм трансформации алгебраического описания диаграммы BF-UFOSh в конструкции языка XPDL

Описание АП не только на формальном, но и на семантическом уровне (с учетом классификации документов), позволяет выявлять классы услуг, сходных не только по структуре, но и по содержанию (составу документов или состоянию документов).

Это позволяет унифицировать применение информационной системы, автоматизирующей оказание государственных и муниципальных услуг, так как можно обеспечить применение одной программной системы или одной настройки такой системы для автоматизации оказания множества услуг.

Разработанный в результате данного исследования метод системного формализованного графоаналитического моделирования АП апробирован в ходе выполнения совместно с ООО «СВЕТО-СОФТ» пилотного проекта по обеспечению оказания муниципальных услуг населению в электронном виде в администрациях Грайворонского и Корочанского районов Белгородской области.

С помощью предложенного метода в целях обеспечения проектирования программной системы выполнено моделирование, в частности, регламента исполнения АП «*Предоставление земельных участков для индивидуального жилищного строительства*» (**АР2**). Визуальное представление данной АП показано на рисунке 2.44. Для преобразования данной диаграммы в алгебраическое выражение предварительно была формализована функциональная составляющая в виде следующего выражения:

(П1?.П211)?.((П2211?.П2212)+(П2121?.П2122?.П22)) = (П1?.П211)?.(П2211?.П2212) ∪ (П1?.П211)?.(П2121?.П2122?.П22) = ((П1, *ИС111, П211), *РА21*, (П2211, *РА1*, П2212)) ∪ ((П1, *ИС111**, П211), *РА21*, (П2121, *ИС211**, П2122, *ИС221*, П22)) =**

= {П1, *ИС111, П211, *РА21*, П2211, *РА1*, П2212, П2121, *ИС211**, П2122, *ИС221*, П22}.**

Далее для диаграммы **АР2** получено общее формальное выражение следующего вида:

$АП_{АР2}$ = <(*ИС111?*, *ОР11!*), ((П1, *ИС111, П211), *РА21*, (П2211, *РА1*, П2212)) ∪ ((П1, *ИС111**, П211), *РА21*, (П2121, *ИС211**, П2122, *ИС221*, П22)), (О_МФЦ, О_УМС)> = <(*ИС111?*, *ОР11!*), (П1, *ИС111**, П211, *РА21*, П2211, *РА1*, П2212, П2121, *ИС211**, П2122, *ИС221*, П22), (О_МФЦ, О_УМС)>.**

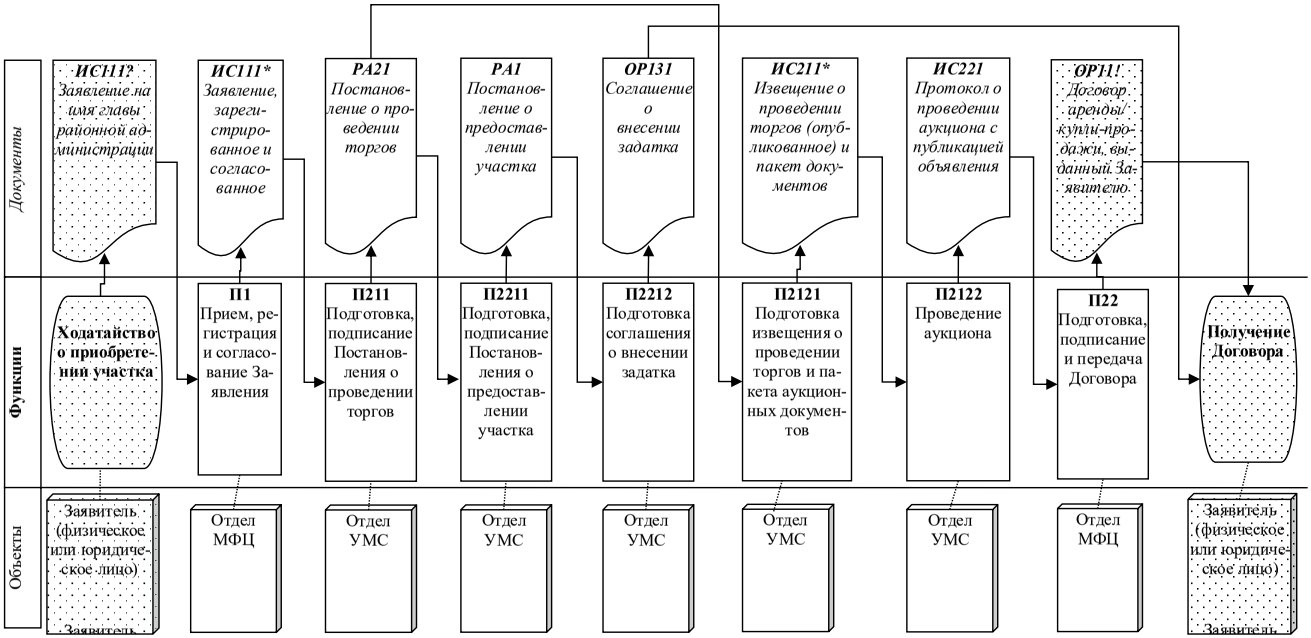


Рис. 2.44. Диаграмма декомпозиции путем развертывания BF-UFOSh для регламента AP2.

Формальное описание функциональной составляющей диаграммы регламента **АР2** на самом верхнем уровне описания показывает, что данный регламент нуждается в минимизации (что характерно для диаграмм «как есть»), так как для операции вида $p^{0R}?_k.(p^0_i+p^0_j)$ отсутствует операция вида $p^{0M}!_{k+1}.(p^0_{i+n}+p^0_{j+m})$. И действительно, в данном случае, после разветвителя **(П1?.П211)** альтернативные ветви не сходятся.

При этом ветвь, начинающаяся с **П2121**, приводит к интерфейсному (выходному) для $\mathbf{АП_{АР2}}$ процессу **П22**, а ветвь от **П2211** – нет. Следовательно, эту ветвь можно и нужно убрать. Удаление лишней ветви привело к тому, что функциональная составляющая диаграммы регламента **АР2** получила следующее выражение:

(П1?.П211)?.(П2121?.П2122?.П22) = (П1, *ИС111, П211), *РА21*, (П2121, *ИС211**, П2122, *ИС221*, П22) = {П1, *ИС111**, П211, *РА21*, П2121, *ИС211**, П2122, *ИС221*, П22}**,

а общее формальное выражение для диаграммы регламента **АР2** приобрело следующий вид:

$\mathbf{АП_{АР2}}$ = <(*ИС111?, ОР11!*), (П1, *ИС111, П211, *РА21*, П2121, *ИС211**, П2122, *ИС221*, П22), (О_МФЦ, О_УМС)>**.

4.3. Системно-объектный метод представления знаний

Интеллектуальный капитал организации – понятие, которое приобрело смысл в последние два десятилетия. Чаще всего интеллектуальный капитал организации определяют, как знания, навыки и производственный опыт конкретных людей (человеческие авуары) и нематериальные активы, включающие патенты, базы данных, программное обеспечение, товарные знаки и др., которые производительно используются в целях максимизации прибыли и других экономических и технических результатов.

Многие ученые отмечают, что интеллектуальный капитал сейчас имеет гораздо большую стоимость чем любой материальный. Основной «золотой запас» любой компании – знания и опыт ее специалистов. От того, как этот запас будет сохраняться и пополняться, зависит, в конечном итоге, эффективность услуг, ориентированных на клиентов, а значит, и конкурентоспособность компании на рынке. На данном этапе специалисты выделяют проблему сохранения интеллектуального капитала «в стенах» организации. Данная проблема заключается в том, что сотрудники компании, как основные носители интеллектуального капитала, являются очень ненадежным способом хранения, так как всегда может

возникнуть ситуация, при которой сотрудник может перейти в другую компанию или просто уволиться. В таком случае организация безвозвратно теряет часть своего интеллектуального капитала. На данном этапе возникает вопрос: как хранить знания, в каком виде их представлять для сохранения и дальнейшего использования. Таким образом, в последнее десятилетие можно наблюдать повышающийся интерес компаний к такому понятию как «организационное знание». Переход мировой экономики в новое качественное состояние (экономики, основанной на знаниях) непосредственно связан с повышением роли внутренних, нематериальных ресурсов предприятия (интеллектуального капитала организации), важнейшими из которых выступают знания. Мировой финансовый кризис заставил современные предприятия мобилизовать свой интеллектуальный потенциал и задуматься о механизме управления организационными знаниями, рассматривать данные процессы как главное условие для создания конкурентных преимуществ предприятия в условиях нестабильности и неопределенности внешней среды.

Организационное знание выражается в улучшении продуктов, процессов, технологий и позволяет организации оставаться конкурентоспособной и жизнеспособной. Приобретая новое знание первой, организация может вместе с этим приобрести уникальное конкурентное преимущество. Таким образом, организационное знание является стратегическим активом. Это предполагает, что организации, желающей остаться конкурентоспособной, следует развивать механизмы приобретения необходимых знаний и распространения знаний точно, последовательно, своевременно, в необходимой форме всем, кому они нужны в организации.

Управление организационными знаниями становится предметом профессионального труда когнитолога – специалиста, который формализует организационные знания и делает их доступными для всеобщего пользования. Работа «инженера по знаниям» включает несколько направлений деятельности. В первую очередь, когнитолог должен организовать информационные потоки таким образом, чтобы удовлетворять потребности каждого конкретного пользователя.

Для обеспечения управления организационными знаниями, в первую очередь, необходимо эти знания иметь в явном, причем, в настоящее время, в компьютерном виде. Организационное знание представляет собой систему, которая, в отличие от знаний вообще, включает в себя одновременно знания о структурных, функциональных и субстанциальных характеристиках организационной системы. Исследование традиционных и современных различных (нейронных,

гибридных и т.д.) способов представления знаний, с точки зрения учета системных свойств знаний организационных, позволяет утверждать, что наиболее адекватным и эффективным средством описания последних является использование системно-объектного подхода «Узел-Функция-Объект» [Жихарев, 2011; 2013, 1 и 2].

Для создания метода представления организационных знаний в терминах «Узел-Функция-Объект» необходимо адаптировать модели, получаемые средством системно-объектного УФО-подхода, к требованиям моделей организационных знаний. Для этого могут быть использованы возможности традиционных моделей знаний (семантической, продукционной и фреймовой), так как каждая из них в отдельности соответствует отдельным элементам организационного знания.

4.3.1. Графические средства описания структурных характеристик организационных знаний

С точки зрения семантической модели знаний (ее возможностей описывать структурные характеристики организационных знаний), понятия и отношения между ними естественным образом представляются в терминах УФО-подхода, так как любая УФО-модель, так же, как и семантическая сеть, представляет собой граф, вершинами которого являются УФО-элементы (понятия), а дуги – связи между ними (отношения между понятиями).

По определению, семантическая сеть представляет собою граф, вершинами которого являются понятия, а ориентированными ребрами – отношения между понятиями. Пример представления семантической сети в программной среде UFO-toolkit (в виде УФО-модели) показан на рисунке 2.45.

На рисунке видно, что узлы УФО-модели – выступают в роли вершин семантической сети, а связи – в роли отношений между вершинами. Таким образом, с точки зрения программной реализации семантической сети в виде УФО-модели, достаточно понятия семантической сети хранить в виде узлов модели, а отношения между понятиями в виде связей между узлами. Специфической особенностью такой семантической сети, связанной с использованием УФО-подхода, является тот факт, что все возможные отношения между понятиями определяются иерархией связей\потоков УФО-модели, создаваемой перед построением сети. Таким образом, процесс создания семантической сети в терминах УФО-подхода можно представить в виде простого алгоритма, как показано на рисунке 2.46.

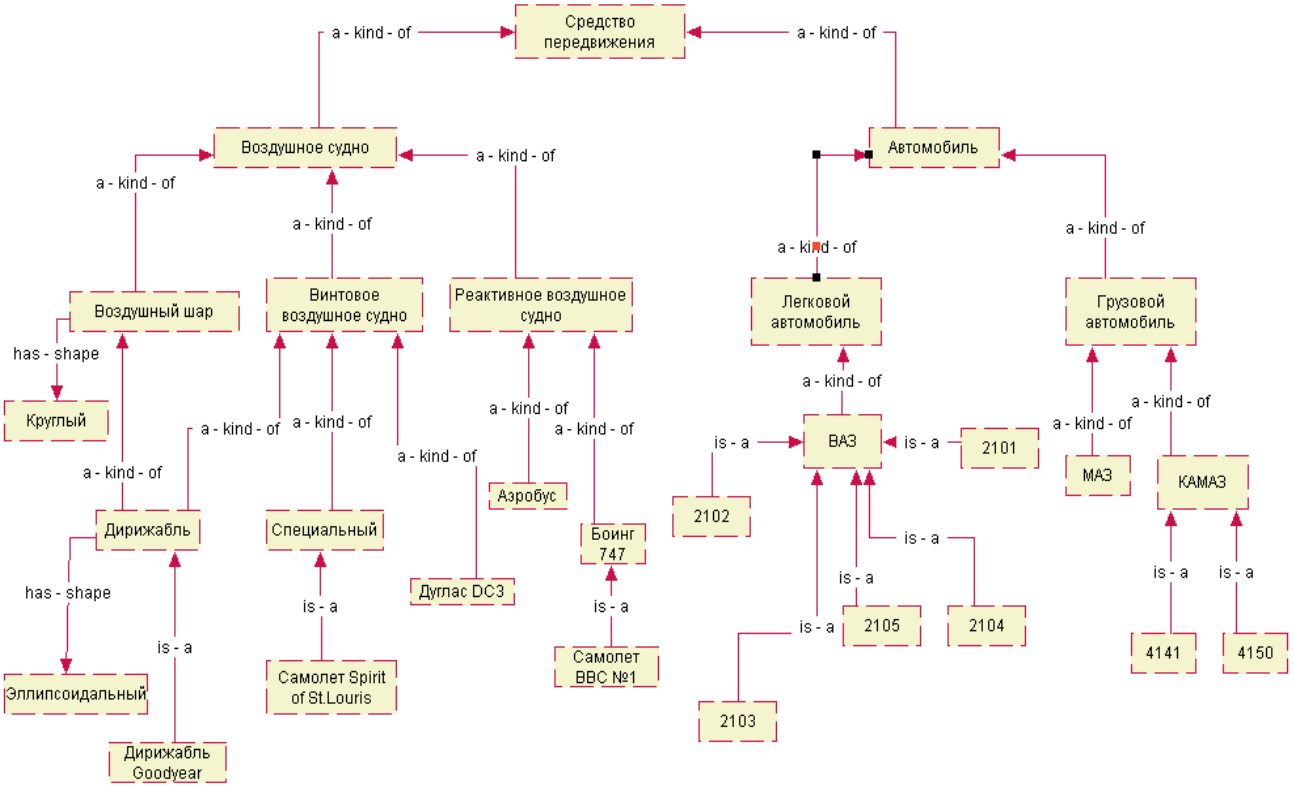


Рис. 2.45. Семантическая сеть в терминах УФО-подхода

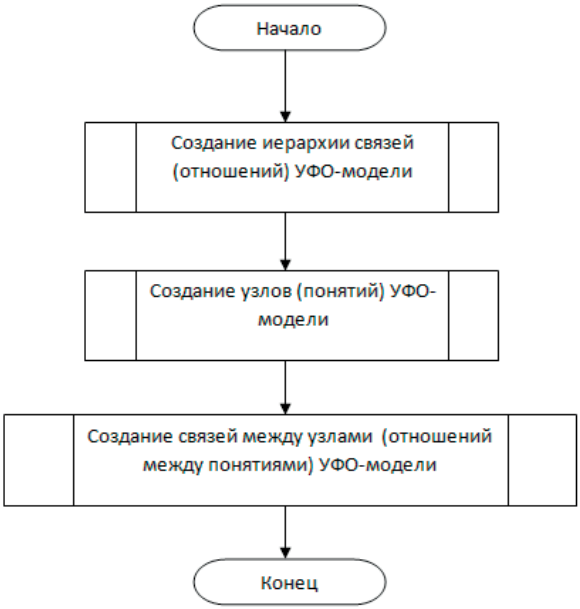


Рис. 2.46. Алгоритм моделирования семантической сети в терминах УФО-подхода

Механизм вывода на семантической сети заключается в поиске подграфа, содержащего необходимое понятие. Таким образом, механизм вывода в случае хранения понятий и отношений между ними с помощью УФО-модели, будет заключаться в поиске узла, соответствующего искомому понятию и определении связей искомого понятия (узла) с другими понятиями (узлами).

4.3.2. Графические средства описания функциональных характеристик организационных знаний

С точки зрения продукционной модели знаний (ее возможности описывать функциональные характеристики знаний организационных), правило $A \rightarrow B$ эквивалентно некоторой функции $F(A)=B$, которую в терминах УФО-подхода можно описать с помощью соответствующего функционального узла (узла, для которого определена функция).

Согласно продукционной модели знаний, продукцию (правило) можно представить с помощью нескольких компонент:

$$R = \{R1, R2, A \rightarrow B, C\} \quad (2.86)$$

Смысл компонент правила следующий:

- **R1** – предметная область продукции;
- **R2** – предусловие продукции;
- **A→B** – ядро продукции;
- **C** – постусловие продукции.

В самом простом случае продукция состоит из двух элементов: имя продукции и ядро. Ядро продукции представляет собою правило вида:

$$\textbf{IF A THEN B}, \tag{2.87}$$

где **A** – условие продукции, **B** – заключение продукции. Исходя из этого, любое простое правило можно рассматривать как функцию преобразования условия в заключение, т.е. правило **A→B** эквивалентно некоторой функции **F(A)=B**. В терминах УФО-подхода продукцию можно рассматривать как функцию УФО-элемента, на вход которой поступают потоки посылок, а на выходе – потоки заключений. Визуально, продукция с именем «Правило 1» и ядром **A→B** в терминах УФО-подхода может быть представлена, как показано на рисунке 2.47.

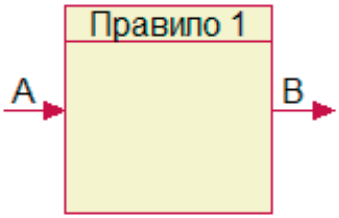


Рис. 2.47. Представление правила в терминах УФО-подхода

Формально, любое правило, в терминах УФО-подхода, представляет собою граф:

$$\textbf{R = (N, L)} \tag{2.88}$$

Представленный граф будет иметь одну вершину с названием правила **N={«Правило 1»}** и два ребра: **L={A,B}**, причем: **A=(0,Правило 1)**, **B=(Правило 1, 0)** – в случае когда «Правило 1» является вершиной графа. Если же «Правило 1» представляет собою подграф графа **R**, тогда связи **A** и **B** будут связующими дугами подграфа.

В продукционной модели представления знаний процесс логического вывода очень прост. Например, имеется множество правил вида **A→B**, относящихся к некоторой предметной области. Когда на данное

множество правил воздействует запрос в виде посылки или факта «**X**», организуется поиск правил, у которых первая часть (условие) равна поступившему запросу (**A=X**). Как только правило найдено, дается заключение **B**. Далее организуется поиск правил, у которых первая часть равна **B** и, если такие правила найдены, дается очередное заключение и так далее, пока не закончится множество правил, соответствующих запросам.

Рассмотрим процедуру логического вывода при хранении правил с помощью УФО-подхода на примере базы знаний (базы правил) с абстрактными правилами (рисунок 2.89). На вход базы знаний, поступает запрос, который представлен входящей связью. В роли запроса может быть факт, посылка и т.д. Например, если в базе знаний хранятся правила выбора снаряжения для отдыха, тогда в роли запроса может быть факт «Место отдыха – горы». Формально база правил представляет собою граф:

$$\mathbf{KB = (R, F)}, \qquad (2.89)$$

где **R**-множество правил, а **F** – множество фактов. Запрос представлен в виде входящего ребра «**C**», которое является свободным, т.е. не соединено ни с одной вершиной. В таком случае процедура логического вывода будет состоять из нескольких действий:

- поиск входящей связи правила, имя которого «X» соответствует запросу «C» (на рисунке 2.48 видно, что для запроса «C» подходит правило 2);
- если, входящая связь, удовлетворяющая условию X=C, найдена, тогда связь запроса соединяем с входящей связью соответствующего правила, если входящих связей, которые удовлетворяют условию, нет, тогда ответ считается нулевым;
- после соединения связи запроса и связи правила, необходимо рассматривать выходящую связь правила в виде запроса на следующей итерации и повторять первую процедуру поиска, и так до тех пор, пока поиск не приведет к отрицательному результату.

После выполнения процедуры логического вывода образуется цепочка использованных правил в виде процесса, где состояниями процесса являются правила базы знаний, а переходами – потоки посылок и заключений. Результатом процедуры логического вывода будет свободная выходящая связь последнего использованного правила. Пример выполнения процедуры логического вывода показан на следующем рисунке:

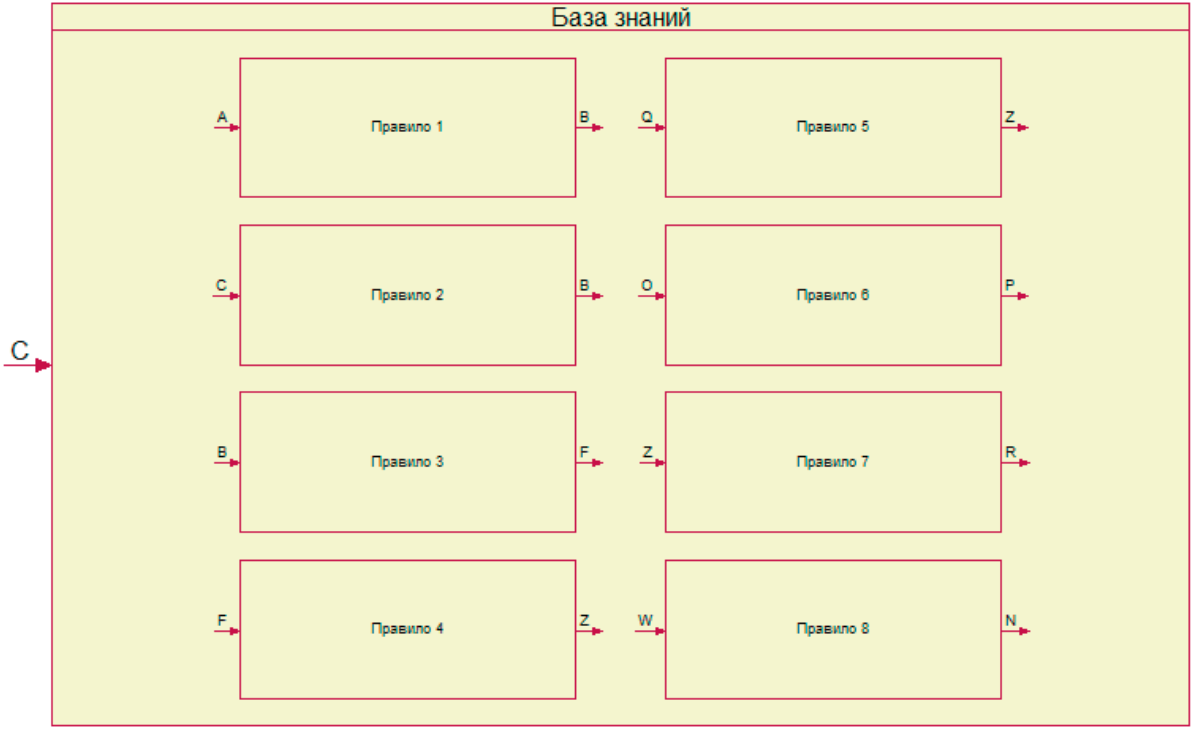


Рис. 2.48. База продукционных правил в терминах УФО-подхода

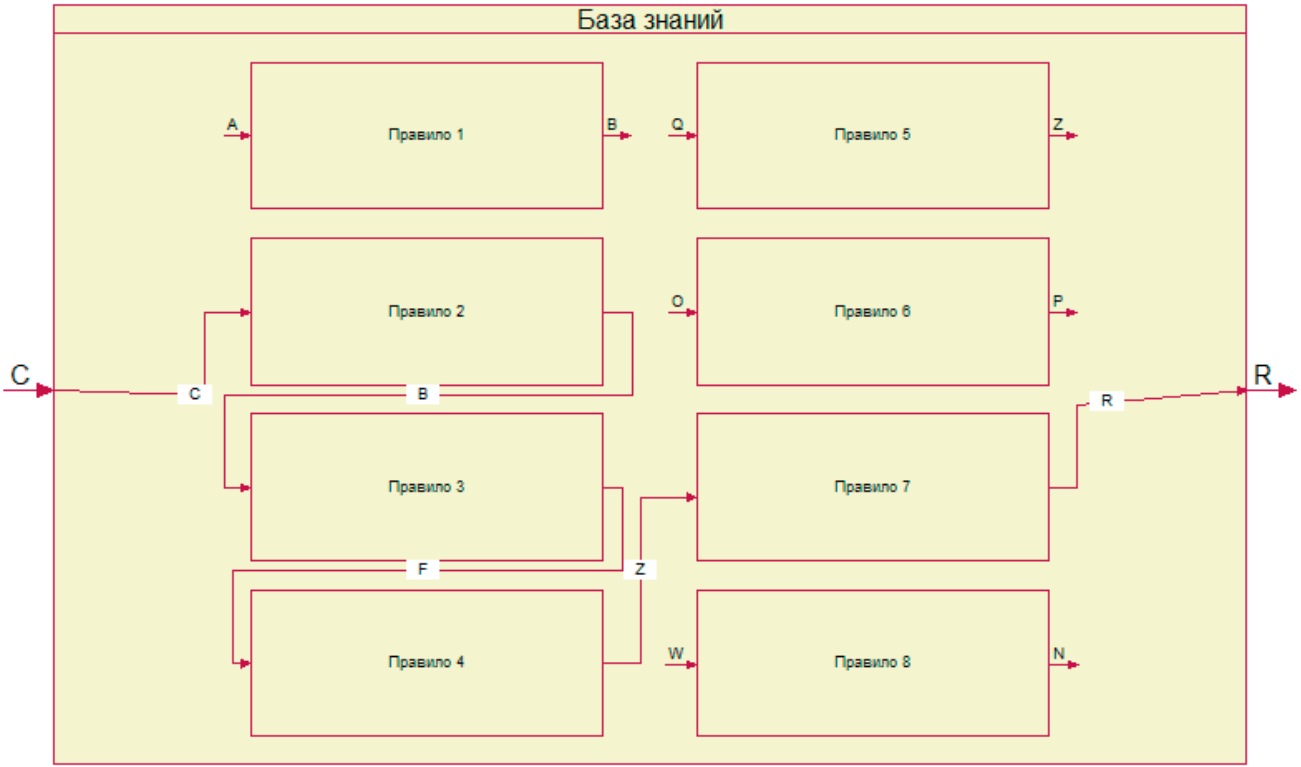


Рис. 2.49. Логический вывод на правилах в терминах УФО-подхода

На рисунке 2.49 показано, как путем перебора правил и соединения подходящих связей, мы получаем цепочку логического вывода и соответствующее заключение. Из полученного вывода можно сформировать новые знания, например, **C→R**, что соответствует запросу и результату, так же можно сформулировать правила из промежуточных посылок и действий. Это обеспечивает самообучение информационной системы, основанной на продукционной модели представления знаний.

Весь механизм логического вывода на правилах в терминах УФО-подхода можно представить в виде блок схемы, показанной на рисунке 2.50.

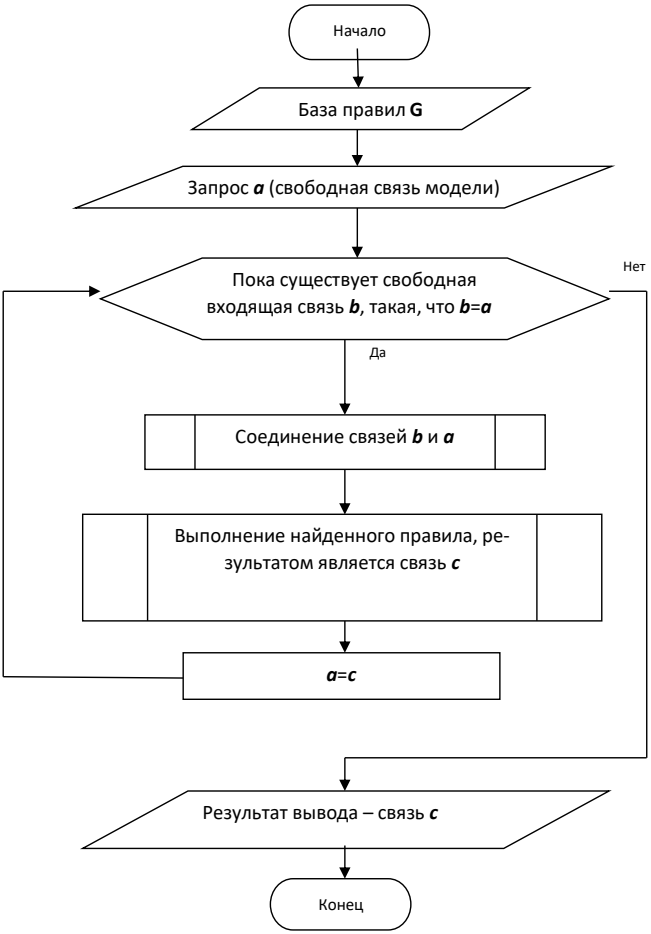


Рис. 2.50. Алгоритм вывода на правилах в терминах УФО-подхода

Пользователь вводит базу правил в виде графа:

$$G\ (M, L), \tag{2.90}$$

где **M** – множество правил в виде узлов, **L** – множество посылок и заключений в виде свободных входящих и выходящих связей. Таким образом, множество $L = B \cup C$, где **B** – множество свободных входящих связей узлов (посылок), **C** – множество свободных исходящих связей узлов (заключений). После ввода базы правил, к базе делается запрос в виде входящей связи ***a***, после чего она сравнивается с элементами множества **B**, если находится связь ***b***, такая что $b = a$, тогда срабатывает соответствующее правило.

В результате работы алгоритма, получается цепочка логических вычислений в виде графа, последнее выходящее ребро, которого и будет результатом работы системы. С точки зрения реализации алгоритма, он основан на выполнении двух операций: поиск связи и соединения двух связей в одну.

4.3.3. Графические средства описания объектных характеристик организационных знаний

С точки зрения фреймовой модели знаний (ее возможности описывать объектные характеристики знаний организационных), фрейм, как структуру данных, можно рассматривать как узел и объект УФО-элемента, а слоты фрейма, как функции узла УФО-элемента.

В общем случае любой фрейм можно представить в следующем виде:

$$F=[(r_1,v_1), (r_2,v_2),\ldots,(r_n,v_n)], \tag{2.91}$$

где **F** – имя фрейма, **r** – имя слота, **v** – значение слота.

Рассмотрение слотов как функций при описании фреймовых знаний с помощью УФО-подхода учитывает тот факт, что слот может содержать не только конкретное значение, но и другие фреймы.

На рисунке 2.51 изображен фрагмент фреймовой сети, на котором представлены иерархические знания о нитях из синтетических волокон. Стоит отметить, что принадлежность класса к классу можно так же описать с помощью вложенности УФО-элементов, т.е. когда конкретную функцию (слот фрейма) реализуют вложенные фреймы (УФО-элементы).

Далее рассматривается алгоритм описания фреймовой сети, представленной на рисунке 2.51 с помощью УФО-подхода.

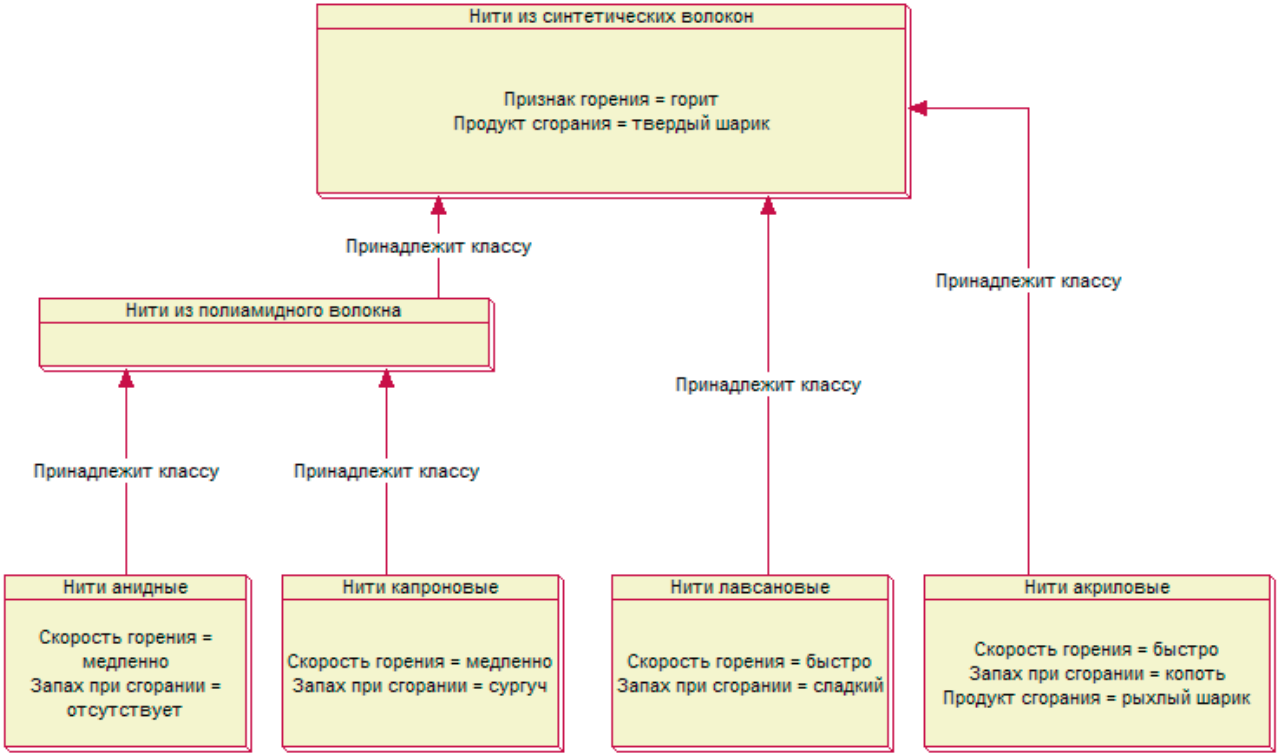


Рис. 2.51. Фреймовая модель в терминах УФО-подхода

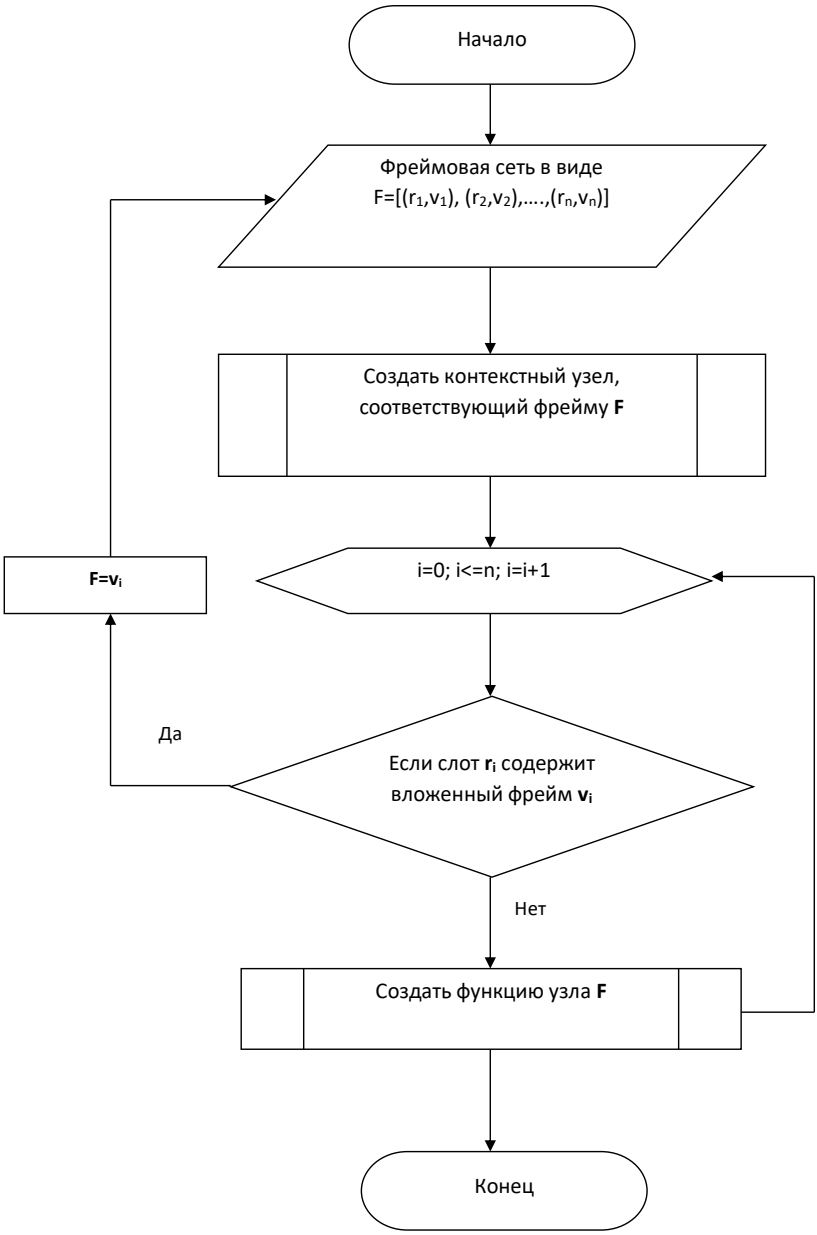


Рис. 2.52. Алгоритм описания фреймовой модели знаний в терминах УФО-подхода

Для описания фреймовой сети, инициализируется фрейм верхнего уровня и соответствующий ему узел «**F**». В цикле обрабатываются все слоты фрейма: если слот представляет собой значение, то создается функция для данного слота с соответствующим телом, если значение слота содержит некоторый вложенный фрейм, то выполняются все предыдущие шаги, и так до тех пор, пока иерархия фреймов не закончится. Блок-схема алгоритма описания фреймовой модели знаний с помощью УФО-подхода представлена на рисунке 2.52.

Рассмотренные алгоритмы описания знаний с помощью технологии графоаналитического моделирования «Узел-Функция-Объект», позволяют программно реализовать автоматизированный перевод баз знаний, представляемых традиционными средствами, в термины УФО-подхода и последующую их обработку.

Способ представления знаний, основанный на графоаналитическом системно-объектном подходе «Узел-Функция-Объект», таким образом, можно рассматривать как универсальный и интегрирующий с точки зрения представления организационных знаний, так как существует возможность описания с помощью данного подхода всех традиционных моделей знаний, которые, в свою очередь, позволяют представлять отдельные элементы организационного знания.

Следовательно, анализ результатов адаптации УФО-подхода к описанию структурных, функциональных и объектных характеристик организационного знания на примерах представления традиционных моделей знаний в терминах «Узел-Функция-Объект» показывает, что УФО-модель организационной системы (или организационного знания) может выступать в качестве и сетевой, и продукционной, и фреймовой моделей. Это позволяет рассматривать способ представления организационных знаний в терминах УФО-подход как интегральный **системно-объектный метод представления знаний** (СОМПЗ), который интегрирует и логическую, и структурную, и процедурную парадигмы представления знаний.

Рассмотренный способ описания организационных знаний является универсальным, так как позволяет описать объектные характеристики, структурные характеристики и функциональные характеристики изучаемой предметной области. Информационная система, основанная на таком методе хранения организационных знаний, позволит хранить и обрабатывать опыт организации в удобном визуальном виде, чего не позволяют реализовать традиционные методы представления знаний. В рамках СОМПЗ мы можем хранить структурные, функциональные и объектные характеристики знаний. Причем на одной модели СОМПЗ имеется возможность использовать продукции, фреймы и сети, что делает модель очень информативной.

4.3.4. Формализация системно-объектного метода представления знаний

Как было показано выше СОМПЗ позволяет (и должен) описывать структурные, функциональные и субстанциальные характеристики организационных знаний, так как представляет собой структурированный набор УФО-элементов (т.е. УФО-модель ОЗ). Для обеспечения автоматизированной обработки моделей организационных знаний и обеспечения вывода на них, необходима формализация СОМПЗ с учетом названных характеристик.

Кроме того, создание универсального метода представления знаний предполагает не только объединение возможностей различных способов представления знаний в одной модели, но и обеспечение вывода на этой универсальной модели. Это может быть достигнуто за счет единообразного описания различных способов представления знаний и универсальной модели знаний (в данном случае системно-объектной УФО-модели) с помощью единого математического аппарата. В настоящее время, по мнению авторов, для создания такого аппарата может быть использована теория графов, теория паттернов Гренандера, исчисление процессов (Calculus of Communication Systems – CCS) Милнера, а также исчисление объектов Абади-Карделли. Обоснование возможности создания требуемого аппарата на основе упомянутых математических теорий представлено в таблице 2.13, в которой комментарии и выводы автора выделены курсивом.

Таблица 2.13

Связь УФО-подхода и математических теорий

УФО-подход	**Теория паттернов**	**Алгебры изображений, процессов, объектов и теория графов**
Узел: перекресток входящих и выходящих связей	**Изображение:** класс эквивалентности, индуцированный на множестве конфигураций, который содержит информацию относительно несоединенных (внешних) связей конфигураций *Понятие «изображение» теории паттернов соответствует понятию «узел» УФО-подхода*	**Алгебра изображений** Определена на множестве регулярных конфигураций, на котором заданы преобразования подобия и операторы присоединения и аннигиляции. *Алгебра изображений позволяет формально описывать взаимодействие систем в целом, которое и осуществляется, собственно, на уровне структурных элементов, т.е. узлов. Следовательно, алгебра изображений – это алгебра для узлов.* **Теория графов** Граф G(V,L) представляет собою множество вершин V и множество дуг L, соединяющие вершины графа.

		Любую УФО-модель можно рассматривать как граф. Узлы модели представляют собою вершины графа а связи модели – ориентированные дуги графа. Таким образом, средства теории графов позволяют формально представить структурные характеристики УФО-модели
Функция: процесс преобразования входа в выход	**Конфигурация:** комбинация образующих, получающаяся при соединении их связей. Кроме того, конфигурация рассматривается как формула функции, задаваемой изображением. *Конфигурация зависит только от связей образующих, т.е., по сути дела, это комбинация не образующих, а комбинация изображений. Следовательно, конфигурация представляет собой описание процесса, т.е. описание функции в терминах УФО-подхода*	**Исчисление процессов (CCS)** Процесс P есть тройка: (S, s0, R), которая задается процессным графом, где S – множество состояний процесса, $s0 \in S$ – начальное состояние, R – множество переходов в S путем выполнения некоторых действий. (S,R) – размеченная система переходов над множеством действий Act(P). Множество действий Act (входныхα?, выходных α!, внутренних $\alpha\tau$), которые интерпретируются как ввод, вывод или передача объекта с именем действия. *Процессный граф можно рассматривать как конфигурацию, в которой состояниям процесса соответствуют образующие/ изображения, а переходам с выполнением действий соответствуют связи/потоки. Следовательно, алгебру процессов можно рассматривать как средство формального описания систем в виде конфигураций, т.е. на уровне функций. Тем самым алгебра процессов – это алгебра функций!*
УФО-подход	**Теория паттернов**	**Алгебры изображений, процессов, объектов и теория графов**
Объект: субстанция, реализующая функцию и занимающая данный функциональный узел	**Образующая:** объект (именованный), обладающий некоторыми признаками α, а также входящими и выходящими связями, в свою очередь характеризующимися некоторыми показателями β. *Понятие «образующая» теории паттернов соответствует понятию «объект» УФО-подхода*	**Исчисление объектов** Объект O представляет собой набор полей и методов. Использование метода объекта – это вызов метода, изменение метода – это переопределение. Поле – это частный случай метода (константный метод). Таким образом, стоит задача описания понятия «образующая» в теории паттернов с помощью исчисления объектов

Представленная таблица позволяет сделать следующие выводы:

- Алгебра изображений теории паттернов, хотя формально и определяется на множестве конфигураций, реально оперирует только

со связями (не зависит от состава конфигураций, т.е. от «образующих») и, следовательно, может рассматриваться как алгебра только для «узлов» (в терминах УФО-подхода). Конфигурация теории паттернов есть граф, вершины которого объекты образующих, а дуги – замкнутые связи этих образующих. Так же теория графов собственными средствами позволяет формально представить структурные характеристики УФО-элементов, т.е. узлы и связи между узлами.

- Алгебра процессов формально оперирует теми же объектами, что и алгебра изображений, т.е. конфигурациями, но учитывает их внутреннюю структуру. Таким образом, это алгебра может рассматриваться как алгебра для «функций» (в терминах УФО-подхода). При этом, очевидно, эти два алгебраических аппарата (алгебра изображений и алгебра процессов) тесно связаны между собой. Процесс в CSS есть граф, вершины которого состояния, а дуги – переходы между ними, которые интерпретируются как ввод, вывод или передача объекта.
- Исчисление объектов это, с одной стороны, алгебра для «объектов» УФО-подхода, а, с другой, аппарат для оперирования «образующими» теории паттернов с учетом их объектных признаков, которые не учитываются средствами алгебры изображений. Так как это исчисление изначально направлено на формализацию объектно-ориентированного программирования, то оно также связано с графами через диаграммы объектов UML.
- Все перечисленные выше алгебраические аппараты или непосредственно используют, или потенциально могут использовать графические формализмы, позволяющие, в конце концов, иметь графовое представление их алгебраических конструкций и, таким образом, тесно связаны с теорией графов.

Представленное во второй главе краткое описание сути УФО-подхода к представлению организационных знаний, показанная связь данного подхода с математическими теориями, а также сделанные выше выводы о взаимосвязи этих теорий позволяют утверждать о возможности создания алгебраического аппарата для описания знаний, представляемых в терминах «Узел-Функция-Объект», т.е. для СОМПЗ.

Формализованное описание структурных характеристик организационных знаний

Для формализации СОМПЗ с точки зрения структурных характеристик организационных знаний применена теория графов, так как структурные характеристики СОМПЗ представляют собой узлы УФО-

модели и связи между этими узлами, а любую УФО-модель можно рассматривать как граф.

Далее рассматриваются основные понятия теории графов и их связь с понятиями УФО-подхода и УФО-модели.

Граф (УФО-диаграмма) это множество точек (узлы диаграммы) или вершин и множество линий (потоки диаграммы) или ребер, соединяющих между собой все или часть этих точек. Подграф графа (диаграмма-декомпозиция) это граф, являющийся подмоделью исходного графа, т.е. подграф содержит некоторые вершины исходного графа и некоторые ребра (только те, оба конца которых входят в подграф).

Вершина графа – элемент (точка) графа, обозначающий объект любой природы, входящий во множество объектов, описываемое графом. Отрезок линии, соединяющий вершины графа, называется ветвью графа.

Определим, как с помощью теории графом можно обозначить компоненты УФО-модели и понятия УФО-подхода. Для этого введем основные правила соотношения понятий УФО-подхода и теории графов:

- Узел УФО-элемента – вершина графа.
- Функция УФО-элемента, представляющая собой деятельность, в рамках которой входящие связи преобразуются в выходящие – следующий уровень графа, т.е. подграф.
- Входящие и выходящие связи – дуги графа.

В сравнительном анализе теории графов и УФО-подхода используются ориентированные графы. Соответственно входящие и выходящие связи будут ориентированными ветвями графа, так как любая связь УФО-диаграммы является направленной, т.е. имеет определенные начало и конец.

Для наглядности рассмотрим все эти обозначения на следующем примере (рисунок 2.53).

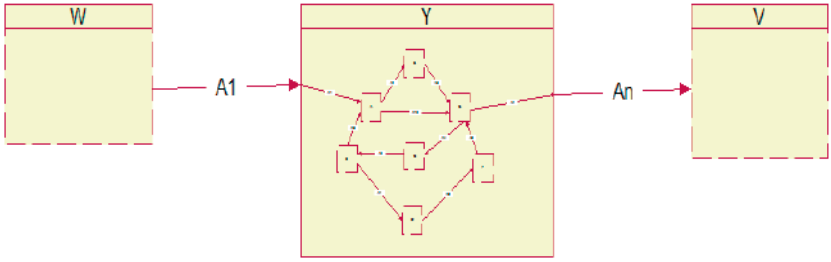


Рис. 2.53. Представление графа в терминах УФО-подхода

Изображенная на рисунке 2.53 УФО-модель представляет собой граф. Обозначим его как **G1(L, M)**, где **L** – множество узлов, **M** – множество связей. Элементы УФО-модели, такие как УФО-элементы **W, Y, V** представляют собой вершины графа. Входящие и выходящие связи A_1 и A_n соответственно будут ориентированными ветвями графа. Тогда компоненты модели на рисунке 2.53 можем представить в следующем формальном виде:

$$L = \{W, Y, V\} \quad (2.92)$$

$$M = \{A_1, A_n\} \quad (2.93)$$

Переходы (связи) модели, в соответствии с теорией графов, имеют следующий вид:

$$A_1 = \{W, Y\}, A_n = \{Y, V\} \quad (2.94)$$

Декомпозиция УФО-элемента **Y** представляет собой подграф графа. Рассмотрим декомпозицию модели на рисунке 2.53 (см. рисунок 2.54):

Обозначим этот граф как **G2(O,P)**, где **O** – множество узлов, **P** - множество переходов. Элементы графа данной УФО-модели такие как **A, B, C, D, E, F, G** представляют собой вершины графа. Входящие и выходящие связи $A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_N$, соответственно, будут ориентированными ветвями графа. По аналогии с предыдущим примером, формально компоненты модели можно представить в следующем виде:

$$O = \{A, B, C, D, E, F, G\} \quad (2.95)$$

$$M = \{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_N\} \quad (2.96)$$

Переходы на модели декомпозиции будут иметь следующее формальное представление: $A_1=\{0, A\}$, $A_2=\{A, B\}$, $A_3=\{B, C\}$, $A_4=\{C, G\}$, $A_5=\{G, E\}$, $A_6=\{E, A\}$, $A_7=\{E, D\}$, $A_8=\{D, F\}$, $A_9=\{F, C\}$, $A_{10}=\{A, C\}$, $A_N=\{C, 0\}$. Можно представить эти две модели как одну и описать ее с помощью понятий теории графов, как единую модель.

Представим модель, показанную на рисунке выше, в понятиях теории графов. Обозначим граф этой модели как **G3(X,Z)**, где **X** – множество узлов, **Z** – множество переходов. Элементы УФО-модели такие как **W, A, B, C, D, E, F, G, V**представляют собой вершины графа. Входящие и выходящие связи $A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_N$, соответственно, будут ориентированными ветвями графа. Таким образом, можем представить данную УФО-модель в следующем виде:

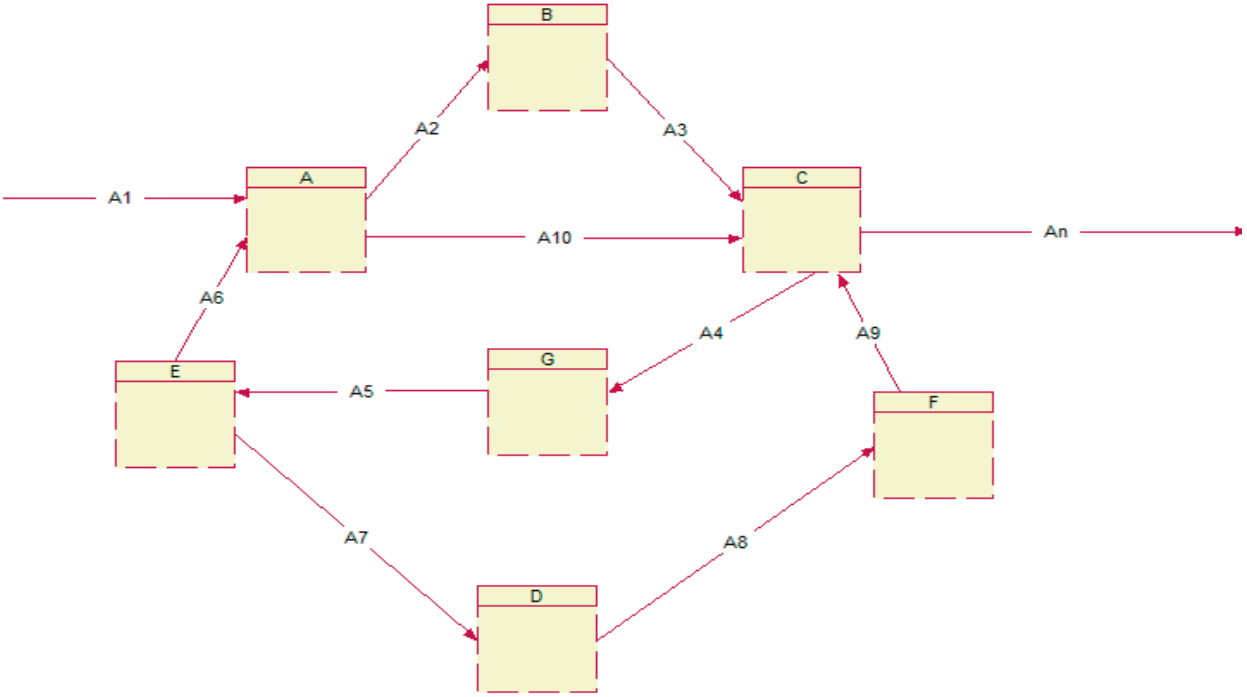


Рис. 2.54. Декомпозиция узла «**Y**»

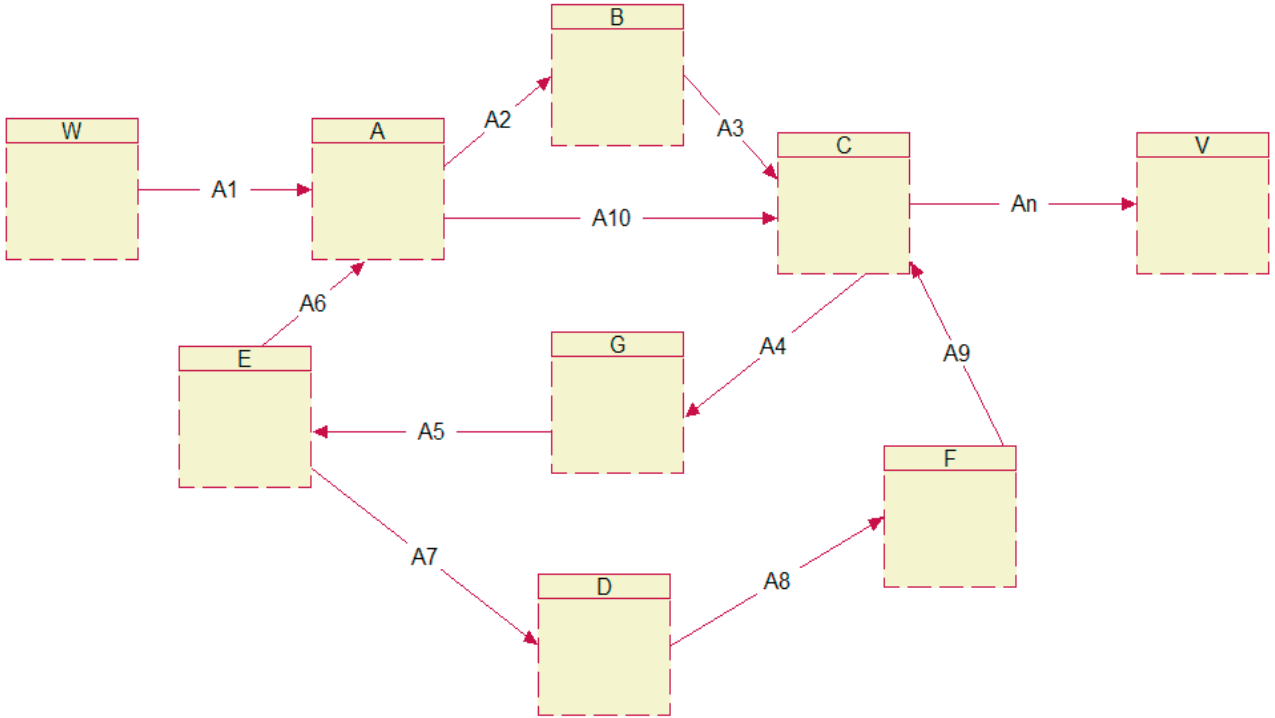


Рис. 2.55. Объединенная модель (2.53 и 2.54)

$$O = \{W, A, B, C, D, E, F, G, V\} \quad (2.97)$$

$$M = \{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_N\} \quad (2.98)$$

Переходы (связи) модели имеют следующий вид: $A_1=\{W, A\}$, $A_2=\{A, B\}$, $A_3=\{B, C\}$, $A_4=\{C, G\}$, $A_5=\{G, E\}$, $A_6=\{E, A\}$, $A_7=\{E, D\}$, $A_8=\{D, F\}$, $A_9=\{F, C\}$, $A_{10}=\{A, C\}$, $A_N=\{C, V\}$. Дуги A_1 и A_N – это соединяющие связи, которые объединяют два уровня модели (в теории графов, два графа – **G1** и **G2**).

Применение теории графов позволяет использовать формальные операций для обработки структурных характеристик организационных знаний. Данные операции формально (в рамках теории графов) представляют собою, соответственно: добавление/удаление ориентированного ребра (связи) и добавление/удаление вершины графа (узла).

Формализованное описание функциональных характеристик организационных знаний

Для формализации СОМПЗ с точки зрения функциональных (процессных) характеристик организационного знания применено исчисление процессов (CCS) Милнера.

Рассмотрим основные понятия данной математической теории. В соответствии с теорией процессов под процессом понимается модель поведения динамической системы на некотором уровне абстракции. Процесс можно представлять себе, как граф **P** (процессный граф), компоненты которого имеют следующий смысл:

- Вершины графа **P** называются состояниями, и изображают ситуации (или классы ситуаций), в которых может находиться моделируемая система в различные моменты своего функционирования.
- Одно из состояний является выделенным, оно называется начальным состоянием процесса **P**.
- Рёбра графа **P** имеют метки, обозначающие действия, которые может исполнять моделируемая система.
- Функционирование процесса **P** описывается переходами по рёбрам графа **P** от одного состояния к другому. Функционирование начинается из начального состояния.

Метка каждого ребра изображает действие процесса, исполняемое при переходе от состояния в начале ребра к состоянию в его конце. Для задания процесса **P**, представляющего собой модель поведения некоторой динамической системы, должно быть указано некоторое множество **Act(P)** действий, которые может выполнять процесс **P**. Предполагается,

что действия всех процессов являются элементами некоторого универсального множества **Act** всех возможных действий, которые может выполнить какой-либо процесс, т.е. для любого процесса **P Act(P) ⊆Act**. Выбор множества **Act(P)** действий процесса **P** зависит от целей моделирования. В разных ситуациях для представления модели анализируемой системы в виде некоторого процесса могут выбираться разные множества действий.

В исчислении процессов множество действий **Act** делят на три следующих класса:

- Входные действия, которые изображаются знакосочетанием вида **α?.** Действие вида **α?** интерпретируется как ввод в процесс некоторого объекта с именем **α.**
- Выходные действия, которые изображаются знакосочетанием вида **α!.** Действие вида **α!** интерпретируется как вывод из процесса некоторого объекта с именем **α.**
- Внутреннее (или невидимое) действие, которое обозначается символом **ατ.**

Внутренним называют такое действие процесса **P**, которое не связано с его взаимодействием с окружающей средой (т.е. с процессами, являющимися внешними по отношению к процессу **P**, и с которыми он может взаимодействовать). Например, внутреннее действие может быть связано с взаимодействием компонентов процесса **P**. В действительности, внутренние действия могут быть самыми разнообразными, но для обозначения всех внутренних действий мы будем использовать один и тот же символ **ατ**. Это отражает желание не различать все внутренние действия, т.к. они не являются «наблюдаемыми» извне процесса **P**.

Так же следует отметить, что объекты, которые вводятся в процесс и выводятся из него, могут иметь самую различную природу (как материальную, так и не материальную). Например, ими могут быть: материальные ресурсы, люди, деньги, информация, энергия и т.д.

Кроме того, сами понятия ввода и вывода могут иметь виртуальный характер, т.е. слова «ввод» и «вывод» могут использоваться лишь как метафоры, а в действительности никакого ввода или вывода какого-либо реального объекта может и не происходить. Говоря неформально, действие процесса **P** рассматривается как:

- входное, если его инициатором является процесс, внешний по отношению к **P**;
- выходное, если оно не является внутренним, и его инициатором является сам процесс **P**.

Исходя из вышесказанного, на языке исчисления процессов, процессом **P** называется тройка вида:

$$\mathbf{P = (S, s^{\circ}, R),} \tag{2.99}$$

компоненты которой имеют следующий смысл:

- **S** – множество, элементы которого называются состояниями процесса **P**.
- $\mathbf{s^{\circ} \in S}$ – некоторое выделенное состояние, называемое начальным состоянием процесса **P**.
- $\mathbf{R \subseteq Act \times S}$ – подмножество вида.
- Элементы множества **R** называются переходами. Если переход из **R** имеет вид (**s1, τ, s2**), то:
- предполагается, что этот переход является переходом из состояния **s1** в состояние **s2** с выполнением действия **τ**;
- состояния **s1** и **s2** называются началом и концом этого перехода соответственно, а действие τ называется меткой этого перехода.

По аналогии с понятием процесса **P** в исчислении процессов введем понятие функции **F** УФО-элемента. Функция **F** есть тройка:

$$\mathbf{(S, S^{\circ}, R),} \tag{2.100}$$

где **S** – множество подпроцессов процесса, соответствующего функции **F**, $\mathbf{S^{\circ} \subset S}$ – множество интерфейсных подпроцессов (причем $\mathbf{S^{\circ} = S? \cup S!}$), **R** – множество переходов в множестве **S**, осуществляемых путем передачи, ввода и вывода объектов:

$$\mathbf{s_i} \xrightarrow{\alpha\tau_{ij}} \mathbf{s_j} \tag{2.101}$$

Иными словами, по аналогии с исчислением процессов рассматривается размеченная система переходов (**S, R**) над множеством потоков **Act(F)**. Элементы множества **Act(F)** потоков (входных **α?**, выходных **α!**, внутренних **ατ**), соответствующего множеству действий в исчислении процессов, также интерпретируются как ввод, вывод или передача объекта с именем потока. При этом, в данном случае (на уровне описания функций) интересны только внутренние потоки, так как внешними (входными и выходными) потоками занимается алгебра изображений теории паттернов и теория графов.

Представленное формальное понимание функции УФО-элемента позволяет использовать понятия исчисления процессов для математического описания функциональных характеристик систем (с точки зрения УФО-подхода). Основы такого описания представлены в таблице 2.14.

Представленные ниже операции префиксного действия и альтернативной композиции, а также другие операции, аналогичные операциям исчисления процессов, могут быть уточнены или расширены с учетом того, что УФО-элемент является, в первую очередь, графическим формализмом. При этом, в соответствии с исчислением функций (в рамках УФО-подхода), любому УФО-элементу соответствует процессный граф, описывающий его функциональность. Таким образом, используя теорию графов, можно описать упомянутые выше операции на более низком уровне как комбинации элементарных операций.

Таблица 2.14

Формальные основы исчисления функций

Исчисление процессов (CCS)	**Исчисление функций (УФО-подход)**
Пустой процесс: NIL = ({s0}, s0, ∅) = 0	Пустая функция: ({s0∈S}, {s0∈S^0}, ∅) = 0
Трасса (протокол) процесса P: последовательность элементов a1, a2, … множества действий Act(P), для которой существует последовательность состояний s0, s1, s2, …такая, что для любого i: si $\xrightarrow{a_{i+1}}$ si+1	Трасса (протокол) функции F: последовательность элементов a1, a2, … множества потоков Act(F) , для которой существует последовательность подпроцессов s0, s1, s2, …такая, что для любого i: si $\xrightarrow{a_{i+1}}$ si+1 (только для ατ).
Префиксное действие: α.P = (S∪{s0'∉S}, s0', R∪{s0',α, s0})	Префиксное действие: s?.F = (S∪{s?∉S}, {s?∈S^0}, R∪{s?, α, {si∈S}}) Постфиксное действие: s!.F = (S∪{s!∉S}, {s!∈S^0}, R∪{{si∈S}, α, s!})
Альтернативная композиция: P1+P2 = (S1∪S2∪{s0'∉S1∪S2}, s0', R1∪R2∪ {s0', α, s1∈R1}∪{s0', α, s2∈R2})	Альтернативная композиция по входу: s?.(F1+F2) = (S1∪S2∪{s?∉S1∪S2}, {s?∈S1?∪S2?}, R1∪R2∪{s?, α1, s1∈S1}∪{s?, α2, s2∈S2}) Альтернативная композиция по выходу: s!.(F1+F2) = (S1∪S2∪{s! ∉S1∪S2}, {s!∈S1!∪S2!}, R1∪R2∪{ s1∈S1, α1 , s!} ∪{ s2∈S2, α2, s!})

В настоящее время в качестве элементарных предлагается рассматривать следующие операции:

- Добавление/удаление связи.
- Добавление/удаление узла.
- Добавление/удаление функции узла.

- Добавление/удаление функционального объекта.

Операции 1–3 исследованы, например, в работе [Зимовец, 2008] и описаны с помощью алгебры изображений теории паттернов. Операция 1 соответствует бинарному оператору (правилу) присоединения образующих/изображений (т.е. узлов). Операция 2 является, в свою очередь, элементарной по отношению к операции 3. Последняя же соответствует процедуре декомпозиции (анализа), текущего узла (как изображения) для представления конфигурации ему соответствующей и описывающей его функциональность. Рассмотрение операций исчисления процессов как комбинаций элементарных операций, представляемых, в свою очередь, средствами алгебры изображений, позволяет конструктивно исследовать и описать взаимодействие двух алгебраических аппаратов (процессов и изображений), тесная связь между которыми отмечена выше. Описание же операции 4, которое, будет использовать средства исчисления объектов, является предметом рассмотрения в следующем подразделе данного учебника.

Приведем примеры использования предложенного исчисления функций, аналогичного исчислению процессов, для формализации знаний, представляемых с помощью УФО-подхода, т.е. средствами СОМПЗ.

Используя обозначения графических формализмов, принятых в формуле префиксного действия (таблица 2.14), результат операции префиксного действия будет выглядеть следующим образом (рис. 2.56):

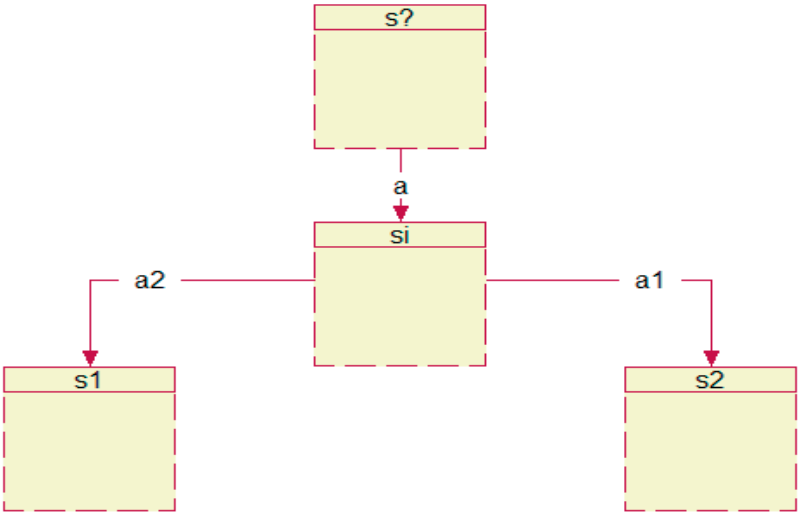


Рис. 2.56. Результат выполнения операции префиксного действия

Как видно из рисунка 2.56, к существующей сети, добавляется новый контекстный узел **s?** а также новый переход от добавленного узла к существующему узлу. Операция постфиксного действия аналогична рассмотренной операции, но, в данном случае добавляется новый контекстный узел, который является исходящим, т.е. соединяет текущую функцию с внешним миром посредством исходящих потоков.

Используя обозначения графических формализмов, принятых в формуле альтернативной композиции по входу (таблица 2.14), результат данной операции примет вид, как показано на рисунке 2.57. Данная операция представляет собою соединение двух подпроцессов в один с добавлением нового контекстного состояния **s?** и двух переходов, соответственно.

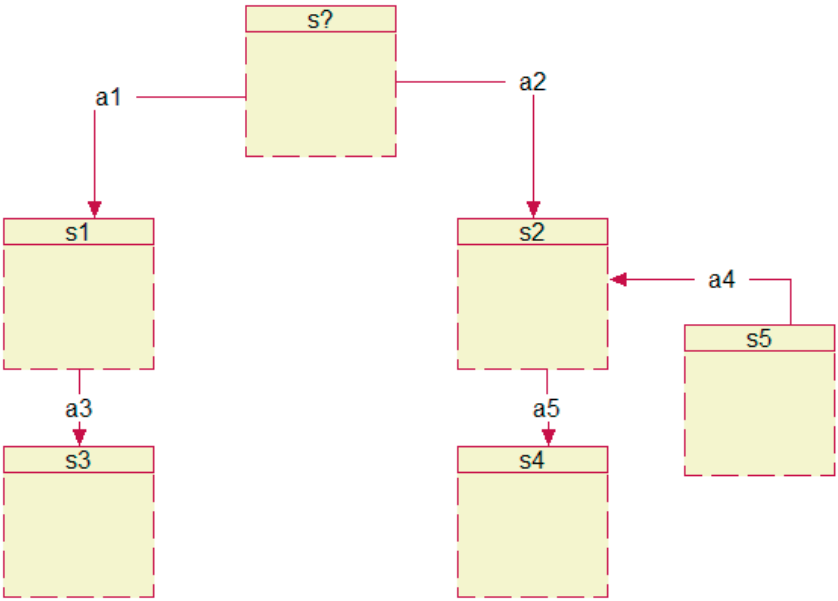


Рис. 2.57. Абстрактный пример результата операции

Операция альтернативной композиции по выходу аналогична рассмотрено, только в данном случае добавляется исходящий контекстный узел.

Далее рассматриваются примеры использования исчисления функций на конкретных предметных областях. Одним из примеров может служить сетевая модель знаний в нотации СОМПЗ, представляющая иерархию понятий, которая изображена на рисунке 2.58.

По аналогии с процессом, в рамках исчисления процессов, сетевую модель иерархии понятий можно рассматривать как введенную выше формально функцию **F**. Трассой данной функции будет конечная последовательность потоков этой функции: $a_1, a_2, a_3, \ldots a_n$, такая, что существует последовательность подпроцессов этой функции: $s_0, s_1, s_2, \ldots s_n$, которая обладает следующими свойствами:

- s_0 соответствует начальному подпроцессу функции, т.е. одному из интерфейсных (входных) подпроцессов из множества S^0;
- для каждого $i \geq 1$ множество **R** содержит переход $s_i \xrightarrow{a_{i+1}} s_{i+1}$.

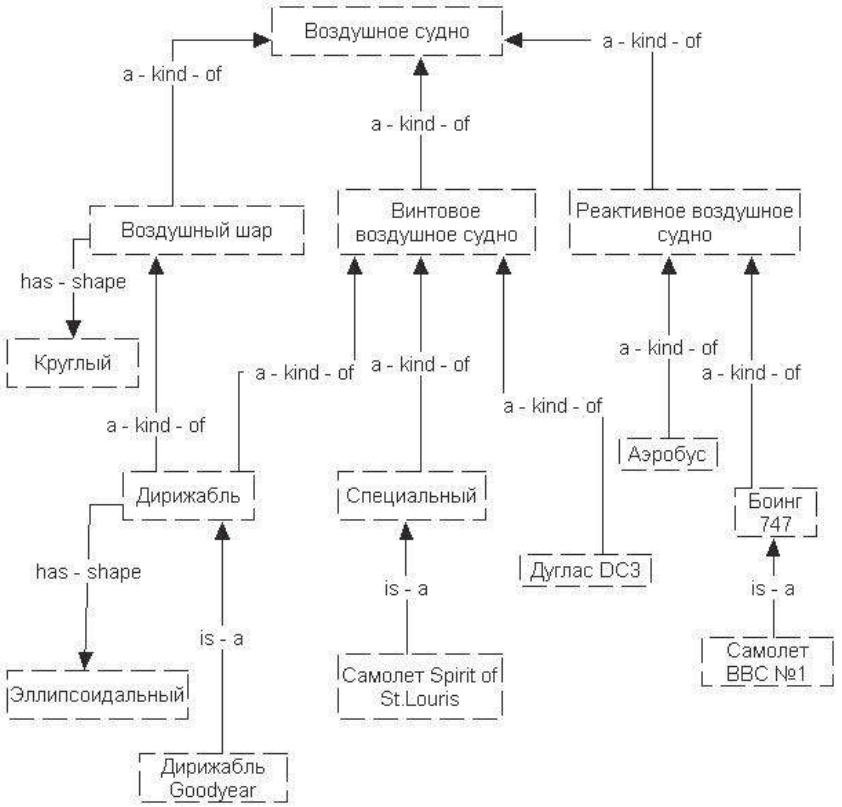


Рис. 2.58. Модель понятий и отношений между ними в терминах СОМПЗ

Заметим, что в данном случае множество всех трасс, является множеством всех возможных логических выводов, т.е., например, трасса:

$$самолетSSL \xrightarrow{is-a} специальный \xrightarrow{a-kind-of} винтовой \xrightarrow{a-kind-of} воздушное\_судно$$

представляет собой заключение о том, что самолет «Spirit of St.Louris» – это объект типа «Специальное винтовое воздушное судно».

Другой пример – фреймовая модель знаний в терминах СОМПЗ, представляющая описание фрагмента расписания занятий в учебном заведении, изображенная на рисунке 2.59.

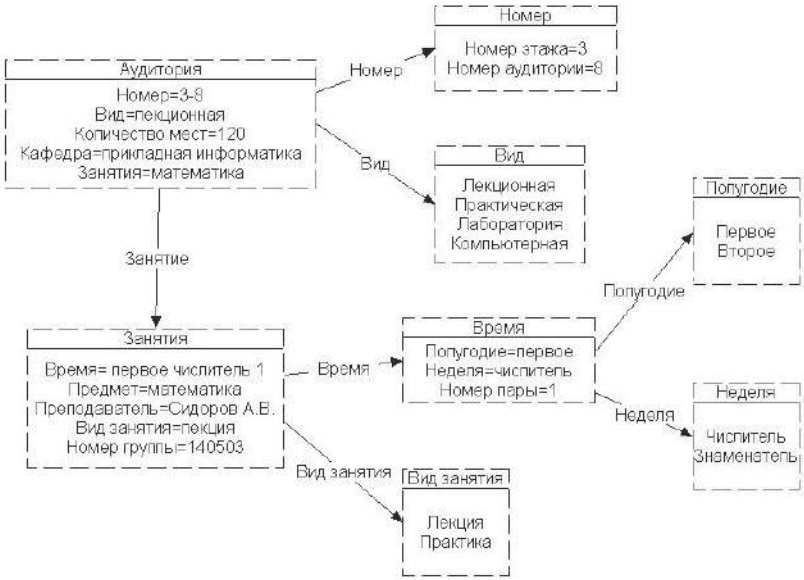


Рис. 2.63. Фрагмент фреймовой сети в терминах СОМПЗ

Фреймовую модель также можно рассматривать в рамках СОМПЗ как введенную выше формально функцию **F**. Интерфейсным (входным) подпроцессом $\mathbf{s_0}$ будет подпроцесс, соответствующий фрейму «Аудитория». Операция «префиксное действие», представляющая собой добавление нового интерфейсного (входного) подпроцесса и внутреннего потока к данной функции (т.е. перехода), будет описывать добавление к модели фрейма и соответствующей связи. Например, если необходимо в рассматриваемой фреймовой модели

учесть описание корпуса некоторого учебного заведения, то это можно сделать путем добавления фрейма «Корпус» со следующими слотами:

- «аудитория»;
- «заведующий»;
- «общая площадь».

В рамках предложенного выше исчисления функций – это добавление есть префиксное действие по отношению к данной функции, в результате которого к множеству подпроцессов добавился входной подпроцесс, соответствующий фрейму «Корпус», а к множеству переходов **R** – новый переход, внутренний поток, который удобно обозначить как «Аудитория». В результате получим фреймовую модель в нотации СОМПЗ, которая приведена на рисунке 2.60.

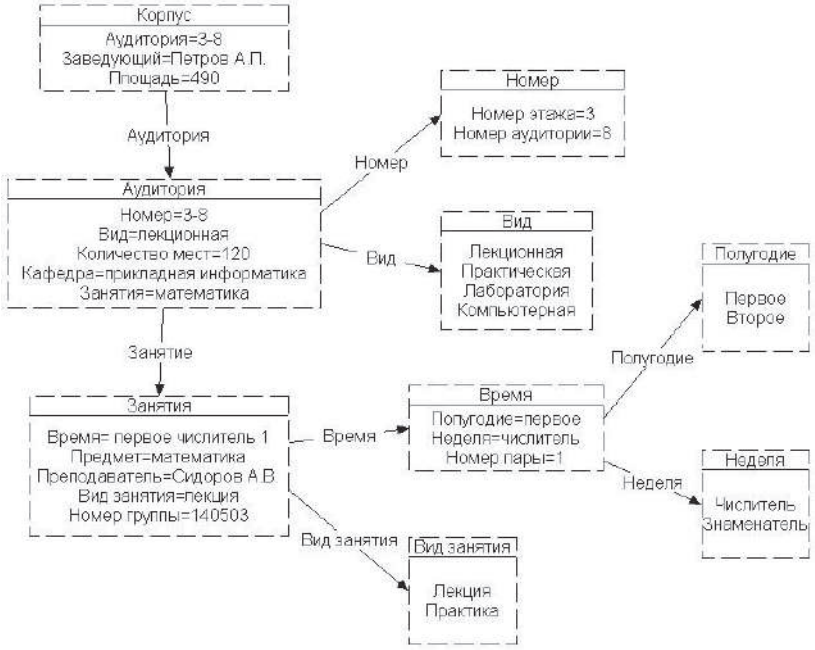


Рис. 2.60. Результат применения операции префиксного действия к фреймовой сети средствами СОМПЗ

Операция альтернативной композиции по входу, согласно исчислению процессов, представляет собой объединение двух процессов в один с добавлением нового начального состояния и двух переходов, соответственно. Рассмотрим пример применения данной операции для двух фрагментов семантической сети, показанных на рисунке 2.61.

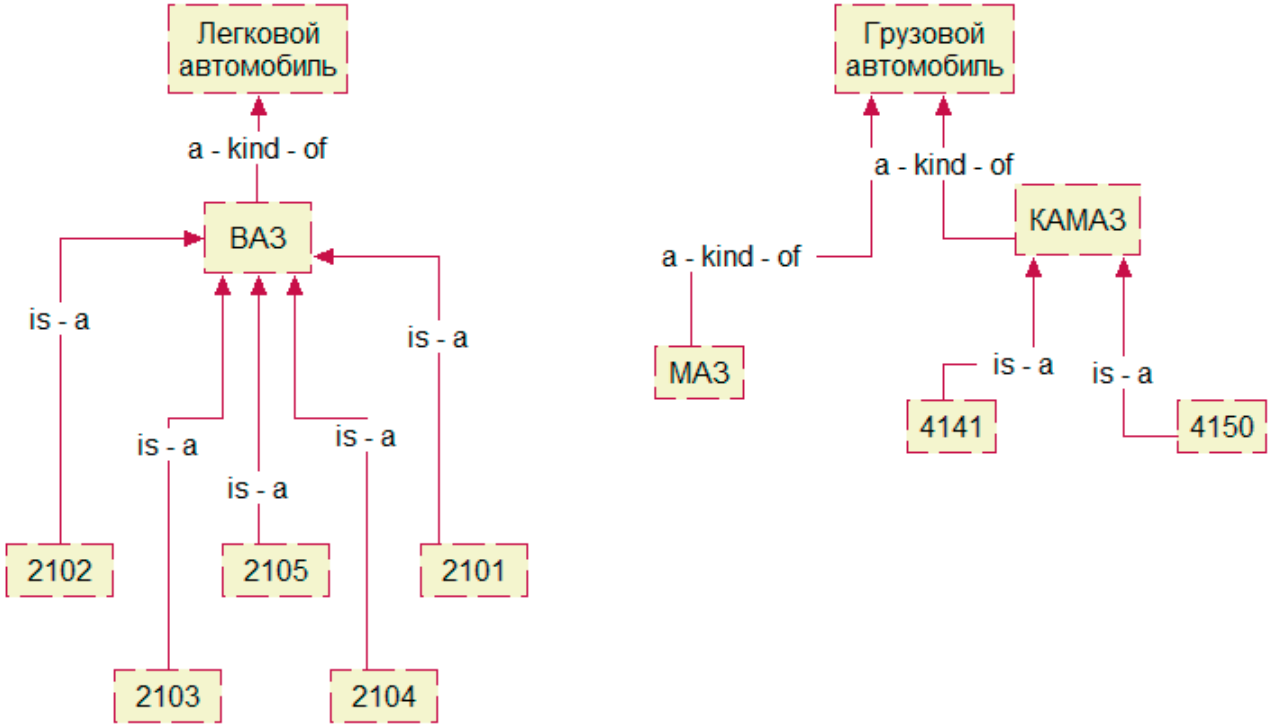


Рис. 2.61. Фрагменты семантических сетей, описывающие средства передвижения

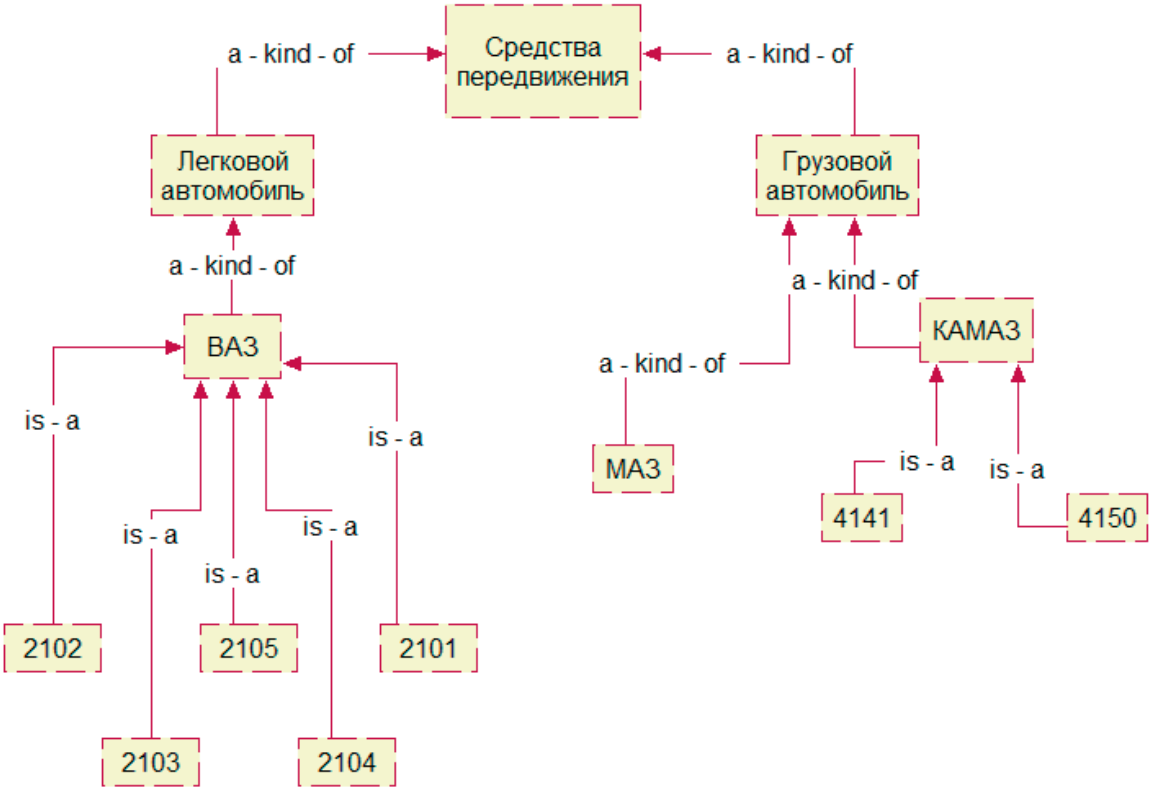


Рис. 2.62. Результат операции альтернативной композиции по входу

Для того, чтобы из представленных отдельных фрагментов семантической сети получит одну семантическую сеть, применима операция альтернативной композиции, которая объединяет два представленных фрагмента семантической сети в один с добавлением нового входящего контекстного узла сети, например, «средства передвижения». Результат применения рассматриваемой операции показан на рисунке 2.62.

Исходя из графического примера операции альтернативной композиции, а так же на основании описания данной операции в теории процессов, можем говорить о том, что данная операция обладает свойствами ассоциативности, т.е. верно выражение $P_1+(P_2+P_3)=(P_1+P_2)+P_3$; так же данная операция является коммутативной, т.е. верно выражение $P_1+P_2=P_2+P_1$. Подробно свойства операции альтернативной композиции рассматриваются в работе.

Итак, с помощью исчисления процессов (т.е. введенного здесь исчисления функций) можно формально описывать функциональные характеристики систем, рассматриваемых в рамках системного графоаналитического подхода «Узел-Функция-Объект», а также организационные знания, моделируемые с помощью СОМПЗ. Таким образом, представленные результаты исследований позволяют утверждать, что с помощью системно-объектного графоаналитического подхода «Узел-Функция-Объект» (т.е. с помощью СОМПЗ) можно хранить и полноценно использовать функциональные характеристики организационного знания.

Показанная возможность формального описания УФО-моделей (моделей, создаваемых средствами СОМПЗ) позволяет говорить о возможности формализованного описания знаний, их преобразования из одного вида в другой, а также возможности создания единого универсального механизма вывода.

Формализованное описание объектных характеристик организационных знаний

Для формализации СОМПЗ с точки зрения объектных характеристик организационных знаний применено исчисление объектов Абади/Карделли. Для чего, в первую очередь, усовершенствовано алгебраическое представление системы в виде УФО-элемента путем интеграции алгебраических средств исчисления объектов и исчисления процессов. При этом в исчисление объектов введен особый класс объектов **G** со специальными полями и методами.

В исчислении объектов абстрактный объект представляет собой набор полей и методов. Использование метода объекта – это вызов метода, изменение метода – это переопределение. Поле – частный

случай метода (константный метод). Изменение значения поля является частным случаем переопределения метода. Методы выполняются в контексте некоторого объекта (имеют ссылку на объект). Таким образом, любой абстрактный объект «o» формально в исчислении объектов представляется в следующем виде:

$$o = [l_i = b_{i,i\in 1..n}, l_j = \sigma(x_j)b_{j,j\in 1..m}], \quad (2.102)$$

где l_i представляют собой поля объекта, в которых записаны характеристики объекта o; l_j – методы данного объекта, в которых в скобках указаны их аргументы, а за скобками результаты их работы; $o \in O$, $b_i \in O$, $b_j \in O$ (O – множество термов исчисления объектов).

Вычисление в исчислении объектов – это последовательность вызовов и переопределения методов, для чего определены правила редукции. Для нас наибольший интерес представляет правило вызова следующего метода (вызов метода l_j объекта o):

$$o.l_j \to b_j\{x_j \mapsto o\} \quad (2.103)$$

Далее в работе средства исчисления объектов применяются для формального описания УФО-элемента с точки зрения его узлового объекта. Например, рассмотрим УФО-элемент, объект которого занимает функциональный узел, имеющий входные потоки, обозначенные как $a?_i$, и выходные потоки, обозначенные как $a!_j$ (см. рис. 2.63).

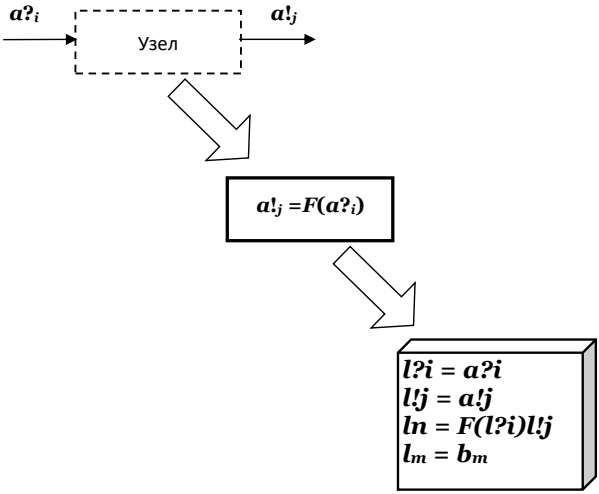


Рис. 2.63. УФО-элемент с точки зрения исчисления объектов

Данные обозначения в стиле теории процессов подчеркивают тот факт, что потоки состоят из объектов, которые по ним (потокам) передаются из одного узла (объекта) в другой. Таким образом, объект **G** рассматриваемого УФО-элемента представляет собой сущность, которая за счет своей функциональности преобразует поток входных объектов $\boldsymbol{a}^{?}_{i}$ в поток выходных объектов $\boldsymbol{a}^{!}_{j}$.

В соответствии с упомянутыми выше правилами исчисления объектов данный узловой объект **G** (и, следовательно, соответствующий УФО-элемент) формально может быть представлен в виде следующего выражения:

$$\boldsymbol{G} = \left[\boldsymbol{l}^{?}_{i} = \boldsymbol{a}^{?}_{i};\ \boldsymbol{l}^{!}_{j} = \boldsymbol{a}^{!}_{j};\ \boldsymbol{l}_{n} = \boldsymbol{F}(\boldsymbol{l}^{?}_{i})\boldsymbol{l}^{!}_{j};\ \boldsymbol{l}_{\mathrm{m}} = \boldsymbol{b}_{m}\right], \qquad (2.103)$$

где:

$\boldsymbol{l}^{?}_{i}$ – поле узлового объекта (может представлять собой набор или множество), которое содержит значение входных потоковых объектов $\boldsymbol{a}^{?}_{i}$ и, соответственно, имеет такой же тип данных;

$\boldsymbol{l}^{!}_{j}$ - поле узлового объекта (может представлять собой набор или множество) которое содержит значения выходных потоковых объектов $\boldsymbol{a}^{!}_{i}$ и имеет такой же тип данных;

$\boldsymbol{l}_{n}$ – метод узлового объекта (может представлять собой набор или множество), преобразующий входные потоковые объекты узла в выходные.

$\boldsymbol{l}_{m}$ – поле узлового объекта (может представлять собой набор или множество), которое содержит основные характеристики данного объекта $\boldsymbol{b}_{m}$.

Данный объект **G** представляет собой специальный класс объектов в исчислении объектов, так как содержит специально выделенные поля и методы.

В соответствии с правилами исчисления объектов каждый метод объекта имеет один аргумент и один результат (см. выше 2.101), т.е. количество аргументов и результатов совпадает. В системном же моделировании, когда моделируются реальные процессы и объекты, такого может и не быть. Возможны следующие варианты. Результатов в реальности больше, чем аргументов, тогда для обеспечения соответствия формальному описанию некоторые аргументы дублируются, что, собственно, будет соответствовать реальному положению вещей. Результатов в реальности меньше, чем аргументов, тогда для обеспечения соответствия формальному описанию необходимо вводить нужное количество пустых результатов.

Нетрудно видеть, с одной стороны, что представленное формальное описание с точки зрения объекта УФО-элемента учитывает и структурную, и процессную, и субстанциальную его характеристики. И, таким образом, может рассматриваться как **новый способ формального описания систем как УФО-элементов**. Действительно, во-первых, имена полей $l?_i$ и $l!_j$ могут рассматриваться как имена потоков, связывающих УФО-элемент с другими элементами, т.е. как структурная характеристика данного УФО-элемента. Во-вторых, метод $l_n = F(l?_i)l!_j$ представляет собой процессную (функциональную) характеристику данного УФО-элемента. В-третьих, поле $l_m = b_m$, а также значения входных и выходных потоков (потоковых объектов) $a?_i$ и $a!_j$ представляют собой объектную характеристику данного УФО-элемента.

С другой стороны, представленное формальное описание УФО-элемента учитывает описательные возможности упомянутого выше алгебраического аппарата теории паттернов. Средствами этой теории УФО-элемент описывается как образующая, т.е. как графический формализм, представляющий собой именованный объект со связями, который характеризуется признаком $\boldsymbol{\alpha}$ и показателями входных и выходных связей $\boldsymbol{\beta}$. Средствами исчисления объектов это учитывается именем поля l_m (которое может рассматриваться как имя объекта), значением данного поля b_m (которое может рассматриваться как признак $\boldsymbol{\alpha}$) и значениями входных и выходных потоков $a?_i$ и $a!_j$ (которые могут рассматриваться как показатели входных и выходных связей $\boldsymbol{\beta}$).

Кроме того, использование исчисления объектов для формализации УФО-элемента позволяет интегрировать в его формальное описание возможности исчисления процессов (CCS). Дело в том, что функциональные характеристики УФО-элемента формально могут быть описаны по аналогии с CCS, что обосновано в предыдущем пункте данного исследования путем введения исчисления функций. Таким образом, в соответствии с этим обоснованием метод l_n узлового объекта **G** (тело этого метода) формально можно представить средствами исчисления функций в виде:

$$F = (S, S^0, a\tau), \tag{2.105}$$

где S – множество подпроцессов процесса, соответствующего данному методу; $S^o \in S$ – множество интерфейсных (входных **S?** и выходных **S!**) подпроцессов (причем $S^o = S? \cup S!$**;** в число входных связей множества подпроцессов **S?** входит множество связей $l?_i$, в число выходных связей множества подпроцессов **S!** входит множество связей $l!_j$); $a\tau$ –

множество потоков в **S**, осуществляющих передачу объектов глубинного яруса связанных подпроцессов:

$$(s_i, \alpha\tau_{ij}, s_j) \tag{2.106}$$

При этом, если УФО-элемент (узловой объект) рассматривается без учета декомпозиции (на контекстном уровне), то выражение в скобках принимает вид, соответствующий так называемой «нулевой функции»:

$$F^0 = (\{s^0 \in S\}, \{s^0 \in S^0\}, \emptyset) = s^0 \tag{2.107}$$

Это означает, что в первом случае метод узлового объекта (УФО-элемента) может быть представлен, в том числе, в виде УФО-диаграммы, а во втором случае только в виде формулы или алгоритма. Введенное в предыдущем пункте исчисление функций позволяет использовать для методов исчисления объектов сформулированные там же по аналогии с CCS операции над функциями: префиксное действие, постфиксное действие и альтернативная композиция.

В рамках УФО-подхода процесс анализа и синтеза организационной системы начинается с построения иерархии связей, использующихся в дальнейшей работе. Иерархия связей представляет все ресурсы, отходы, результаты производства, документы и т.п., которые участвуют в моделируемой организационной системе. После анализа и построения иерархии связей системы, начинается разработка УФО-диаграмм, на которых связи выступают в роли материальных и (или) информационных потоков, передающих соответствующие объекты от одного узла (УФО-элемента) к другому узлу (УФО-элементу). Сказанное позволяет ввести в терминологию УФО-подхода понятие «потоковый объект», которое дополняет существующее понятие об объекте, реализующем функциональный узел в рамках УФО-элемента. Т.е., далее рассматриваются два вида объектов: **узловой объект** в рамках УФО-элемента (будем далее обозначать заглавными латинскими буквами) и **потоковый объект** в рамках потока\связи (будем далее обозначать строчными латинскими буквами).

Средствами исчисления объектов можно формально описать не только узловые объекты, но и потоковые. Потоковый объект в рамках потока\связи можно представить как объект, обладающий только набором полей, содержащих основные характеристики объекта, т.е. методы и входы, выходы объекта в данном случае не учитываются. Такой объект (также представляющий собой еще один специальный класс объектов) формально средствами исчисления объектов представляется с помощью следующего выражения:

$$a_i = [l_j = b_j], \quad (2.108)$$

где: a_i – потоковый объект с именем **a**; $l_j = b_j$ – поля потокового объекта с некоторыми значениями b_j.

4.3.5. Метод вывода на системно-объектных графоаналитических моделях организационных знаний, представляемых средствами СОМПЗ

Весь процесс построения модели знаний в терминах СОМПЗ можно поделить на несколько этапов:

- Построение иерархии потоковых объектов (аналогично процедуре построения иерархии связей в УФО-анализе);
- Разработка моделей организационно-деловых и производственно-технологических процессов (аналогично построению УФО-модели);
- Описание конечных методов узловых объектов с применением скриптового языка;
- Использование модели (осуществление логического вывода на модели).

Графически алгоритм моделирования знаний с помощью СОМПЗ представлен на рисунке 2.64.

Имея иерархию потоковых объектов с необходимыми характеристиками, можно приступить к разработке модели организационно-деловых и производственно-технологических процессов, в которых перерабатываются потоковые объекты. Для этого необходимо выделить узлы организационной системы (перекрестки потоковых объектов) и, таким образом, выделить узловые объекты модели. Узловыми объектами могут быть, например, люди, которые реализуют функционал узла модели, так же в роли узловых объектов может выступать оборудование или различные технологические комплексы. Далее необходимо описать конечные методы узловых объектов. Т.е., если метод объекта не содержит узлов нижнего уровня, тогда для него записываются конкретные математические или логические действия над входными объектами с использованием скриптового языка, иначе узел подвергается декомпозиции и так до тех пор, пока не будет достигнут необходимый уровень декомпозиции модели.

После построения модели и описания методов узловых объектов, модель готова к использованию. Использование модели заключается в запуске механизма логического вывода. Данный механизм подробно рассмотрен в следующем подразделе.

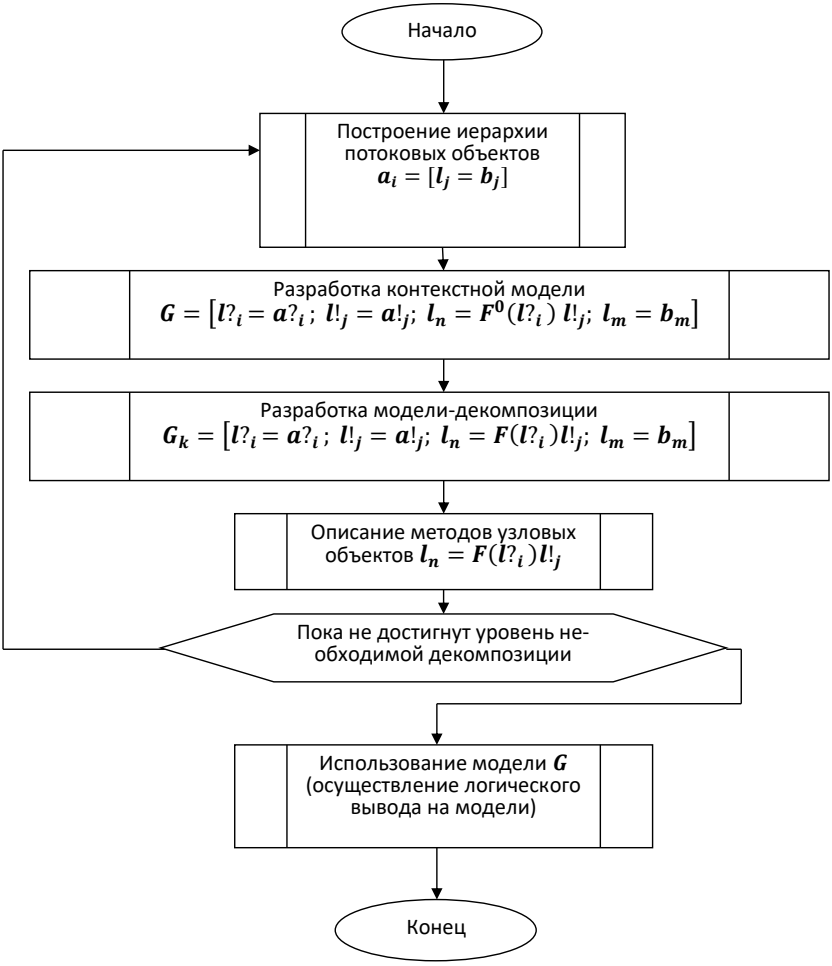


Рис. 2.64. Алгоритм описания знаний с помощью СОМПЗ

Если для хранения и обработки организационных знаний представлять их в виде УФО-элементов, то, с учетом формального их описания средствами исчисления объектов, манипулирование этими знаниями, в частности вывод на них, можно обеспечить путем организации цепочки вызовов методов узловых объектов со стороны соответствующих потоковых объектов. Цепочка организуется на уровне декомпозиции УФО-элемента, метод объекта которого не принимает вид «нулевой функции» (см. выше). В нашем случае вызов метода узлового объекта формально записывается следующим образом:

$$G.l_n \rightarrow l!_j \{l?_i \mapsto G\} \quad (2.109)$$

Подобный вызов метода (например, метода l_n) узлового объекта (например, объекта G) будет иметь место в том случае, если на вход узлового объекта поступает поток, наименование объектов которого (потоковых) совпадает со значением поля узлового объекта, которое содержит значение входных потоковых объектов (например, поля $l?_i$).

Старт процедуры логического вывода в модели знаний, представленных средствами СОМПЗ, осуществляется путем инициализации некоторого контекстного потокового объекта соответствующей УФО-модели, которому задаются конкретные значения его характеристик. После этого значение контекстного потокового объекта попадает в соответствующее входное поле интерфейсного узлового объекта, который содержится в узле, для которого контекстные потоковые объекты являются входящими. Вызов метода узлового объекта происходит в случае, если входящие потоковые объекты содержат конкретные значения, либо, когда вызов осуществляется явно из тела другого метода.

После выполнения действий над значениями входящих потоковых объектов вызывается следующий метод узлового объекта и так, пока не достигается конец модели. Формально с учетом описания УФО-элементов средствами исчисления объектов упомянутая процедура вывода может быть представлена следующим образом:

$$a_i=[l_m=b_m]: a_i= a?_i = l?_i \rightarrow \rightarrow a_{i+1}=[l_{m+1}=b_{m+1}]: a_{i+1} = a?_{i+1} =$$
$$l?_{i+1} \rightarrow G_{k+1}.l_{n+1} \rightarrow l!_{j+1}\{l?_{i+1} \mid\!\rightarrow G_{k+1}\} \rightarrow a_{i+2}=[l_{m+2}=b_{m+2}]: a_{i+2} = a?_{i+2} =$$
$$l?_{i+2} \rightarrow G_{k+2}.l_{n+2} \rightarrow l!_{j+2}\{l?_{i+2} \mid\!\rightarrow G_{k+2}\} \rightarrow a_{i+3}=[l_{m+3}=b_{m+3}]: \quad ... \quad (2.110)$$

Графически, выражение 2.109, представляет собою последовательность УФО-элементов с соответствующими потоками, как показано на рисунке 2.65.

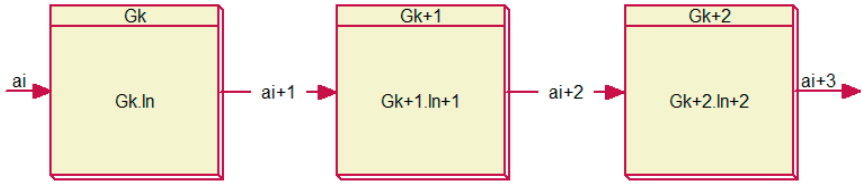


Рис. 2.65. Графическое представление механизма логического вывода в СОМПЗ

Организация цепочки вызовов методов узловых объектов со стороны потоковых объектов аналогична определению трассы (протокола) функции, описанной выше, по аналогии с трассой (протоколом)

процесса в CCS. Если $l?_i^{i\in 1..n} = a_i^{i\in 1..n}$, то можно организовать следующую цепочку вызовов в рамках некоторого УФО-элемента, для которого $a_i^{i\in 1..n} \in a\tau$:

$$\xrightarrow{a_i} F(l?_i)l!_j \xrightarrow{a_{i+1}} F(l?_{i+1})l!_{j+1} \xrightarrow{a_{i+2}} F(l?_{i+2})l!_{j+2} \xrightarrow{a_{i+3}} \dots \quad (2.111)$$

Представленные выражения можно рассматривать как формальное описание механизма вывода знаний в случае их представления в виде УФО-элементов с учетом формального их описания средствами исчисления объектов и исчисления процессов, т.е. как описание механизма вывода в рамках СОМПЗ. На рисунке 2.66 представлена блок-схема механизма логического вывода на моделях СОМПЗ.

Математически и программно механизм логического вывода на таких моделях (согласно правилам исчисления объектов) будет заключаться в последовательном вызове методов узловых объектов. Причем вызов первого метода узлового объекта, стоящего на пути контекстного потокового объекта осуществляется при инициализации полей узлового объекта, которые соответствуют входящему потоковому объекту. Далее работа механизма логического вывода регулируется методами узловых объектов. Т.е. каждый i-й метод, выполнив свои действия, вызывает следующий метод потокового объекта и так до тех пор, пока не закончатся узлы узловые объекты. После выполнения всех методов узловых объектов модели, генерируется новая модель для рассматриваемого конкретного случая, которая эквивалентна последовательности вызванных методов (протоколу функции).

4.4. Моделирование финансовых систем

Рассматривая применения системно-объектного метода «Узел-Функция-Объект» нельзя обойти вопросы моделирования финансовых систем, поскольку финансовые системы являются важнейшими подсистемами большинства экономических и производственных систем.

Для построения модели финансовой системы необходимо осуществить решение трёх взаимосвязанных задач: идентификации, формализации и реализации. Проблема идентификации состоит в том, что финансовые системы существуют как композиционные составляющие более крупных и сложных систем.

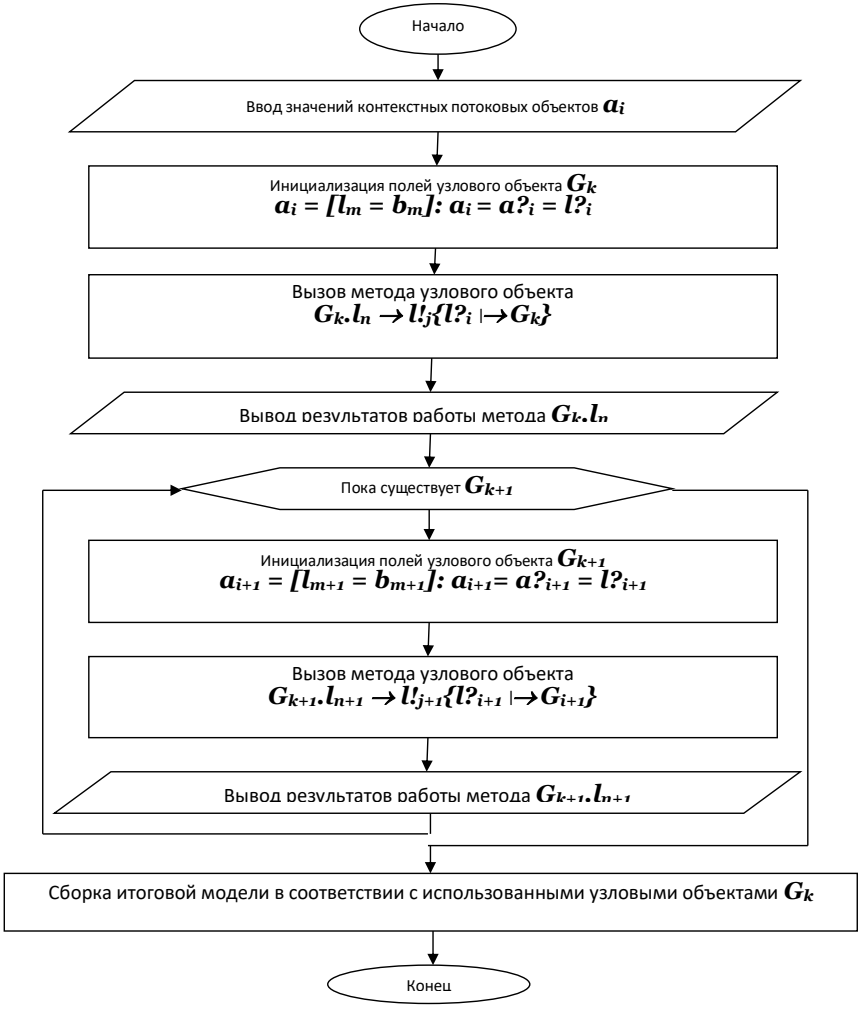


Рис. 2.66. Схема механизма логического вывода на моделях СОМПЗ

Например, система финансирования инвестиционного проекта является частью производственной экономической системы, состоящей из инвесторов, подрядчиков, других физических и юридических лиц, между которыми формируются организационные, производственные, финансовые и иные связи. Абстрагирование от всех видов связей и выделение только связей, обусловленных финансирова-

нием, а также агентов, поддерживающих эти связи, является необходимым условием построения адекватной модели финансирования инвестиционного проекта. Проблема формализации заключается в том, что построение модели основано на декларативных и процедурных знаниях о финансовой системе, существующих в виде вербальных или слабо формализованных моделей (таблицы, графика и т.п.). Для построения модели нужно согласовать между собой знания, инкапсулированные в подобных моделях, и представить их в формализованном виде. Проблема реализации модели это – проблема воплощения в ней трёх различных аспектов моделирования: визуализации элементов и связей системы, аналитической обработки инкапсулированных в модели знаний о системе и имитации функционирования финансовой системы.

Рассматриваемый далее метод построения компьютерных моделей финансовых систем предлагает следующее решение указанных проблем: идентификации – на основе процессного подхода и монетарного представления бизнес-процессов; формализации – на основе специализации системно-объектного метода «Узел-Функция-Объект»; реализации – на основе трёхкомпонентной архитектуры и объектно-ориентированном проектировании.

В контексте УФО метода для представления структуры произвольной системы используется теория паттернов Гренандера, для описания функций узлов – исчисление процессов Милнера, а для представления объектов – исчисление объектов Абади-Кардели. При моделировании финансовых систем с помощью специализированного варианта УФО – ДВ-УФО метода процессы идентификации, формализации и реализации значительно упрощаются, что делает метод весьма эффективным и практичным [Маторин, 2019; Тубольцев, 2018; Тубольцева, 2014].

На уровне абстрагирования, принятом в финансовой математике, под финансовой системой будем понимать совокупность системных объектов – репозиториев, которые представляют собой абстракцию мест временного хранения денежных средств и/или других высоколиквидных финансовых активов, а также событий перемещения этих активов между репозиториями. События перемещения финансовых активов – транши в настоящее время можно рассматривать как мгновенные события передачи информации, абстрагируясь от физического перемещения финансовых активов, которое может осуществляться в течение некоторого промежутка времени.

Следует сразу подчеркнуть, что, хотя понятие транша (финансового события) в контексте финансовой математики и ДВ-УФО метода практически идентичны, понятия финансового потока значительно различаются. Это связано с тем, что в финансовой математике финансовый поток является атрибутом (характеристикой) репозитория, называемого владельцем потока. А в ДВ-УФО модели финансовый поток сам характеризуется двумя узлами: источником и приёмником траншей финансового потока. Поэтому, необходимо не только уточнить терминологию, но и внести ряд изменений в стандартные формулы, приспособив их для балансировки финансовых потоков в узле ДВ-УФО диаграммы.

Методы финансовой математики всегда были ориентированы на изучение финансовых операций, которые являются абстрактными описаниями реальных финансовых инструментов. Большинство финансовых инструментов (кредиты, займы, депозиты, учёт векселей и др.) порождают финансовые потоки между двумя репозиториями (узлами, в терминах УФО метода). Любой из этих репозиториев можно назвать владельцем финансовой операции и рассматривать только его, вместе со всеми финансовыми событиями (траншами), которые составляют финансовую операцию.

При таком подходе, во-первых, отпадает необходимость в постоянных ссылках на репозитории, поскольку они фиксируются в самой постановке задачи анализа финансового инструмента; что сокращает описание траншей до формата «**Д**еньги, **В**ремя» (ДВ формат), который используется и в финансовой математике, и в ДВ-УФО методе. Во-вторых, все транши ассоциируются только с одним репозиторием, поэтому входящим в репозиторий траншам (траншам из входящих в репозиторий финансовых потоков) приписываются положительные значения в позиции «Деньги» поскольку они увеличивают баланс узла (репозитория), а выходящим траншам (траншам из выходящих финансовых потоков) – отрицательные значения, поскольку они баланс репозитория уменьшают.

Поэтому под финансовым событием в финансовой математике принято понимать упорядоченную пару (x, d), где x – денежная сумма, поступающая на счёт владельца финансового потока или снимаемая со счёта (положительные значения x соответствуют поступлению средств, отрицательные – их выбытию), а d – момент времени, в который это происходит. Под финансовым потоком в финансовой математике понимается множество всех траншей, ассоциированных

с конкретным юридическим или физическим лицом {(x, d)}. Количество траншей всегда конечно, поэтому математически более корректное обозначение финансового потока выглядит так: $\{(x_i,d_i)\}$, где $i=1,...,K$, а K – количество траншей в финансовом потоке. Обычно предполагается, что нумерация событий соответствует последовательности их появления: более позднему событию соответствует больший номер.

При таком понимании траншей и финансовых потоков термин «финансовый поток» становится синонимом термина «финансовая операция». Для задач, решаемых финансовой математикой это не приводит к проблемам в интерпретации результатов, поскольку в финансовой математике абстрагируются от всего, кроме самих результатов вычислений.

На рис. 2.67 показана операция кредитования в том виде, как это принято в финансовой математике. В некоторый момент времени t_1 кредитор предоставляет заёмщику кредит в размере К, а в последующий момент времени t_2 заёмщик возвращает кредитору сумму кредита с процентами в размере S.

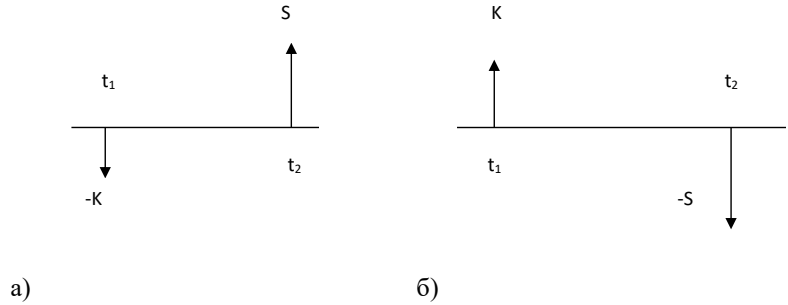


Рис. 2.68. Финансовый поток операции кредитования:
а) с точки зрения кредитора; б) с точки зрения заёмщика

На рис. 2.68 представлена операция кредитования с использованием графической нотации ДВ-УФО метода. В отличие от графической нотации, используемой в финансовой математике, в графической нотации ДВ-УФО метода структура финансовой системы имеет единственное представление, что облегчает понимание и интерпретацию ДВ-УФО диаграмм. Отметим, что нотация, используемая в финансовой математике, неприменима, если в финансовой системе более двух репозиториев, поскольку в этом случае нельзя приписать знак денежным суммам, перемещаемым траншами. Например, нельзя в том же

виде, как на рис. 2.67, представить операцию форфейтинга, поскольку в ней участвуют три актора: покупатель, продавец и коммерческий банк. Если использовать графическую нотацию ДВ-УФО метода, то затруднений не возникает.

Таким образом, можно утверждать, что формализованное представление структуры финансовой системы с использованием графической нотации ДВ-УФО метода является не менее информативным, чем это требуется для решения любых задач финансовой математики. Это важно, поскольку аналитический потенциал ДВ-УФО метода целиком определяется алгоритмами финансовой математики.

Поскольку ДВ-УФО метод ориентирован на использование далеко за пределами финансовой математики, для соответствия стандартам ИСО 9001 необходимо в дополнение к репозиториям и финансовым потокам добавить граничные узлы двух типов: спонсоров и абсорберов. Через спонсоров в финансовую систему могут поступать извне финансовые активы, а через абсорберы – покидать её. Спонсоры и абсорберы в отличие от репозиториев являются не анализируемыми сущностями. Считается, что они гарантированно выполнят свои финансовые обязательства перед репозиториями.

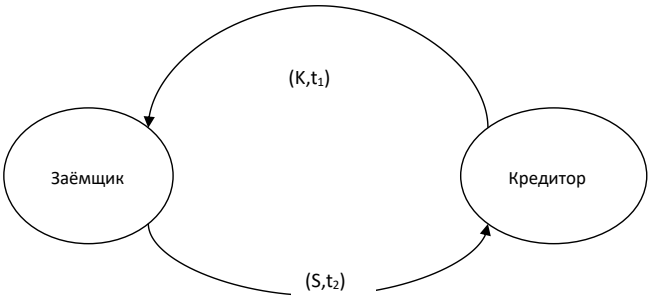


Рис. 2.68. Финансовые потоки операции кредитования в графической нотации ДВ-УФО метода

Отметим следующее существенное обстоятельство. Поскольку перемещения финансовых активом являются на всех уровнях иерархии финансовой системы единственным типом потоков, то транши становятся единственным видом объектов глубинного яруса, которыми обмениваются объекты верхнего яруса: спонсоры, репозитории и абсорберы. Таким образом, для описания структуры финансовой системы

достаточно графической нотации с четырьмя терминальными знаками. На рис. 2.69 приведён возможный вариант такой нотации.

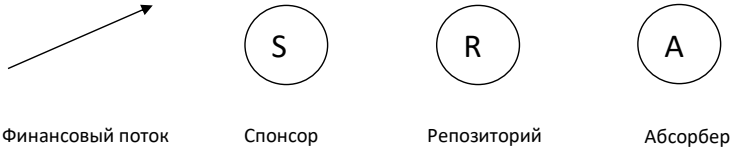


Рис. 2.69. Возможный вариант терминальных знаков ДВ-УФО метода

В принципе, единственным требованием к терминальным знакам является их хорошая различимость и узнаваемость. Практика ДВ-УФО моделирования показала, что пиктограммы не имеют преимуществ перед буквенными обозначениями с добавлением нумерации.

Основными числовыми характеристиками финансового потока (а, следовательно, и самой финансовой операции) являются чистое приведённое значение (**NPV**, **N**et **P**resent **V**alue), а также – уровень внутренней доходности (**IRR**, **I**nternal **R**ate of **R**eturn). Финансовая операция считается выгодной при заданной ставке сравнения (обычно это доходность наилучшего альтернативного вложения средств), если NPV положительно. Значение NPV вычисляется по формуле:

$$NPV\left(\left\{\left(x_i,d_i\right)\right\}_{i=1}^{K},d_0,r\right)=\sum_{i=1}^{K}\frac{x_i}{\left(1+r\right)^{d_i-d_0}}, \qquad (2.112)$$

где r – ставка сравнения, d_0 – момент дисконтирования. Значение NPV как функция трёх аргументов зависит от финансового потока, момента дисконтирования и ставки сравнения.

Сравнение инвестиционных проектов с помощью NPV может быть затруднено, поскольку они могут начинаться в различные моменты времени и иметь разную продолжительность, что влияет на величину NPV. Более удобной оценкой эффективности инвестирования в финансовую операцию или проект является уровень внутренней доходности (IRR), определяемый как ставка сравнения, при которой NPV = 0. Данный показатель более удобен, поскольку он (IRR) от момента дисконтирования не зависит. При сравнении инвестиционных проектов выбирается тот, IRR которого больше.

В ДВ-УФО моделях применяется следующий подход: на визуальном уровне в ДВ-УФО диаграммах показываются только сами финансовые потоки, а составляющие финансовый поток транши представлены на уровне аналитических процедур в том формате, который

расширяет формат представления траншей, принятый в финансовой математике. При этом, значения в позиции «Деньги» для любого транша всегда положительны, поскольку в ДВ-УФО моделях нет необходимости с помощью знака числа отражать факт поступления или выбытия денежных средств в репозиторий. Это и так понятно, поскольку каждый финансовый поток в ДВ-УФО модели «знает» откуда и куда он перемещает денежные средства.

Таким образом, если способ графического представления финансового потока, принятый в финансовой математике, ориентирован на визуализацию одного конкретного потока, то ДВ-УФО метод позволяет сделать это сразу для множества финансовых потоков; но, при этом, теряется возможность визуализации отдельных траншей.

Из сказанного следует, что используемый в ДВ-УФО методе формат представления транша должен содержать данные, достаточные для ответа на следующий набор вопросов: «***Кто***? ***Кому***? ***Когда***? ***Сколько***? ***Основание***?». По сравнению с финансовой математикой добавлена новая позиция «Основание», поскольку ДВ-УФО модель системы финансирования проекта инкапсулирует данные сразу о множестве бизнес процессов, поэтому бизнес-процессы (а, следовательно, и финансовые потоки, соответствующие им) необходимо каким-то образом различать. Простейшим способом сделать это является включение в описание каждого финансового события основания (причины) его возникновения. Транши, имеющие общее основание, составляют отдельный финансовый поток.

На основании сказанного, можно определить формат представления траншей в ДВ-УФО модели (предполагая реализацию в объектно-ориентированном стиле программирования) следующим образом:

- поле **s** (source, источник) содержит ответ на вопрос «Кто?» и должно быть ссылкой на узел ДВ-УФО диаграммы, который генерирует транш;
- поле **r** (receiver, приёмник) содержит ответ на вопрос «Кому?» и должно быть ссылкой на узел ДВ-УФО диаграммы, который принимает транш;
- поле **w** (when, когда) содержит ответ на вопрос «Когда?» и должно быть значением хронологической даты;
- поле **m** (money, деньги) содержит ответ на вопрос «Сколько?» и должно быть значением целого или дробного числового типа;
- поле ***f*** (flow, поток) содержит ответ на вопрос «Основание?» и должно быть ссылкой на финансовый поток ДВ-УФО диаграммы,

который содержит транш (ассоциирован с траншем), либо строковой константой – маркёром финансового потока.

В таком представлении транша (наборе полей) содержится вся необходимая для целей анализа системы финансирования проекта информация. Понятие финансового события (транша), используемое в ДВ-УФО моделях, является небольшим расширением понятия транша, применяемого в финансовой математике. Расширение осуществляется за счёт дополнительного поля ***f*** (***f****low*, поток) в объекте финансового события, реализованного на аналитическом уровне ДВ-УФО модели, связывающего этот объект с ассоциированным финансовым потоком, принадлежащий визуальному уровню ДВ-УФО модели.

Введём несколько обозначений. Множество всех траншей в ДВ-УФО модели будем обозначать прописной латинской буквой *T*. Выражение $t \in T$ есть факт того, что *t* – некоторый транш; а через *t.s, t.r, t.w, t.m, t.f* обозначим поля транша *t*. Через *N* (***N****ode*, узел) – обозначим множество узлов ДВ-УФО модели, а через N_R, N_A, N_S – множества узлов, являющихся репозиториями, абсорберами и спонсорами соответственно. В этих обозначениях запись $a \in N_R$ говорит о том, что объект с именем a является репозиторием. Для любого узла $n \in N$ через $In(n)$ обозначим множество входящих в узел траншей, а через $Out(n)$ – множество траншей, выходящих из него. Очевидно, что $Out(n \in N_A) = \emptyset$ – пустое множество траншей, поскольку абсорбер не имеет выходящих финансовых потоков; а для спонсоров нет финансовых потоков, входящих в него: $In(n \in N_S) = \emptyset$.

После спецификации формата представления финансовых событий появляется возможность определения функции узла (в контексте УФО моделирования) и алгоритма балансировки, являющегося основой аналитических процедур. В УФО методе термин «Функция» употребляется в широком смысле как назначение (роль) системного объекта, а для формализованного описания функции узла требуется использовать исчисление процессов Милнера. В контексте ДВ-УФО моделирования задача определения функции узла и алгоритма балансировки сильно упрощается за счёт использования языка финансовой математики.

Функция узла показывает в разрезе конкретного узла соотношение между желаемым финансовым результатом (*In(n)*) и требуемыми для этого финансовыми вложениями (*Out(n)*). Алгоритм вычисления функции узла состоит из последовательности следующих действий (шагов):

1. для данного узла n определяются два пустых множества *In(n)* и *Out(n)*;

2. из множества *T* (множества всех траншей) извлекается транш *t*;

3. если *t.r=n*, то *t* добавляется в *In(n)*, если *t.s=n*, то *t* добавляется в *Out(n)*;

4. если *T* не пусто – переход к шагу 2, иначе – конец алгоритма.

Алгоритм балансировки финансовых потоков осуществляется только для репозиториев, поскольку только репозитории имеют как входящие, так и выходящие финансовые потоки и для них имеет смысл говорить о балансе потоков. Поэтому, репозитории должны иметь поле начального баланса *n.b* и метод вычисления баланса в заданный момент времени *d n.B(d)*. Временной аспект в ДВ-УФО моделировании ранее почти не рассматривался, поэтому необходимо ввести некоторые обозначения, позволяющие отразить данный аспект.

Для произвольного узла *n* определим значение функции $d_0(n)$ следующим образом: $d_0(n) = \min t.w, t \in In(n) \cup Out(n)$, т.е. как наиболее раннюю дату генерации или приёма транша этим узлом. Аналогично определим для узла *n* функцию $d_1(n) = \max t.w, t \in In(n) \cup Out(n)$ как наиболее позднюю дату генерации или приёма транша этим узлом. Тогда можно период работы системы финансирования проекта весьма точно задать как временной сегмент [d_{min}, d_{max}], где $d_{\min} = \min d_0(n), n \in N$, а $d_{\max} = d_1(n), n \in N$. Вне временного сегмента [d_{min}, d_{max}] система финансирования не генерирует и не обрабатывает никаких траншей.

Алгоритм вычисления баланса некоторого репозитория $n \in N_R$ на момент календарной даты d не сложен и состоит из следующей последовательности действий (шагов):

1. целочисленной переменой *l* присваиваем *l = n.b* – начальное значение баланса денежных средств в репозитории;

2. вычисляем функцию узла $F_n=F(n)\equiv<In(n), Out(n)>$;

3. для множества *In(n)* вычисляем сумму полей *t.m* для всех $t \in In(n) : t.w \le d$ и прибавляем к *l*, получая новое значение баланса;

4. для множества *Out(n)* вычисляем сумму полей *t.m* для всех $t \in Out(n) : t.w \le d$ и вычитаем из *l*, получая новое значение баланса;

5. возвращаем значение $n.B(d)\equiv l$. Конец алгоритма.

Завершая рассмотрение знаковой системы аналитического уровня ДВ-УФО модели, отметим, что она представляет собой язык (финансовой) математики, адаптированный к потребностям ДВ-УФО

моделирования. Формулы для вычисления уровня внутренней доходности (IRR) операции в финансовой математике и формула для вычисления доходности репозитория в ДВ-УФО модели несколько отличаются друг от друга тем, что в финансовой математике у финансовой операции начальный баланс всегда равен нулю, а при анализе репозиториев так считать нельзя.

В частности, чистое приведенное значение финансового потока в репозитории $n \in N_R$ вычисляется по формуле:

$$NPV \equiv NPV(F_n, d, r) = \frac{n.b}{(1+r)^{d_{\min}-d}} + \sum_{t \in In(n)} \frac{t.m}{(1+r)^{t.w-d}} - \sum_{t \in Out(n)} \frac{t.m}{(1+r)^{t.w-d}}, \qquad (2.113)$$

где, как и в формуле 2.112, d – дата, на которую осуществляется дисконтирование, а r – ставка сравнения (дисконтирования). Формулу 2.113 можно рассматривать как способ балансировки дисконтированных потоков в репозитории, которая необходима для вычисления уровня внутренней доходности репозитория, но, при этом, требуется учитывать и начальный баланс репозитория. Фактически, начальный баланс интерпретируется как дополнительный входящий в репозиторий транш, поступающий в момент начала финансирования проекта.

На аналитическом уровне ДВ-УФО модели определяется функция узла и алгоритм балансировки финансовых потоков, на основе которых осуществляется анализ финансовой системы. Кроме этого, на аналитическом уровне инкапсулируются декларативные и процедурные знания о предметной области, представленные, главным образом, в виде базовых аналитических процедур.

В ДВ-УФО методе определяется несколько локальных (для отдельного узла) и глобальных (для всей финансовой системы) аналитических процедур. Глобальные аналитические процедуры: валидация финансовой системы; вычисление доходности финансовой системы, а также локальные аналитические процедуры: вычисление доходности узла (репозитория); минимизация начального баланса узла.

Аналитические процедуры базовой редакции ДВ-УФО метода определены, исходя из того, что ДВ-УФО модели финансовых систем на практике будут использоваться на этапе инстанцирования инвестиционного или иного проекта в ходе переговорного процесса по формированию экономической производственной системы, которая будет осуществлять проект. Таким образом, ДВ-УФО моделирование есть инструмент поддержки принятия решений так называемой фазы 0.

Инвесторы или другие потенциальные участники проекта определённо задаются в ходе переговорного процесса следующими вопросами. Осуществим ли проект? Насколько проект доходен? Каков конкретно доход того или иного участника? Каких затрат потребует участие в проекте?

В базовой редакции ДВ-УФО метода определены аналитические процедуры, отвечающие на перечисленные базовые (инвариантные к области применения) вопросы. Отметим, что ответы на базовые вопросы должны быть получены на основе информации, генерируемой участниками проекта в ходе переговорного процесса. В ДВ-УФО модели эта информация инкапсулируется в структурной диаграмме, множестве траншей и начальных балансах.

Первый вопрос наиболее важный, поскольку с ним связаны основные риски для всех участников проекта. Аналитическая процедура валидации финансовой системы проекта показывает насколько участники в состоянии выполнять свои обязательства по финансированию проекта.

Для того чтобы реализовать валидацию, необходимо формализовать это понятие. На множестве всех траншей *T* из ДВ-УФО модели некоторой системы финансирования проекта определим предикат *V(T)* следующим образом: *V(T)* равно истина (true), если система финансирования выполнима, и ложь (false) – в противном случае. Алгоритм валидации системы финансирования проекта, т.е. вычисления предиката *V(T)* состоит из последовательности следующих действий (шагов):

1. из множества *T* извлекается транш *t* с минимальным значением поля *t.w*;
2. для транша *t* определяется источник *s=t.s*;
3. поле баланса *s.b* уменьшаем на величину поля *t*.m;
4. если *s.b<0*, то – конец алгоритма, возвращаем *V(T)=false;*
5. для транша *t* определяется приёмник *r=t.r*;
6. поле баланса *r.b* увеличиваем на величину поля *t*.m;
7. если множество *T* не пусто, то переход к шагу 1;
8. конец алгоритма, возвращаем *V(T)=true*.

На рис. 2.70 представлена блок-схема алгоритма валидации. Шаги алгоритма достаточно понятны; главным является шаг 4, который в случае, если *s.b<0*, прерывает валидацию, поскольку обнаруживает участника проекта, не имеющего возможности выполнить свои обязательства в рамках представленной в ДВ-УФО модели системы финансирования проекта.

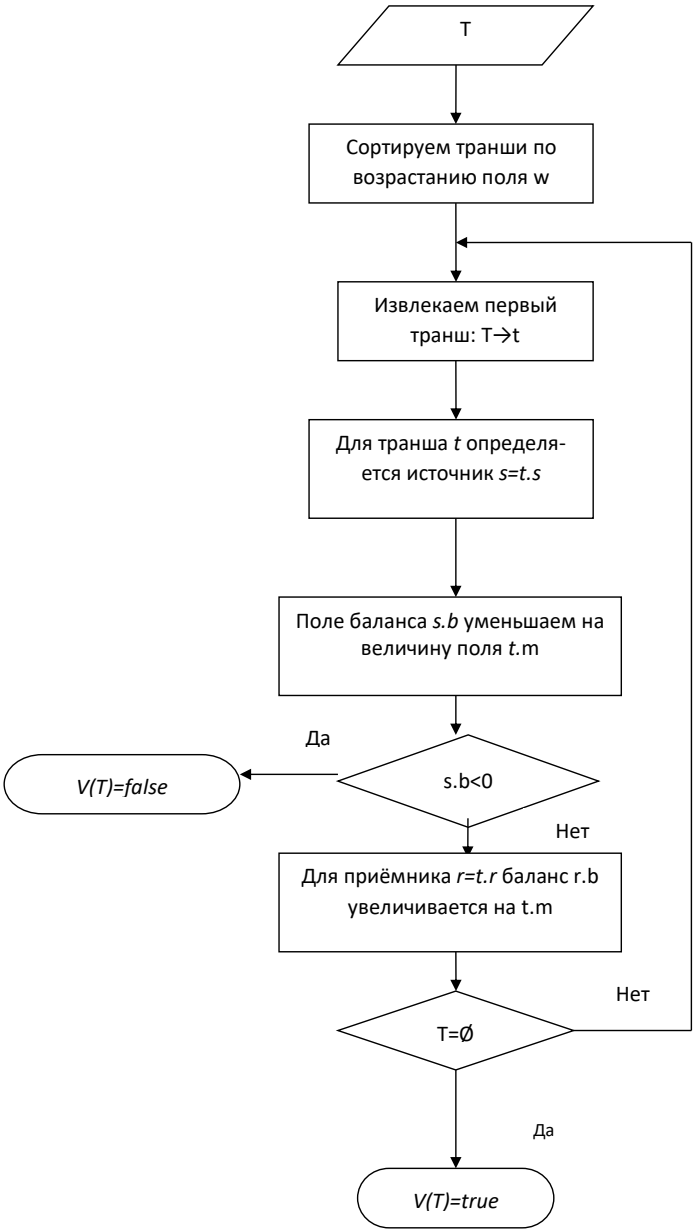


Рис. 2.70. Блок-схема алгоритма валидации

Валидная система финансирования проекта имеет право на существование, поскольку может быть реализована. Однако это ещё не означает, что она действительно будет реализовываться, поскольку не ясно насколько она эффективна. Все участники финансирования проекта заинтересованы в получении наибольшей отдачи от вложенных в проект средств, которая прямо зависит от доходности проекта. Поэтому второй по значимости задачей анализа системы финансирования проекта является определение доходности проекта в целом. Задача вычисления доходности проекта в целом является более простой с вычислительной точки зрения задачей, чем аналогичная задача для отдельного продюсера. Это связано с тем, что большинство траншей будут существовать внутри системы финансирования проекта, перемещая между репозиториями денежные средства, не влияя, при этом, на общий баланс средств в системе финансирования. На доходность всей системы финансирования проекта оказывают влияние только ресурсные потоки, связанные с граничными элементами, и начальные балансы продюсеров. Все эти параметры легко вычислимы.

Если обозначить продолжительность работы системы финансирования проекта в годах через H, а $B_н$ и $B_к$ начальный и конечный балансы, то общая доходность i проекта (в процентах годовых) может быть вычислена по формуле эффективного процента:

$$i = \left(\frac{B_к}{B_н}\right)^{\frac{1}{H}} - 1 = \left(1 + \frac{S - A}{B_н}\right)^{\frac{1}{H}} - 1. \qquad (2.113)$$

Выражение в скобках при $S>A$ будет больше 1, а $i>0$. Таким образом, если потребители продукции проекта вложат в систему финансирования проекта больше средств, чем выведут поставщики, то проект будет прибыльным, и его доходность будет равна i (при переводе в проценты умножить на 100).

После рассмотрения ключевых вопросов финансирования проекта в целом, перейдём к анализу отдельных элементов. Это также важно, поскольку каждый участник финансирования проекта имеет собственные интересы, связанные с участием в проекте, и ДВ-УФО модель должна в наглядной форме представлять результаты такого участия. В базовой редакции ДВ-УФО метода представлены следующие локальные аналитические процедуры: минимизация репозитория, определение доходности репозитория.

Процедура минимизации репозитория является важной для повышения эффективности использования средств. Если валидация системы финансирования проекта прошла успешно, то это означает наличие в любом репозитории средств, достаточных для выполнения своих обязательств перед другими участниками проекта. Говоря более конкретно – в любой момент времени репозиторий имеет на балансе больше средств, чем потребуется при ближайшей по времени генерации и отправке транша.

Может случиться так, что часть средств из начального баланса репозитория совсем не будет использована в процессе финансирования проекта. Безусловно, это повышает надёжность и устойчивость системы финансирования, однако одновременно снижается доходность на вложенный в проект денежный ресурс. Поэтому целесообразно знать: какой минимальный объём средств необходимо иметь в начале на балансе репозитория, достаточный для реализации проекта. Эту задачу решает процедура минимизации репозитория.

Алгоритм минимизации репозитория $n \in N_p$ состоит из следующей последовательности действий (шагов):

1. вычисляется минимум функции n.B(d) на сегменте [dmin, dmax] и присваивается целой переменной L: $L = \min n.B(d),\ d \in [d_{\min}, d_{\max}]$;
2. начальный баланс репозитория n.b уменьшается на L. Конец алгоритма.

Блок-схема алгоритма минимизации репозитория представлена на рис. 2.71. Очевидно, что все сложности алгоритма «спрятаны» в функции текущего баланса продюсера n.B(d) и алгоритм вычисления минимума этой функции на сегменте [dmin, dmax]. Сегмент [dmin, dmax] это – время работы системы финансирования проекта, а функция n.B(d) кусочно-постоянная на этом сегменте, поэтому найти минимальное значение функции n.B(d) не сложно.

Процедура минимизации репозитория является локальной потому, что минимизация всех репозиториев сразу может сделать систему финансирования проекта неустойчивой к сбоям в генерации траншей, что на практике вполне возможно. Вопрос минимизации репозитория может решаться только в индивидуальном порядке, поскольку для некоторых репозиториев минимизация начального баланса важна, а для других – важнее сохранить устойчивость (гарантированную способность выполнять обязательства).

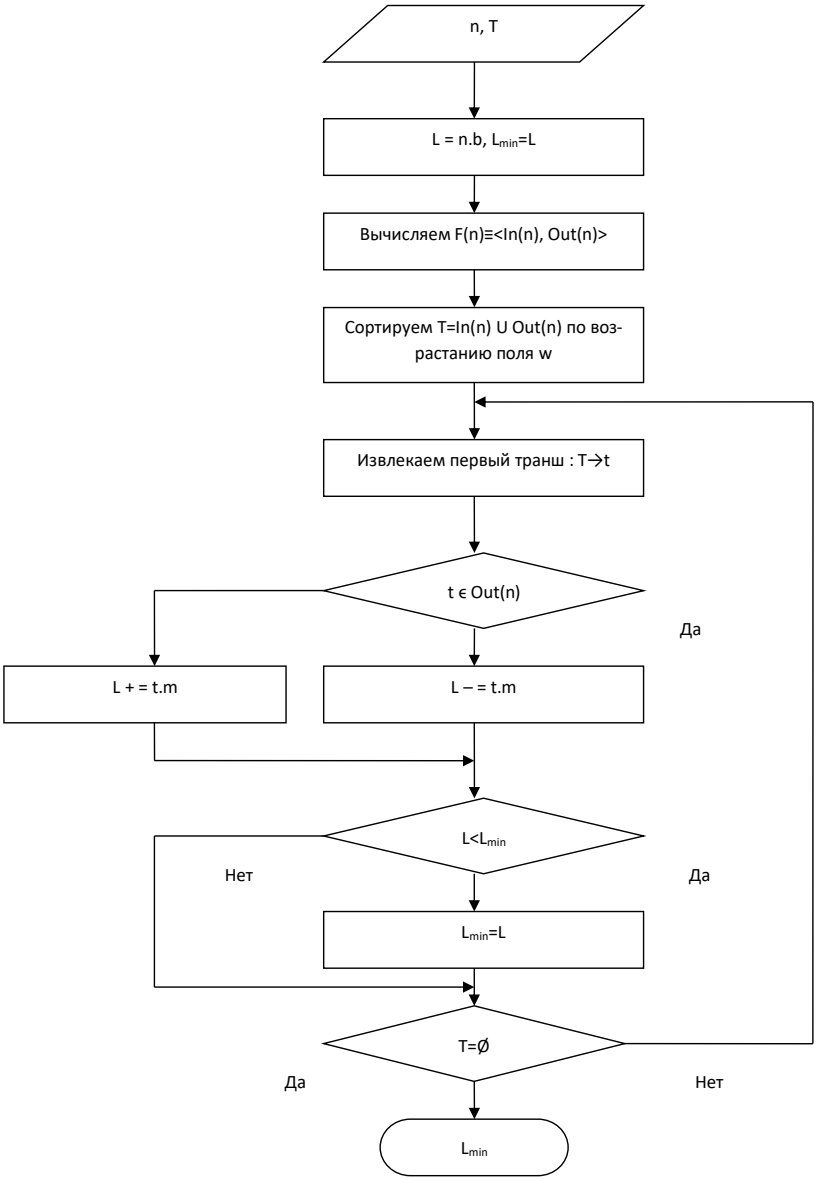


Рис. 2.71. Блок-схема алгоритма минимизации репозитория

Для вычисления доходности репозитория $n \in N_P$ необходимо решить относительно неизвестной ставки сравнения r уравнение:

$$\frac{n.b}{(1+r)^{d_{\min}-d}} + \sum_{t \in In(n)} \frac{t.m}{(1+r)^{t.w-d}} - \sum_{t \in Out(n)} \frac{t.m}{(1+r)^{t.w-d}} = 0, \qquad (2.114)$$

которое представляет собой условие баланса дисконтированных потоков, входящих и выходящих из репозитория.

Рассмотренные аналитические процедуры составляют аналитический базис ДВ-УФО метода в его базовой редакции. Они применимы к любым финансовым системам, поскольку опираются на самые общие и универсальные методы финансовой математики. Очевидно, что ими не исчерпываются все возможности анализа в конкретных прикладных областях, но более специальные методики анализа систем финансирования являются дополнением, а не заменой для них.

Как уже отмечалось, ДВ-УФО моделирование систем финансирования проектов (финансовых систем) имеет практический смысл только в контексте компьютерных информационных технологий. В первую очередь это связано с необходимостью проведения моделирования в реальном масштабе времени в рамках переговорного процесса. Наличие компьютерного инструмента поддержки ДВ-УФО моделирования делает возможным участникам переговорного процесса осуществлять моделирование самостоятельно, поскольку инструментарий «берёт на себя» все трудности моделирования, делая процесс моделирования столь же простым, как набор текста. Поскольку в переговорах участвуют лица принимающие решения, для них важно, что анализ системы финансирования проекта разрабатывается ими самостоятельно, так как это повышает их доверие к результатам анализа, что, в целом, должно способствовать выработке эффективных решений.

Проблема реализации ДВ-УФО модели представляет собой задачу инкапсуляции трёх различных аспектов моделирования: визуализации данных, их аналитической обработки и имитации функционирования финансовой системы. Аспект визуализации включает в себя предоставление интерфейса для сбора декларативных знаний о финансовой системе, упаковку этих декларативных знаний в формат, удобный для последующей обработки и анализа, а также представление результата анализа в удобной для инвесторов форме. Аналитический аспект должен включать в себя привычные для инвесторов аналитические процедуры (такие, как NPV и IRR анализ), дополнив их принципиально новыми методиками анализа на основе современных

информационных технологий. Имитационный аспект целиком основан на компьютерных информационных технологиях, и предоставляет возможность валидации ДВ-УФО модели, а также постановки задач оптимизации как отдельных элементов, так и финансовой системы в целом.

Оригинальный УФО метод, для идентификации системных связей использует ресурсное представление процессов. Специализированный ДВ-УФО метод использует монетарное описание бизнес процессов. Основное ограничение состоит в том, что монетарное представление имеет не любой процесс, а только такой процесс (в дальнейшем – бизнес процесс), входы и выходы которого допускают монетарную оценку (можно оценить в денежном выражении). Анализ бизнес процессов позволяет абстрагироваться от всех аспектов, не связанных с финансами, и чётко определить элементы финансовой системы и существующие между ними финансовые связи, что позволяет решить задачу идентификации. При этом также фиксируются финансовые события и финансовые потоки, необходимые для целей формализации и анализа.

На рис. 2.72 представлена архитектура компьютерной (цифровой) модели финансовой системы.

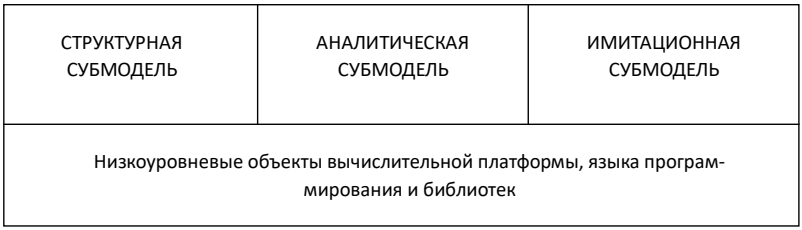


Рис. 2.72. Архитектура цифровой модели системы финансирования

Цифровая модель системы финансирования имеет три субмодели, которые являются относительно независимыми, каждая решает свою специфическую задачу и использует знаковую систему, ориентированную на решение этой специфической задачи. Кроме этих субмоделей на рис. 2.72 обозначены низкоуровневые бинарные объекты, предназначенные для поддержки высокоуровневых объектов финансовой системы и её субмоделей.

Выделенные субмодели решают базовые задачи идентификации, формализации и реализации, а также ряд вспомогательных задач.

Рассмотрим каждую субмодель в контексте решаемых ими задач.

Структурная субмодель содержит описание элементов финансовой системы и сложившихся между ними связей. Задача идентификации решается на основе монетарного представления бизнес процессов с помощью специальной графической знаковой системы. Формализация декларативных знаний о финансовой системе осуществляется с помощью специальной графической нотации. Реализация структурной субмодели ориентирована на поддержание адекватности модели в целом. В частности, недопустима визуализация финансового потока, которого не может быть в реальности (например, идущего из абсорбера).

Аналитическая субмодель решает задачу формализации декларативных знаний о финансовой системе, поступивших в ДВ-УФО модель через интерфейс структурной субмодели. Формализация осуществляется с помощью знаковой системы финансовой математики, адаптированной к реализации в стиле объектно-ориентированного программирования. Фактически, все декларативные знания о финансовой системе инкапсулированы в множестве T всех траншей и начальных балансах репозиториев. Реализация аналитической субмодели также максимально поддерживать адекватность модели, обеспечивая согласованность со структурной субмоделью. Это означает отсутствие траншей источник которых отсутствует в структурной субмодели и т.п.

Имитационная субмодель в базовой редакции ДВ-УФО метода содержит единственный имитационный эксперимент, проверяющий валидность финансовой системы. Реализация имитационной субмодели сильно зависит от вычислительной платформы и языка программирования, но должна содержать контроллеры элементов структурной субмодели и машинный код алгоритмов аналитических процедур.

Завершая обзор ДВ-УФО метода, отметим наиболее перспективные области применения. Во-первых, как уже отмечалось, ДВ-УФО метод вполне пригоден для использования в финансовых вычислениях, поскольку он обладает способностью визуального представления практически любых задач финансовой математики, а ограниченность вычислительных возможностей базовой редакции ДВ-УФО метода в расширенных редакциях легко преодолевается. Во-вторых, эскизная модель любого инвестиционного проекта легко реализуется как ДВ-УФО модель некоторой финансовой системы. При этом, нет необходимости в расширении базовой редакции ДВ-УФО метода и разработки специального инструментария для компьютера. В-третьих, базовая редакция ДВ-УФО метода вполне может использоваться для поддержки

принятия решений на этапе инстанцирования любого социально-экономического проекта, когда методы и процедуры проектного и предпроектного анализа ещё не могут быть использованы.

В целом же, перспективы ДВ-УФО метода во многом определяются тем, насколько наберет силу наметившаяся тенденция использования так называемых «минимальных» моделей: узкоспециализированных упрощённых моделей, использующих простейшие методы анализа и алгоритмы малой сложности, но обеспечивающих (по Парето) 80% результата.

III. ТЕХНОЛОГИИ СИСТЕМНОГО АНАЛИЗА

1. Структурно-функциональное (процессное) моделирование

1.1. Нотация DFD

Диаграммы потоков данных (***DFD***) являются средством моделирования функциональных требований к проектируемой системе. С их помощью эти требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных. Главная цель таких средств – продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

Возможности этих диаграмм учитывать общесистемные закономерности представлены в Приложении №1.

Нормативная система.

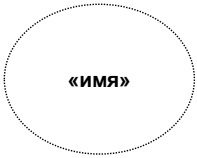
Для изображения DFD-диаграмм традиционно используются две различные нотации: Йордана и Гейна-Сарсона. Алфавит, используемый для построения DFD-диаграмм, представлен в таблице 3.1.

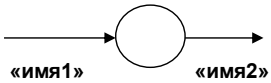
Таблица 3.1

Нормативная система (нотация) DFD.

Элемент алфавита	*Нотация Йордана*	*Нотация Гейна-Сарсона*
Поток данных Используется для моделирования передачи информации (или даже физических компонент) из одной части системы в другую. На диаграммах изображаются именованными стрелками, ориентация которых указывает направление потока	«имя» →	«имя» →
Процесс Используется для моделирования процесса преобразования входного потока в выходной. Его имя должно содержать глагол в неопределенной форме с последующим дополнением (например, «Вычислить высоту»). Кроме того, каждый процесс должен иметь уникальный номер для ссылок на него внутри диаграммы, который совместно с номером диаграммы уникален во всей модели	«имя»	«имя» «номер»

Элемент алфавита	*Нотация Йордана*	*Нотация Гейна-Сарсона*
Хранилище (накопитель) данных Используется для моделирования данных (или даже физических компонент), которые будут сохраняться между процессами. Информация, которую оно содержит, может использоваться в любое время после ее определения, при этом данные могут выбираться в любом порядке. Имя хранилища должно идентифицировать его содержимое и быть существительным. В случае, когда поток данных входит или выходит в/из хранилища, и его структура соответствует структуре хранилища, он должен иметь то же самое имя, которое нет необходимости отражать на диаграмме		
Внешняя сущность (терминатор) Используется для моделирования сущностей вне системы (контекстных сущностей), являющихся источником или приемником системных данных. Ее имя должно содержать существительное, например, «Склад». Предполагается, что объекты, представленные такими сущностями, не должны участвовать ни в какой обработке		
Управляющий поток Используется для моделирования связи, через которую передается управляющая информация. Его имя не должно содержать глаголов, а только существительные и прилагательные. Обычно управляющий поток имеет дискретное, а не непрерывное значение. Это может быть, например, сигнал, представляющий состояние или вид операции Имеются следующие типы управляющих потоков: а) **Т-поток** (trigger flow) – поток управления, который может вызывать выполнение процесса. При этом процесс как бы включается одной короткой операцией. Это аналог выключателя света, единственным нажатием которого «запускается» процесс горения лампы		

Элемент алфавита	*Нотация Йордана*	*Нотация Гейна-Сарсона*
б) **А-поток** (activator flow) – поток управления, который может изменять выполнение процесса. Используется для обеспечения непрерывности выполнения процесса до тех пор, пока поток «включен» (т.е. течет непрерывно); с «выключением» потока выполнение процесса завершается. Это – аналог переключателя лампы, которая может быть как включена, так и выключена. в) **E/D-поток** (enable/disable flow) – поток управления, который может переключать выполнение процесса. Течение по E-линии вызывает выполнение процесса, которое продолжается до тех пор, пока не возбуждается течение по D-линии. Это аналог выключателя с двумя кнопками: одной для включения света, другой для его выключения. Можно использовать 3 типа таких потоков: E-поток, D-поток, E/D-поток		
Управляющий процесс Используется для моделирования преобразователя входных управляющих потоков в выходные управляющие потоки; при этом точное описание этого преобразования должно задаваться в спецификации управления. Его имя указывает на тип управляющей деятельности, описанной в спецификации. Логически управляющий процесс есть командный пункт, реагирующий на изменения внешних условий, которые сообщаются ему с помощью управляющих потоков, и продуцирующий в соответствии со своей внутренней логикой команды для других процессов системы		
Управляющее хранилище Используется для моделирования управляющей информации, которая будут сохраняться между процессами. Содержащаяся в нем управляющая информация может использоваться в любое время после ее занесения в хранилище, при этом соответствующие данные могут быть использованы в		

Элемент алфавита	*Нотация Йордана*	*Нотация Гейна-Сарсона*
произвольном порядке. Имя управляющего хранилища должно идентифицировать его содержимое и быть существительным. Управляющее хранилище отличается от традиционного тем, что может содержать только управляющие потоки; все другие их характеристики идентичны		
Групповой узел Используется для моделирования расщепления и объединения потоков.		
Узел-предок Используется для связывания входящих и выходящих потоков между детализируемым процессом и детализирующей DFD-диаграммой		
Неиспользуемый узел Используется для моделирования ситуации, когда декомпозиция данных производится в групповом узле, при этом требуются не все элементы входящего в узел		
Узел изменения имени Используется для обеспечения возможности неоднозначного именования потоков, содержимое которых эквивалентно. Например, если при проектировании разных частей системы один и тот же фрагмент данных получил различные имена, то эквивалентность соответствующих потоков данных обеспечивается узлом изменения имени. При этом один из потоков данных является входным для данного узла а другой – выходным		
Текст В свободном формате в любом месте диаграммы		

Построение модели.

Главная цель построения модели в виде иерархического множества DFD-диаграмм заключается в том, чтобы сделать требования ясными и понятными на каждом уровне детализации, а также разбить эти требования на части с точно определенными отношениями между ними. Для достижения этого рекомендуется пользоваться следующими правилами [Калянов, 1997]:

1. Размещать на каждой диаграмме от 3 до 6–7 процессов. Верхняя граница соответствует человеческим возможностям одновременного восприятия и понимания структуры сложной системы с множеством внутренних связей, нижняя граница выбрана по соображениям здравого смысла: нет необходимости детализировать процесс диаграммой, содержащей всего один или два процесса.

2. Не загромождать диаграммы несущественными на данном уровне детализации сущностями.

3. Декомпозицию потоков данных осуществлять параллельно с декомпозицией процессов; эти две работы должны выполняться одновременно, а не одна после завершения другой.

4. Выбирать ясные, отражающие суть дела, имена процессов и потоков для улучшения восприятия диаграмм, при этом не рекомендуется использовать аббревиатуры.

5. Однократно определять функционально идентичные процессы на самом верхнем уровне иерархии, где такой процесс необходим, и ссылаться на него на нижних уровнях иерархии.

6. Пользоваться простейшими диаграммными техниками: если что-либо возможно описать с помощью DFD-диаграмм, то это и необходимо делать, а не использовать для описания более сложные объекты.

7. Отделять управляющие структуры от обрабатывающих структур (т.е. процессов), локализовать управляющие структуры.

В соответствии с этими рекомендациями процесс построения модели разбивается на следующие этапы [там же]:

1. Идентификация внешних (контекстных) объектов, с которыми система должна быть связана.

2. Расчленение множества требований и организация их в основные функциональные группы.

3. Идентификация основных видов информации, циркулирующей между системой и внешними объектами.

4. Предварительная разработка контекстной диаграммы, на которой основные функциональные группы представляются процессами, внешние (контекстные) объекты – внешними сущностями, основные виды информации – потоками данных между процессами и внешними сущностями.

5. Изучение предварительной контекстной диаграммы и внесение в неё изменений по результатам ответов на возникающие при этом изучении вопросы.

6. Построение контекстной диаграммы путём объединения всех процессов предварительной диаграммы в один процесс, а также группирования потоков.

7. Формирование DFD-диаграммы первого уровня на базе процессов предварительной контекстной диаграммы.

8. Проверка основных требований по DFD-диаграмме первого уровня.

9. Декомпозиция каждого процесса текущей DFD-диаграммы с помощью детализирующей диаграммы или спецификации процесса.

10. Проверка основных требований по DFD-диаграмме соответствующего уровня.

11. Добавление определений новых потоков в словарь данных при каждом их появлении на диаграммах.

12. Параллельное (с процессом декомпозиции) изучение требований (в том числе и вновь поступающих), разбиение их на элементарные и идентификация процессов или спецификаций процессов, соответствующих этим требованиям.

13. После построения двух-трех уровней проведение ревизии с целью проверки корректности и улучшения понимаемости модели.

14. Построение спецификации процесса (а не простейшей диаграммы) в случае, если некоторую функцию сложно или невозможно выразить комбинацией процессов.

Важную специфическую роль в модели играет специальный вид DFD-диаграммы – ***контекстная диаграмма***, моделирующая систему наиболее общим образом. Контекстная диаграмма отражает интерфейс системы с внешним миром, а именно, информационные потоки между системой и внешними сущностями, с которыми она должна быть связана. Она идентифицирует эти внешние сущности, а так же, как правило, единственный процесс, отражающий главную цель или природу системы насколько это возможно. И хотя контекстная диаграмма выглядит тривиальной, несомненная ее полезность заключается в том, что она устанавливает границы анализируемой системы. Каждый проект должен иметь ровно одну контекстную диаграмму, при этом нет необходимости в нумерации единственного ее процесса.

Декомпозиция DFD-диаграммы осуществляется на основе процессов: каждый процесс может раскрываться с помощью DFD-диаграммы нижнего уровня. DFD-диаграмма первого уровня строится как декомпозиция процесса, который присутствует на контекстной

диаграмме. Построенная диаграмма первого уровня также имеет множество процессов, которые в свою очередь могут быть декомпозированы. Таким образом, строится иерархия DFD-диаграмм с контекстной диаграммой в корне дерева. Этот процесс декомпозиции продолжается до тех пор, пока процессы могут быть эффективно описаны с помощью коротких (до одной страницы) спецификаций процессов.

При таком построении иерархии DFD-диаграмм каждый процесс более низкого уровня необходимо соотнести с процессом верхнего уровня. Обычно для этой цели используются структурированные номера процессов. Так, например, если мы детализируем процесс номер 2 на диаграмме первого уровня, раскрывая его с помощью DFD-диаграммы, содержащей три процесса, то их номера будут иметь следующий вид: 2.1, 2.2 и 2.3. При необходимости можно перейти на следующий уровень, т.е. для процесса 2.2 получим 2.2.1, 2.2.2. и т.д.

Пример иерархии DFD-диаграмм, описывающих ***систему управления лифтом***, аналогичную представленной в работе [Йордан, 1999] (Elevator Control System – ECS), подготовленный с применением инструментального пакета BPwin, изображен на рисунках 3.1 – 3.6. Дополнительные возможности нотации позволяют, в частности, с помощью двойных стрелок отличить материальные потоки от информационных. Данный пример используется далее для описания различных аспектов технологии моделирования ЗУМ.

Диаграммы (3.1–3.6) построены с учетом следующих требований к системе управления лифтом [Йордан, 1999, стр. 15–17]:

- лифт используется для перевозки людей в 40-а этажном офисном здания обычным способом (при вызове лифта с этажа он должен останавливаться при попутном движении; лифт не должен менять направления, пока едущие в нем пассажиры не достигнут назначенных этажей; пустой лифт должен стоять там, где он остановился последний раз);
- ECS должна реагировать на электромагнитные сигналы от датчиков (каждого этажа и состояния лифта), а также кнопок (панелей вызова лифта с каждого этажа и панели назначения этажа в лифте);
- должна обеспечиваться индикация этажа прибытия лифта и принятия ECS вызовов и назначений;
- ECS должна обеспечивать «связанное с этажом» составление расписания движения лифта, путем выработки управляющих сигналов «стоп», «вверх» «вниз» на мотор в зависимости от текущего этажа, нажатых кнопок и состояния лифта.

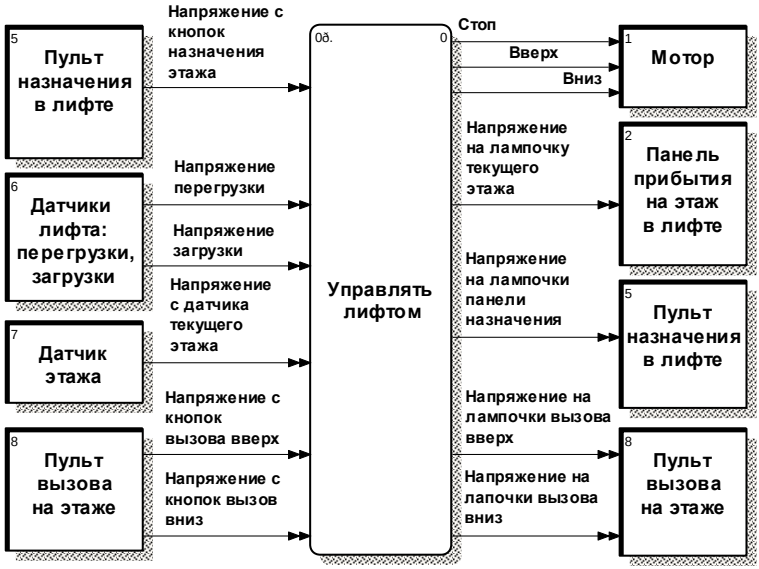


Рис. 3.1. Контекстная DFD-диаграмма ECS

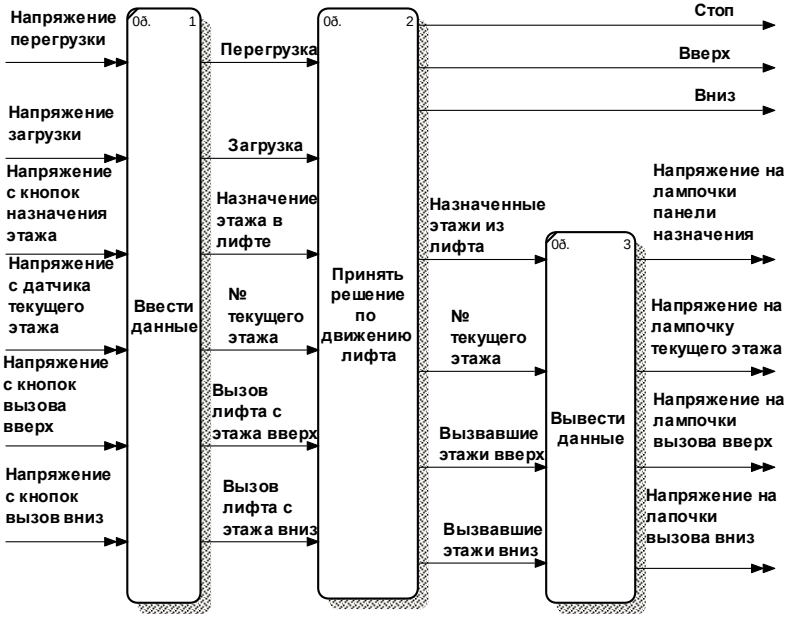


Рис. 3.2. DFD-диаграмма ECS первого уровня

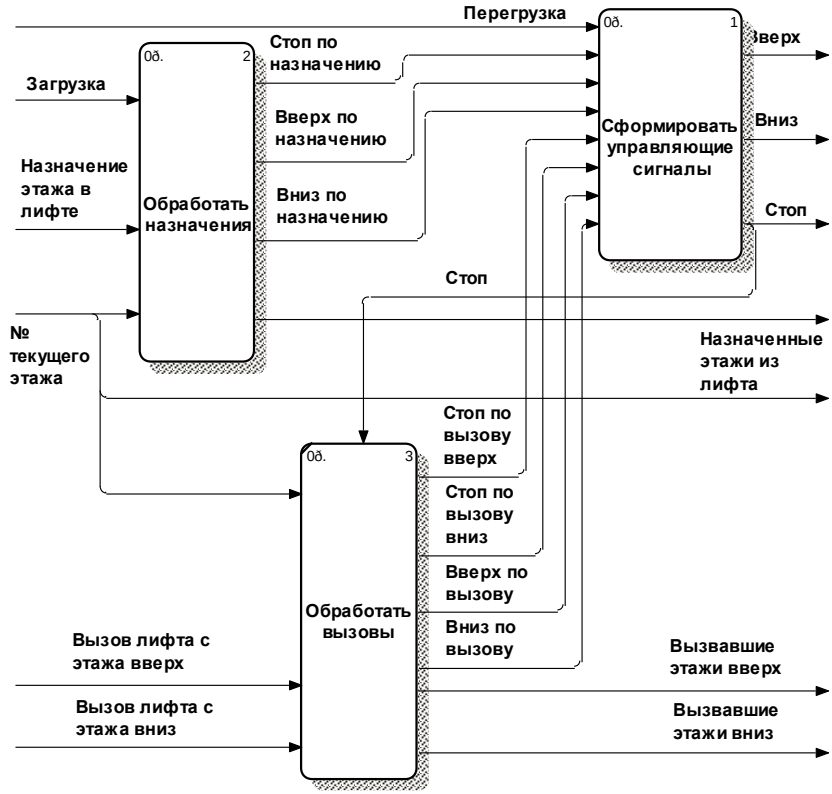


Рис. 3.3. DFD-диаграмма второго уровня (процесса принятия решений)

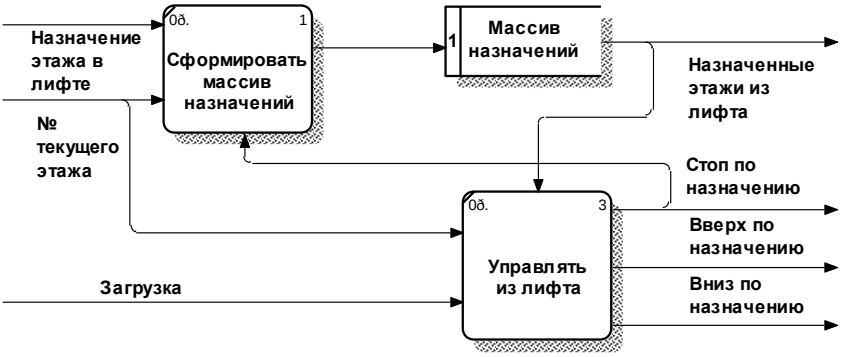


Рис. 3.4. DFD-диаграмма третьего уровня (обработка назначений)

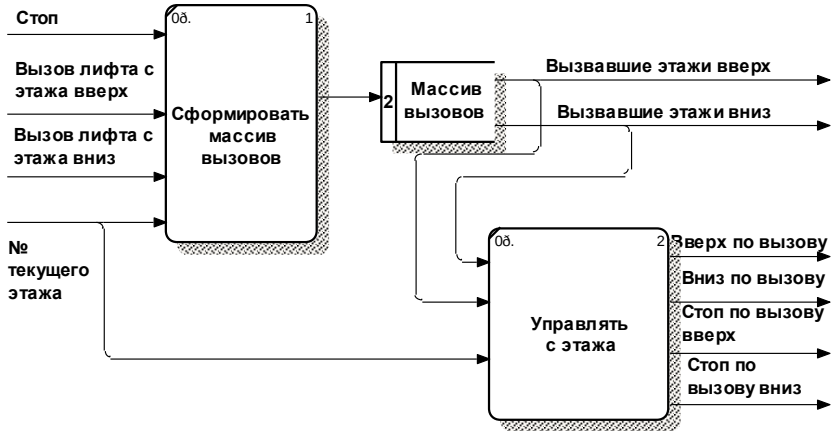


Рис. 3.5. DFD-диаграмма третьего уровня (обработка вызовов)

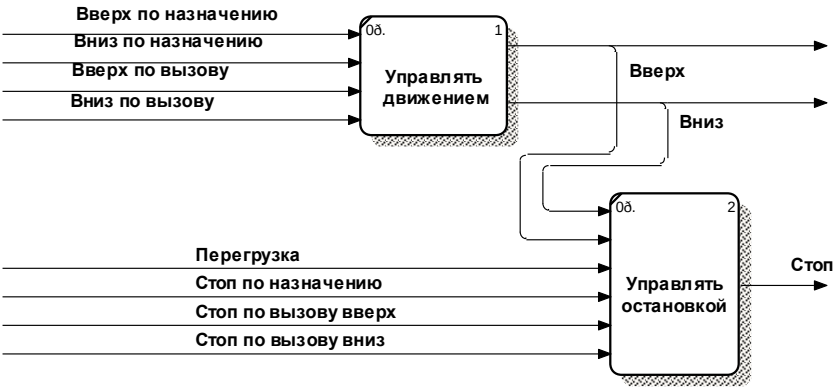


Рис. 3.6. DFD-диаграмма третьего уровня (формирование управляющих сигналов)

Словарь данных.

Диаграммы потоков данных обеспечивают описание взаимодействий функциональных компонент системы (ее структуры), но не имеют, сами по себе, средств для описания связей между компонентами и их функций, а именно, какая информация преобразуется процессами и как она преобразуется. Для решения первой из перечисленных проблем предназначено текстовое средство моделирования, служащее для описания содержания преобразуемой информации. Оно получило название «словарь данных».

Словарь данных представляет собой определенным образом организованный список всех элементов, составляющих потоки данных системы с их точными определениями. Это дает возможность различным категориям пользователей (от системного аналитика до программиста) иметь общее понимание всех входных и выходных потоков и компонент хранилищ.

Определение элементов данных в словаре осуществляется с помощью описаний следующих видов [Калянов, 1997]:

- описанием характеристик потоков и хранилищ, изображенных на DFD-диаграммах;
- описанием композиций данных, движущихся вдоль потоков, т.е. комплексных данных, которые могут расчленяться на элементарные (например, «Адрес» содержит «Индекс», «Город», «Улица» и т.д.);
- описанием композиций данных в хранилище.

Для каждого потока данных в словаре необходимо хранить ***имя*** потока, его ***тип*** и ***атрибуты***. Информация по каждому потоку состоит из ряда словарных статей, каждая из которых начинается с ключевого слова – заголовка соответствующей статьи, которому предшествует символ «**@**».

Типы потоков определяются в словаре как:

- простые (элементарные) или групповые (комплексные) потоки;
- внутренние (существующие только внутри системы) или внешние (связывающие систему с другими системами) потоки;
- потоки данных или потоки управления;
- непрерывные (принимающие любые значения в пределах диапазона) или дискретные (принимающие определенные значения) потоки.
- Атрибуты потока данных включают:
- имена-синонимы потока данных в соответствии с узлами изменения имени;
- определение с помощью БНФ-нотации (см. далее) в случае группового потока;
- единицы измерения потока;
- диапазон значений для непрерывного потока, типичное его значение и информацию по обработке экстремальных значений;
- список значений и их смысл для дискретного потока;
- список номеров диаграмм, в которых поток встречается;
- список потоков, в которые данный поток входит;

- комментарий, включающий дополнительную информацию (например о цели введения данного потока).

БНФ-нотация позволяет формально описать расщепление/объединение потоков. Поток может расщепляться на собственные отдельные ветви, на компоненты потока-предка или на то и другое одновременно.

При расщеплении/объединении потока существенно, чтобы каждый компонент потока-предка являлся именованным. При объединении подпотоков нет необходимости осуществлять исключение общих компонент, а при расщеплении подпотоки могут иметь такие общие (одинаковые) компоненты.

Точные определения потоков содержатся в словаре данных, а не на диаграммах. Например, на диаграмме может иметься групповой узел с входным потоком X и выходными подпотоками Y и Z. Однако, это вовсе не означает, что соответствующее определение в словаре данных обязательно должно быть X=Y+Z. Это определение может быть следующим: X=A+B+C; Y=A+B; Z=B+C. Такие определения хранятся в словаре данных в так называемой БНФ-статье.

БНФ-статья используется для описания компонент данных в потоках данных и в хранилищах. Ее синтаксис имеет следующий вид:

@БНФ = <простой оператор> ! <БНФ-выражение>,

где

<***простой оператор***> есть текстовое описание, заключенное в «/», а <***БНФ-выражение***> есть выражение в ***форме Бэкуса-Наура***, заключенное в «***[ ]***» и допускающее следующие операции и отношения: **=** – означает «***композиция из***»; **+** – означает «***И***»; **!** – означает «***ИЛИ***»; **()** – означает, что компонент в скобках не обязателен; **{}** – означает итерацию компонента в скобках; **« »** – означает литерал.

Итерационные скобки могут иметь нижний и верхний предел, например:

3 {болт} 7 – от 3 до 7 итераций
1 {болт} – 1 и более итераций
{шайба} 3 – не более 3 итераций

БНФ-выражение может содержать произвольные комбинации операций:

@БНФ = [винт ! болт + 2 {гайка} 2 + (прокладка) ! клей]

Ниже приведен пример описания потока данных подсистемы принятия решений ECS «№ текущего этажа» (рис. 3.2–3.5) с помощью БНФ-статьи:

@ИМЯ = № текущего этажа
@ТИП = Простой, внутренний, дискретный поток данных
@БНФ = [«1» ! «2» ! … ! «39» ! «40»]

Далее приведен пример описания входных потоков данных блока формирования управляющих сигналов на примере потока «Перегрузка», а также выходных потоков этого блока на примере потока «Стоп» (рис. 3.6):

@ИМЯ = перегрузка
@ТИП = Простой, внутренний, дискретный поток данных
@БНФ = [«0» ! «1»]
@ИМЯ = стоп
@ТИП = Простой, внешний, дискретный поток управления
@БНФ = [«0» ! «1»]

Спецификация процесса.

Спецификация процесса (СП) используется для описания процесса, когда нет необходимости детализировать его с помощью DFD-диаграммы (т.е. если он невелик и его описание может занимать не более одной страницы текста). СП представляет собой описание алгоритма, соответствующего данному процессу и трансформирующего входные потоки данных в выходные. Множество всех СП является полной спецификацией системы. СП содержит номер и/или имя процесса, списки входных и выходных данных и тело (описание) процесса.

Известно большое число разнообразных методов, позволяющих задать тело процесса: от **структурированного естественного языка** или псевдокода до **визуальных языков проектирования** (типа **FLOW-форм**), а также формальных языков программирования.

Независимо от используемого метода СП должна начинаться со следующих ключевых слов:

@ВХОД = <имя символа данных >
@ВЫХОД = <имя символа данных>
@СПЕЦПРОЦ**,** где

<***имя символа данных***> – соответствующее имя из словаря данных.

Например, для процессов формирования массивов (рис. 3.4 и 3.5),

@ВХОД = Назначение этажа в лифте; № текущего этажа; Стоп по назначению
@ВЫХОД = Массив назначений
@СПЕЦПРОЦ

Сформировать очередь №№ этажей, назначенных в лифте
@
@ВХОД = Вызов лифта с этажа вверх; Вызов лифта с этажа вниз; № текущего этажа; Стоп
@ВЫХОД = Массив вызовов
@СПЕЦПРОЦ
Сформировать очередь №№ вызвавших этажей
@

Иногда в СП задаются **пред- и пост-условия** выполнения данного процесса. В пред-условии записываются объекты, значения которых должны быть истинны перед началом выполнения процесса, что обеспечивает определенные гарантии безопасности для пользователя. Аналогично, в случае наличия пост-условия гарантируется, что значения всех входящих в него объектов будут истинны при завершении процесса.

СП должна удовлетворять следующим требованиям [Калянов, 1997]:

- для каждого процесса нижнего уровня должна существовать одна и только одна спецификация;
- спецификация должна определять способ, но не метод преобразования входных потоков в выходные;
- спецификация должна стремиться к ограничению избыточности: не следует переопределять то, что уже было определено на диаграмме или в словаре данных;

Ниже рассматриваются некоторые наиболее часто используемые методы задания СП.

Структурированный естественный язык является разумной комбинацией строгости языка программирования и читабельности естественного языка. Он состоит из подмножества слов, организованных в определенные логические структуры, арифметических выражений и диаграмм. В состав структурированного языка входят следующие основные символы:

- глаголы, ориентированные на действия и применяемые к объектам данной предметной области;
- существительные, определенные на любой стадии анализа, моделирования и проектирования (например, при разработке информационной системы: задачи, процедуры, символы данных и т.п.);
- предлоги и союзы, используемые в логических отношениях;
- общеупотребительные математические, логические, физические и технические термины;

- арифметические выражения;
- таблицы, диаграммы, графики и т.п.;
- комментарии.

Управляющие конструкции структурированного языка имеют вид:

Последовательная конструкция:
ВЫПОЛНИТЬ функция1
ВЫПОЛНИТЬ функция2
ВЫПОЛНИТЬ функция3

Конструкция выбора:
ЕСЛИ<условие>ТО
ВЫПОЛНИТЬ функция1
ИНАЧЕ
ВЫПОЛНИТЬ функция2
КОНЕЦЕСЛИ

Итерация:
ДЛЯ <условие>
ВЫПОЛНИТЬ функция
КОНЕЦДЛЯ

Или
ПОКА <условие>
ВЫПОЛНИТЬ функция
КОНЕЦПОКА

При использовании структурированного естественного языка приняты следующие соглашения [Калянов, 1997]:

- логика процесса выражается в виде комбинации последовательных конструкций, конструкций выбора и итераций;
- ключевые слова ЕСЛИ, ВЫПОЛНИТЬ, ИНАЧЕ и т.д. должны быть написаны заглавными буквами;
- слова или фразы, определенные в словаре данных, должны быть написаны заглавными буквами;
- глаголы должны быть активными, недвусмысленными и ориентированными на целевое действие (заполнить, вычислить, извлечь, а не модернизировать, обработать);
- логика процесса должна быть выражена четко и недвусмысленно.

Таблицы и деревья решений применяются в тех случаях, когда структурированный естественный язык оказывается неприемлем. Например, если действие зависит от нескольких переменных, которые в совокупности могут продуцировать большое число комбинаций, то его описание будет слишком запутанным и с большим числом уровней вложенности. Для описания подобных действий традиционно используются таблицы и деревья решений.

Проектирование СП с помощью **таблиц решений** (ТР) заключается в задании матрицы, отображающей множество входных **условий** во множество выходных **действий.**

ТР состоит из двух частей. Верхняя часть таблицы используется для определения условий. Обычно условие является **ЕСЛИ**-частью оператора **ЕСЛИ-ТО** и требует ответа «да-нет». Нижняя часть ТР используется для определения действий, т.е. **ТО**-части оператора **ЕСЛИ-ТО**.

Левая часть ТР содержит наименования условий и действий, а в правой части перечисляются все возможные комбинации условий и, соответственно, указывается, какие конкретно действия и в какой последовательности выполняются, когда определенная комбинация условий имеет место.

Построение ТР рекомендуется осуществлять путем выполнения следующих шагов [Калянов, 1997]:

1. Идентифицировать все условия (или переменные), участвующие в описываемом процессе. Идентифицировать все значения, которые каждая переменная может иметь.
2. Вычислить число комбинаций условий.
3. Идентифицировать каждое из возможных действий, которые могут вызываться в описываемом процессе.
4. Построить пустую таблицу, включающую все возможные условия и действия, а также номера комбинаций условий.
5. Выписать и занести в таблицу все возможные комбинации условий.
6. Редуцировать комбинации условий (если это возможно).
7. Проверить каждую комбинацию условий и идентифицировать соответствующие выполняемые действия.
8. Выделить комбинации условий, для которых в данном процессе нет выполняемых действий.
9. Обсудить построенную таблицу.

Поясним сказанное выше на примере СП управления движением лифта (рис. 3.6, процесс №1) и СП управления остановкой лифта (рис. 3.6, процесс №2) (см. таблицы 3.2 и 3.3 соответственно).

При составлении таблиц использованы следующие обозначения входных потоков процессов управления движением и остановкой лифта, как условий: перегрузка – **П**, стоп по назначению – **Сн**, вверх по назначению – **Вн**, вниз по назначению – **Нн**, стоп по вызову вверх – **Свв**, стоп по вызову вниз – **Снв**, вверх по вызову – **Вв**, вниз по вызову – **Нв**; и выходных потоков этих процессов, как действий: стоп – **С**, вверх – **В**, вниз – **Н**.

Таблица 3.2

Таблица решений процесса управления движением лифта

Условия	1	2	3	4	5	6	7	8	9
Вн	0	0	0	0	0	0	1	1	1
Нн	0	0	0	1	1	1	0	0	0
Вв	0	0	1	0	0	1	0	0	1
Нв	0	1	0	0	1	0	0	1	0
Действия									
В	0	0	1	0	0	0	1	1	1
Н	0	1	0	1	1	1	0	0	0

Таблица 3.3

Таблица решений процесса управления остановкой лифта

Условия	1	2	3	4	5	6	7	8	9	10	11	12	13	14
П	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Сн	0	0	0	0	0	0	0	0	0	0	0	0	1	*
Свв	0	0	0	0	0	0	1	1	1	1	1	1	*	*
Снв	0	0	0	1	1	1	0	0	0	1	1	1	*	*
В	0	0	1	0	0	1	0	0	1	0	0	1	*	*
Н	0	1	0	0	1	0	0	1	0	0	1	0	*	*
Действия														
С	0	0	0	1	1	0	1	0	1	1	1	1	1	1

Обсудим построенные таблицы. Данные таблицы визуализируют обычную логику работы лифта в офисном здании. Согласно табл. 3.2, если из лифта было назначено движение вниз или вверх, то независимо от вызовов лифт будет выполнять эти назначения. При этом назначений вверх и вниз одновременно быть не может, что обеспечивается формированием очереди №№ этажей, назначенных в лифте (см. рис. 3.4). При отсутствии назначений будут выполняться вызова с этажей, которые по той же причине (см. рис. 3.5) не могут быть одновременно и вверх, и вниз. Согласно табл. 3.3 наличие перегрузки или требования «остановится» из лифта вызывают безусловную остановку. Требования «остановится» с этажа выполняются только при попутном движении лифта. Управляющий сигнал «**С**» при наличии требования на остановку с этажа (вызова с этажа) в случае, когда лифт стоит (**В** = 0; **Н** = 0), используется для открывания двери на вызвавшем этаже механической системой лифта.

Вариантом таблицы решений является **дерево решений** (ДР), позволяющее взглянуть на процесс условного выбора с позиции схемы.

Обычно ДР используется при малом числе действий и когда не все комбинации условий возможны, а ТР – при большом числе действий и когда возможно большое число комбинаций условий. На основе ТР легко осуществляется автоматическая кодогенерация.

Визуальные языки проектирования являются относительно новой, оригинальной методикой разработки СП. Они базируются на основных идеях структурного программирования и позволяют определять потоки управления с помощью специальных иерархически организованных схем.

Одним из наиболее известных подходов к визуальному проектированию СП является подход с использованием **FLOW-форм.** Каждый символ FLOW-формы имеет вид прямоугольника и может быть вписан в любой прямоугольник любого другого символа. Символы помечаются с помощью предложений на естественном языке или с использованием математической нотации.

Символы FLOW-форм приведены на рисунке 3.7. Каждый символ является блоком обработки. Каждый прямоугольник внутри любого символа также представляет собой блок обработки.

Рис. 3.7. Символы FLOW-форм

На рисунке 3.8 приведен пример использования данного подхода при проектировании СП, обеспечивающего упорядочивание определенным образом элементов массива и являющегося фрагментом алгоритма сортировки методом «поплавка» [Калянов, 1997].

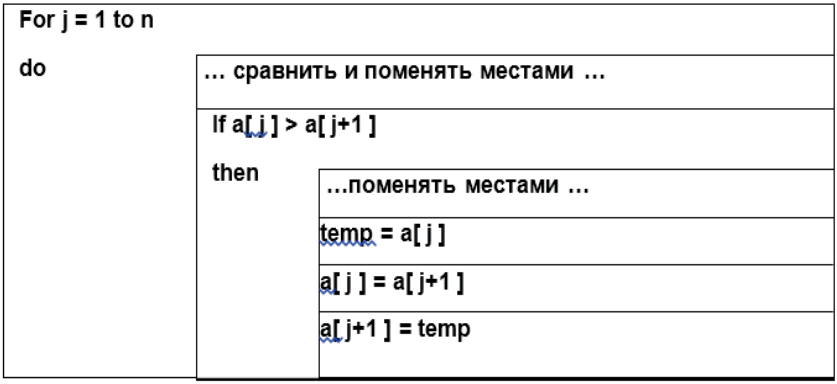


Рис. 3.8. Пример FLOW-формы

Дальнейшее развитие FLOW-формы получили в **диаграммах Насси-Шнейдермана.** На этих диаграммах символы последовательной обработки и цикла изображаются так же, как и соответствующие символы FLOW-форм. В символах условного выбора и case-выбора собственно условие располагается в верхнем треугольнике, выбираемые варианты – на нижних сторонах треугольника, а блоки обработки – под выбираемыми вариантами. Диаграмма Насси-Шнейдермана для вышеприведенного примера изображена на рисунке 3.9.

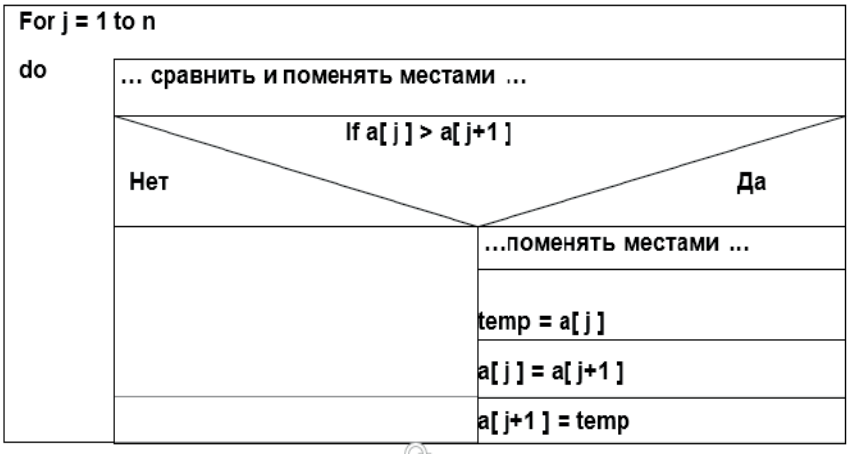


Рис. 3.9. Пример диаграммы Насси-Шнейдермана

В литературе отмечаются следующие недостатки DFD-диаграмм:

1. Не поддерживают объектно-ориентированного проектирования.

2. Требуют проведения специального сквозного контроля диаграмм одного или разных типов, т.е. соответственно вертикального и горизонтального балансирования диаграмм, для выявления весьма вероятных ошибок.

3. Обеспечивают удобное описание функционирования компонент системы, но не снабжают аналитика средствами описания деталей этих компонент, а именно, какая информация преобразуется процессами и как она преобразуется.

4. Ориентированы на системных аналитиков и программистов и не учитывают особенности восприятия менеджерами своей предметной области.

5. Не соответствуют требованию «лишения разработчика той части творческих возможностей, которые ведут к разнообразию представления организационных моделей». Данное требование связано с тем, что инструментарий моделирования должен быть средством поддержки принятия решений, а не художественного творчества.

6. Приспособлены для хорошо специфицированных и стандартизованных «западных» бизнес-процессов. При моделировании больших, сложных, иерархических систем создаваемые диаграммы становятся слишком сложными для понимания.

1.2. Стандарт IDEF0

Далее рассмотрим два стандарта системного структурного анализа, входящие в серию федеральных стандартов США «***Icam Definition***», в соответствии с [Калянов, 1997; Верников, 1–3; Маклаков, 2002; Технология моделирования; Дубейковский, 2004]. Подробную информацию по всем стандартам этой серии можно найти на сайте http://www.idef.com.

Стандарт ***IDEF0*** (FIPS183) предназначен для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции. Данный документ представляет собой оформление (по инициативе Министерства обороны США) в виде стандарта технологии анализа сложных систем ***SADT*** (Structured Analysis and Design Technique), разработанной группой американских аналитиков во главе с Дугласом Россом в 1973 году.

Метод, предлагаемый стандартом IDEF0, предназначен для функционального моделирования, то есть моделирования выполнения функций объекта, путем создания описательной графической модели, показывающей что, как и кем делается в рамках функционирования предприятия. Функциональная модель представляет собой структурированное изображение функций производственной системы или среды, информации и объектов, связывающих эти функции. Модель строится методом декомпозиции: от крупных составных структур к более мелким, простым. Элементы каждого уровня декомпозиции представляют собой действия по переработке информационных или материальных ресурсов при определенных условиях с использованием заданных механизмов. Каждое действие раскладывается на более мелкие операции по переработке определенной части информационных или материальных ресурсов при определенных условиях с использованием части заданных механизмов. Аналогично раскладываются операции. Последний шаг декомпозиции должен приводить к получению модели, степень детализации которой удовлетворяет требованиям, заданным в самом начале процесса создания модели.

Возможности этих моделей учитывать общесистемные закономерности представлены в Приложении №1.

Методология IDEF0 основана на следующих положениях:

1. ***Система и модель***. Модель – искусственный объект, представляющий собой отображение (образ) системы и ее компонентов. ***M*** моделирует ***A***, если ***M*** отвечает на вопросы относительно ***A***. Здесь ***M*** – модель, ***A*** – моделируемый объект (оригинал). Модель разрабатывают для понимания, анализа и принятия решений о реконструкции (реинжиниринге) или замене существующей, либо проектировании новой системы. Система представляет собой совокупность взаимосвязанных и взаимодействующих частей, выполняющих некоторую полезную работу. Частями (элементами) системы могут быть любые комбинации разнообразных сущностей, включающие людей, информацию, программное обеспечение, оборудование, изделия, сырье или энергию. Модель описывает, что происходит в системе, как ею управляют, какие сущности она преобразует, какие средства использует для выполнения своих функций и что производит.

2. ***Блочное моделирование и его графическое представление***. Основной концептуальный принцип методологии IDEF – представление любой изучаемой системы в виде набора взаимодействующих и взаимосвязанных блоков, отображающих процессы, операции, действия, происходящие в изучаемой системе. В IDEF0 все, что происходит

в системе и ее элементах, принято называть функциями. Каждой функции ставится в соответствие блок. На IDEF0-диаграмме (чаще говорят SADT-диаграмме), основном документе при анализе и проектировании систем, блок представляет собой прямоугольник. Интерфейсы, посредством которых блок взаимодействует с другими блоками или с внешней по отношению к моделируемой системе средой, представляются стрелками, входящими в блок или выходящими из него. Входящие стрелки показывают, какие условия должны быть одновременно выполнены, чтобы функция, описываемая блоком, осуществилась.

3. ***Строгость и формализм***. Разработка моделей IDEF0 требует соблюдения ряда строгих формальных правил, обеспечивающих преимущества методологии в отношении однозначности, точности и целостности сложных многоуровневых моделей. Эти правила описываются ниже с точки зрения технологии SADT. Здесь отмечается только основное из них: все стадии и этапы разработки и корректировки модели должны строго, формально документироваться с тем, чтобы при ее эксплуатации не возникало вопросов, связанных с неполнотой или некорректностью документации.

4. ***Итеративное моделирование***. Разработка модели в IDEF0 представляет собой пошаговую, итеративную процедуру. На каждом шаге итерации разработчик предлагает вариант модели, который подвергают обсуждению, рецензированию и последующему редактированию, после чего цикл повторяется. Такая организация работы способствует оптимальному использованию знаний системного аналитика, владеющего методологией и техникой IDEF0, и знаний специалистов – экспертов в предметной области, к которой относится объект моделирования.

5. ***Отделение «организации» от «функций»***. При разработке моделей следует избегать изначальной «привязки» функций исследуемой системы к существующей организационной структуре моделируемого объекта (предприятия, фирмы). Это помогает избежать субъективной точки зрения, навязанной организацией и ее руководством. Организационная структура должна явиться результатом использования (применения) модели. Сравнение результата с существующей структурой позволяет, во-первых, оценить адекватность модели, а во-вторых – предложить решения, направленные на совершенствование этой структуры.

Примеры задач, решаемых на основе методологии моделирования IDEF0:

- Анализ и реинжиниринг бизнес-процессов.
- Разработка информационной системы управления данными о качестве.

- Разработка регламентов и процедур обеспечения качества продукции и создания систем обработки данных о качестве. Функциональная модель позволяет заменить традиционные руководства по качеству в виде описательных текстовых бумажных документов – стандартизованными электронными моделями, целостность и непротиворечивость которых поддерживается автоматически. При необходимости из них всегда можно получить бумажный отчет в привычном виде.
- Проектирование информационной инфраструктуры, процедур и регламентов информационного взаимодействия.
- Задачи анализа рисков в плане информационной безопасности.

Рассмотрим в соответствии с работой [Верников, 2] принципы построения диаграмм по технологии SADT (IDEF0).

Графический язык SADT прост и гармоничен. В основе методологии лежат четыре основных понятия. Первым из них является понятие «***функциональный блок***». Функциональный блок графически изображается в виде прямоугольника (см. рис. 3.10) и олицетворяет собой некоторую конкретную функцию в рамках рассматриваемой системы.

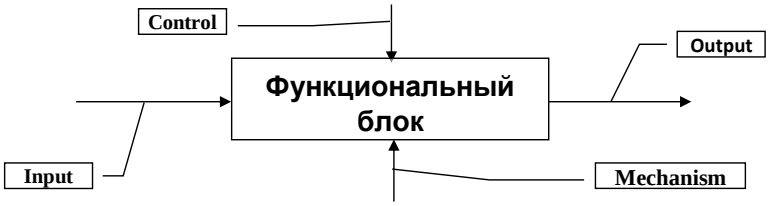


Рис. 3.10. Функциональный блок на SADT-диаграмме

По требованиям стандарта название каждого функционального блока должно быть сформулировано в глагольном наклонении (например, «***производить услуги***», а не «***производство услуг***»). Каждая из четырех сторон функционального блока имеет своё определенное значение (роль), при этом: верхняя сторона имеет значение «***управление***» (***control***); левая сторона имеет значение «***вход***» (***input***); правая сторона имеет значение «***выход***» (***output***); нижняя сторона имеет значение «***механизм***» (***mechanism***). Каждый функциональный блок в рамках единой рассматриваемой системы должен иметь свой уникальный идентификационный номер.

Вторым «китом» методологии SADT является понятие «***интерфейсная дуга***». Также интерфейсные дуги часто называют потоками или стрелками. Интерфейсная дуга отображает элемент системы,

который обрабатывается функциональным блоком или оказывает иное влияние на функцию, отображенную данным функциональным блоком. Графическим отображением интерфейсной дуги является однонаправленная стрелка. Каждая интерфейсная дуга должна иметь свое уникальное наименование. По требованию стандарта, наименование должно быть оборотом существительного.

С помощью интерфейсных дуг отображают различные объекты, в той или иной степени определяющие процессы, происходящие в системе. Такими объектами могут быть элементы реального мира (детали, вагоны, сотрудники и т.д.) или потоки данных и информации (документы, данные, инструкции и т.д.).

В зависимости от того, к какой из сторон подходит данная интерфейсная дуга, она носит название «входящей», «исходящей» или «управляющей». Кроме того, «источником» (началом) и «приемником» (концом) каждой функциональной дуги могут быть только функциональные блоки, при этом «источником» может быть только выходная сторона блока, а «приемником» любая из трех оставшихся.

Необходимо отметить, что любой функциональный блок по требованиям стандарта должен иметь, по крайней мере, одну управляющую интерфейсную дугу и одну исходящую. Это и понятно – каждый процесс должен происходить по каким-то правилам (отображаемым управляющей дугой) и должен выдавать некоторый результат (выходящая дуга), иначе его рассмотрение не имеет никакого смысла. Кроме того, необходимо понимать, что некоторый результат не может быть получен процессом без некоторого входа, так как любой процесс есть процесс преобразования входа в выход.

При построении SADT-диаграмм важно правильно отделять входящие интерфейсные дуги от управляющих, что часто бывает непросто, так как имеется сходство природы входящих и управляющих интерфейсных дуг. Однако, для систем одного класса всегда есть определенные разграничения. Например, в случае рассмотрения предприятий и организаций существуют пять основных видов объектов: материальные потоки (детали, товары, сырье и т.д.), финансовые потоки (наличные и безналичные, инвестиции и т.д.), потоки документов (коммерческие, финансовые и организационные документы), потоки информации (информация, данные о намерениях, устные распоряжения и т.д.) и ресурсы (сотрудники, станки, машины и т.д.). При этом в различных случаях входящими и исходящими интерфейсными дугами могут отображаться все виды объектов, управляющими только относящиеся к потокам документов и

информации, а дугами-механизмами только ресурсы. При этом управляющие потоки никогда не преобразуются в результате исполнения процесса.

Третьим основным понятием технологии SADT является понятие «***декомпозиция***». Принцип декомпозиции применяется при разбиении сложного процесса на составляющие его функции. При этом уровень детализации процесса определяется непосредственно разработчиком модели. Декомпозиция позволяет постепенно и структурированно представлять модель системы в виде иерархической структуры отдельных диаграмм, что делает ее менее перегруженной и легко усваиваемой.

Модель SADT всегда начинается с представления системы как единого целого – одного функционального блока с интерфейсными дугами, простирающимися за пределы рассматриваемой области. Такая диаграмма с одним функциональным блоком называется контекстной диаграммой, и обозначается идентификатором «A-0».

В пояснительном тексте к контекстной диаграмме должна быть указана ***цель*** построения диаграммы в виде краткого описания и зафиксирована ***точка зрения***. Определение и формализация цели разработки SADT-модели является крайне важным моментом. Фактически цель определяет соответствующие области в исследуемой системе, на которых необходимо фокусироваться в первую очередь. Например, если мы моделируем деятельность предприятия с целью построения в дальнейшем на базе этой модели информационной системы, то эта модель будет существенно отличаться от той, которую бы мы разрабатывали для того же самого предприятия, но уже с целью оптимизации логистических цепочек. Точка зрения определяет основное направление развития модели и уровень необходимой детализации. Четкое фиксирование точки зрения позволяет разгрузить модель, отказавшись от детализации и исследования отдельных элементов, не являющихся необходимыми, исходя из выбранной точки зрения на систему. Например, функциональные модели одного и того же предприятия с точек зрения главного технолога и финансового директора будут существенно различаться по направленности их детализации. Это связано с тем, что в конечном итоге, финансового директора не интересуют аспекты обработки сырья на производственных станках, а главному технологу ни к чему прорисованные схемы финансовых потоков. Правильный выбор точки зрения существенно сокращает временные затраты на построение конечной модели.

В процессе декомпозиции, функциональный блок, который в контекстной диаграмме отображает систему как единое целое, подвергается

детализации на другой диаграмме. Получившаяся диаграмма второго уровня содержит функциональные блоки, отображающие главные подфункции функционального блока контекстной диаграммы и называется дочерней по отношению к ней (каждый из функциональных блоков, принадлежащих дочерней диаграмме соответственно называется дочерним блоком). В свою очередь, функциональный блок-предок называется родительским блоком по отношению к дочерним блокам, а диаграмма, к которой он принадлежит – родительской диаграммой. Каждая из подфункций дочерней диаграммы может быть далее детализирована путем аналогичной декомпозиции соответствующего ей функционального блока. В каждом случае декомпозиции функционального блока все интерфейсные дуги, входящие в данный блок, или исходящие из него фиксируются на дочерней диаграмме. Этим достигается структурная целостность SADT-модели.

Часто бывают случаи, когда отдельные интерфейсные дуги не имеет смысла продолжать рассматривать в дочерних диаграммах ниже какого-то определенного уровня в иерархии, или наоборот – отдельные дуги не имеют практического смысла выше какого-то уровня. Например, интерфейсную дугу, изображающую «деталь» на входе в функциональный блок «Обработать на токарном станке» не имеет смысла отражать на диаграммах более высоких уровней – это будет только перегружать диаграммы и делать их сложными для восприятия. С другой стороны, случается необходимость избавиться от отдельных «концептуальных» интерфейсных дуг и не детализировать их глубже некоторого уровня. Для решения подобных задач в технологии SADT предусмотрено понятие «**туннелирование**». Обозначение «туннеля» в виде двух круглых скобок вокруг начала интерфейсной дуги обозначает, что эта дуга не была унаследована от функционального родительского блока и появилась (из «туннеля») только на этой диаграмме. В свою очередь, такое же обозначение вокруг конца (стрелки) интерфейсной дуги в непосредственной близи от блока-приёмника означает тот факт, что в дочерней по отношению к этому блоку диаграмме эта дуга отображаться и рассматриваться не будет. Чаще всего бывает, что отдельные объекты и соответствующие им интерфейсные дуги не рассматриваются на некоторых промежуточных уровнях иерархии – в таком случае, они сначала «погружаются в туннель», а затем, при необходимости «возвращаются из туннеля».

Последним из специфических понятий SADT является понятие «***глоссарий***». Для каждого из элементов SADT-диаграмм: функциональных блоков, интерфейсных дуг – существующий стандарт

подразумевает создание и поддержание набора соответствующих определений, ключевых слов, повествовательных изложений и т.д., которые характеризуют объект, отображенный данным элементом. Набор, называемый ***глоссарием***, является описанием сущности данного элемента. Глоссарий гармонично дополняет наглядный графический язык, снабжая диаграммы необходимой дополнительной информацией. Если технологию SADT рассматривать как аналог технологии DFD, то глоссарий будет выполняет роль словаря данных, используемого при построении DFD-диаграммы.

Обычно SADT-модели несут в себе сложную и концентрированную информацию, и для того, чтобы ограничить их перегруженность и сделать удобочитаемыми, в соответствующем стандарте приняты соответствующие ограничения сложности:

- ограничение количества функциональных блоков на диаграмме тремя-шестью. Верхний предел (шесть) заставляет разработчика использовать иерархии при описании сложных предметов, а нижний предел (три) гарантирует, что на соответствующей диаграмме достаточно деталей, чтобы оправдать ее создание;
- ограничение количества подходящих к одному функциональному блоку интерфейсных дуг четырьмя.

Разумеется, строго следовать этим ограничениям вовсе необязательно, однако, они являются весьма практичными в реальной работе.

Технология SADT содержит набор процедур, позволяющих разрабатывать и согласовывать модель большой группой людей, принадлежащих к разным областям деятельности моделируемой системы. Обычно процесс разработки является итеративным и состоит из следующих условных этапов [Верников, 2]:

- Создание черновика модели группой специалистов, относящихся к различным сферам деятельности предприятия. Эта группа в терминах SADT (IDEF0) называется авторами. Построение первоначальной модели является динамическим процессом, в течение которого авторы опрашивают компетентных лиц о структуре различных процессов. На основе имеющихся положений, документов и результатов опросов создается черновик модели.
- Распространение черновика для рассмотрения, согласований и комментариев. На этой стадии происходит обсуждение черновика модели с широким спектром компетентных лиц (в терминах SADT (IDEF0) – читателей) на предприятии. При этом каждая из диаграмм черновой модели письменно критикуется и комментируется, а

затем передается автору. Автор, в свою очередь, также письменно соглашается с критикой или отвергает её с изложением логики принятия решения и вновь возвращает откорректированный черновик для дальнейшего рассмотрения. Этот цикл продолжается до тех пор, пока авторы и читатели не придут к единому мнению.

- Официальное утверждение модели. Утверждение согласованной модели происходит руководителем рабочей группы в том случае, если у авторов модели и читателей отсутствуют разногласия по поводу ее адекватности. Окончательная модель представляет собой согласованное представление о предприятии (системе) с заданной точки зрения и для заданной цели.

Примеры контекстных SADT-диаграмм и их декомпозиции, подготовленный с применением инструментального пакета BPwin, представлен на рисунках 3.11–3. 15.

Наглядность графического языка SADT делает модель вполне читаемой и для лиц, которые не принимали участия в проекте ее создания, а также эффективной для проведения показов и презентаций. В дальнейшем, на базе построенной модели могут быть организованы новые проекты, нацеленные на производство изменений на предприятии (в системе).

Кроме недостатков DFD-диаграмм 1, 5 и 6 в литературе отмечаются следующие недостатки IDEF0-диаграмм:

1. Недостаточно выразительных средств для моделирования систем информационных. В результате данные диаграммы практически используется относительно редко (менее чем в 10% существующих CASE-средств).
2. Для создания динамических моделей требуется использование дополнительных специальных расширений или других средств, с которыми данные диаграммы плохо согласуются.
3. Принципиально ограниченное количество типов связей и типов отношений (взаимодействий) между блоками не позволяют гарантировать во всех случаях адекватность модели объекту и затрудняют понимание диаграмм.
4. Изображение функциональных связей каждого элемента в виде входа, управления, механизма (или ресурса) и выхода не обеспечивается никаким методом распределения связей в конкретных случаях по данным категориям. Результатом этого является представление, например, производственного подразделения как элемента, ресурсом которого изображаются люди, которые в нем работают, т.е. которые составляют, на самом деле, его части, а не входы.

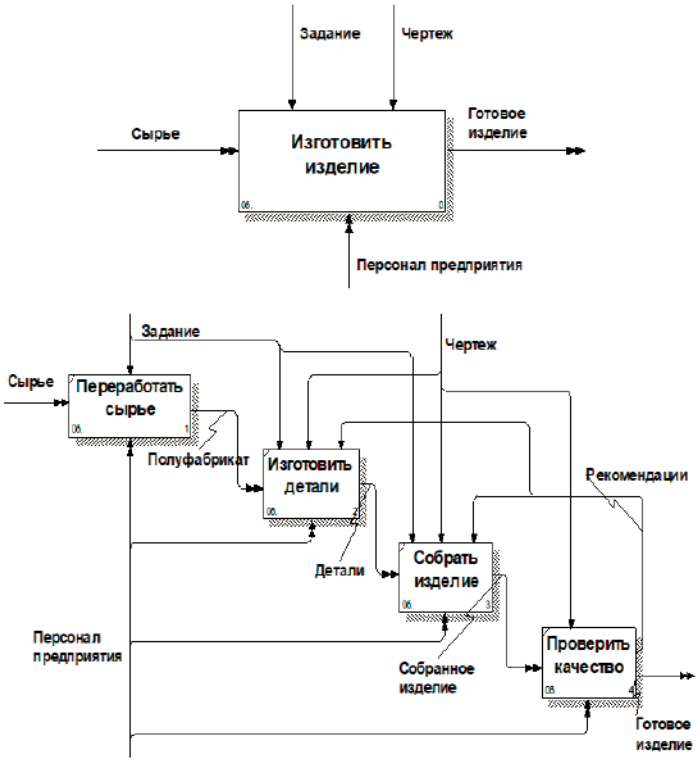


Рис. 3.11. Контекстная SADT-диаграмма производства и ее декомпозиция

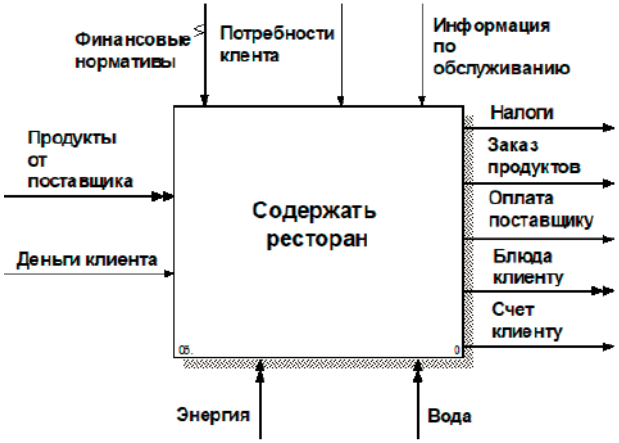


Рис. 3.12. Контекстная SADT-диаграмма ресторана

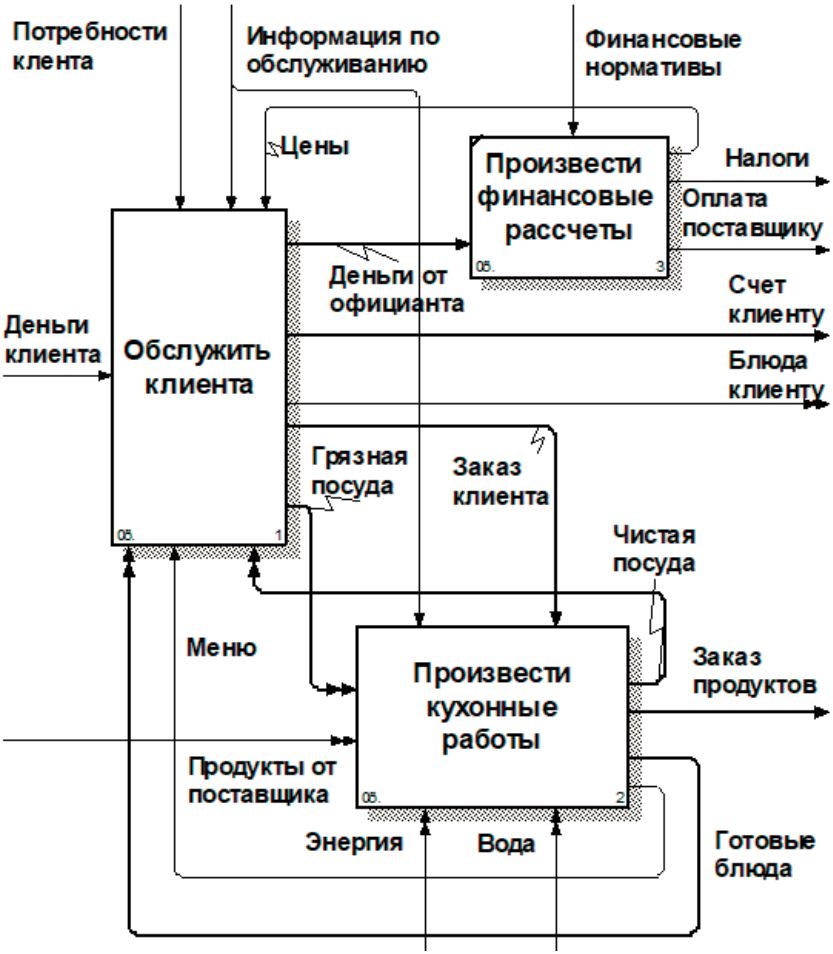


Рис. 3.13. SADT-диаграмма ресторана первого уровня

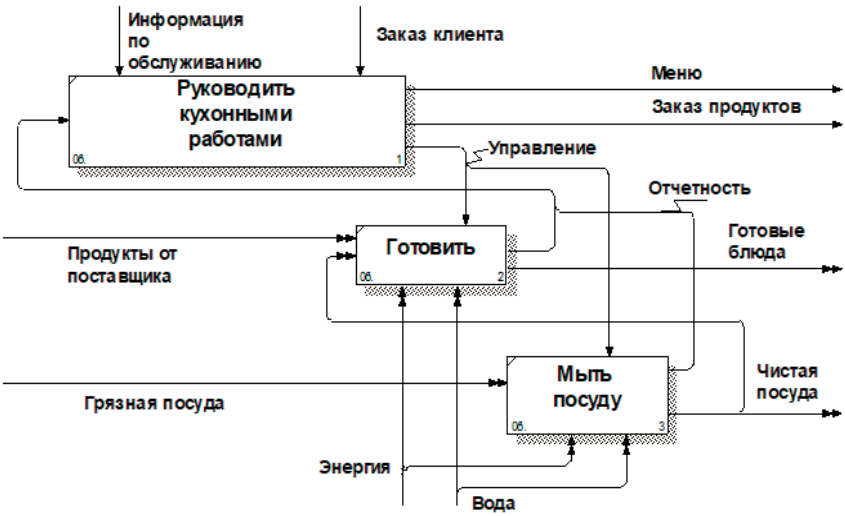


Рис. 3.14. SADT-диаграмма ресторана второго уровня (кухонные работы)

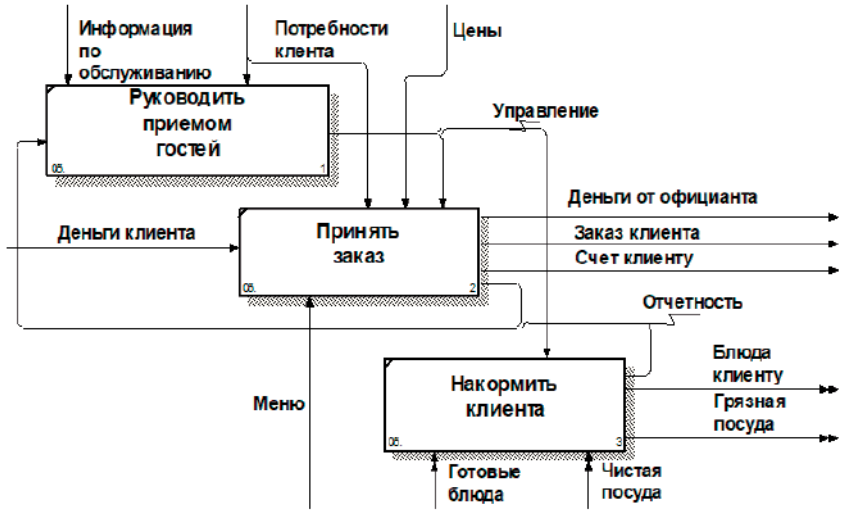


Рис. 3.15. SADT-диаграмма ресторана второго уровня (обслуживание клиентов)

1.3. Стандарт IDEF3

Стандарт ***IDEF3*** предназначен для документирования технологических процессов, происходящих на предприятии, и предоставляет инструментарий для наглядного исследования и моделирования их ***сценариев***.

Рассмотрим особенности данного стандарта, используя работу [Верников, 3].

Сценарием называется описание последовательности изменений свойств объекта, в рамках рассматриваемого процесса (например, описание последовательности этапов обработки детали в цеху и изменение её свойств после прохождения каждого этапа). Исполнение каждого сценария сопровождается соответствующим документооборотом, который состоит из двух основных потоков: документов, определяющих структуру и последовательность процесса (технологических указаний, описаний стандартов и т.д.), и документов, отображающих ход его выполнения (результатов тестов и экспертиз, отчетов о браке, и т.д.). Для эффективного управления любым процессом, необходимо иметь детальное представление о его сценарии и структуре сопутствующего документооборота. Средства документирования и моделирования IDEF3 позволяют выполнять следующие задачи:

- Документировать имеющиеся данные о технологии процесса, выявленные, скажем, в процессе опроса компетентных сотрудников, ответственных за организацию рассматриваемого процесса.
- Определять и анализировать точки влияния потоков сопутствующего документооборота на сценарий технологических процессов.
- Определять ситуации, в которых требуется принятие решения, влияющего на жизненный цикл процесса, например изменение конструктивных, технологических или эксплуатационных свойств конечного продукта.
- Содействовать принятию оптимальных решений при реорганизации технологических процессов.
- Разрабатывать имитационные модели технологических процессов, по принципу «Как будет если...».

Существуют два типа диаграмм в стандарте IDEF3, представляющих описание одного и того же сценария технологического процесса в разных ракурсах. Диаграммы, относящиеся к первому типу, называются ***диаграммами Описания Последовательности Этапов Процесса*** (***Process Flow Description Diagrams – PFDD***), а ко второму – ***диаграммами Состояния Объекта в Процессе его Трансформации*** (***Object State Transition Network – OSTN***).

Предположим, требуется описать процесс окраски детали в производственном цеху на предприятии. С помощью диаграмм PFDD (см. рис. 3.16 и [Верников, 3]) документируется последовательность и описание стадий обработки детали в рамках исследуемого технологического процесса. Диаграммы OSTN (см. рис. 3.17 и [Верников, 3]) используются для иллюстрации трансформаций детали, которые происходят на каждой стадии обработки.

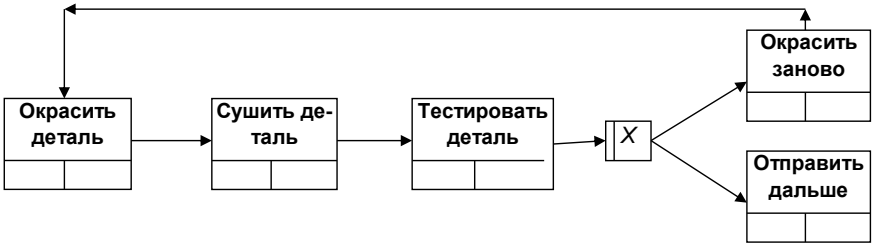


Рис. 3.16. Пример диаграммы PFDD

На рис. 3.16 изображена диаграмма PFDD, являющаяся графическим отображение сценария обработки детали. В целом, этот процесс состоит непосредственно из самой окраски, производимой на специальном оборудовании и этапа контроля ее качества, который определяет, нужно ли деталь окрасить заново (в случае несоответствия стандартам и выявления брака) или отправить ее на дальнейшую обработку.

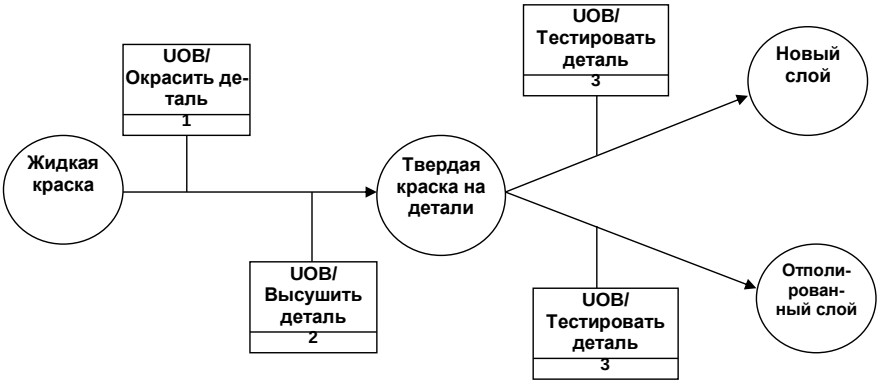


Рис. 3.17. Пример диаграммы OSTN

Прямоугольники на диаграмме PFDD называются функциональными элементами, единицами работы или элементами поведения (***Unit of Behavior – UOB***) и обозначают событие, стадию процесса или

принятие решения. Каждый UOB имеет свое имя, отображаемое в глагольном наклонении и уникальный номер. Стрелки являются отображением перемещения детали между UOB-блоками в ходе процесса. Они бывают следующих видов:

- Предшествование – сплошная одинарная стрелка, связывающая UOB в соответствии с последовательностью выполнения работ. Рисуется слева направо или сверху вниз.
- Отношение – пунктирная одинарная стрелка, использующаяся для изображения различного вида связей между UOB.
- Поток объектов – сплошная стрелка с двумя наконечниками, используемая для описания того факта, что объект, порождаемый одной работой, используется для выполнения другой.

Объект, обозначенный ***J1*** – называется ***перекрестком*** (***junction***). Перекрестки используются для отображения логики взаимодействия стрелок (потоков) при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (fan-in junction) и разветвления (fan-out junction) стрелок. Все перекрестки в PFDD диаграмме нумеруются, каждый номер имеет префикс «J». Классификация возможных типов перекрестков приведена в таблице 3.4.

Таблица 3.4

Возможные перекрестки в стандарте IDEF3.

Обозначение	***Наименование***	***Слияние стрелок (Fan-in Junction)***	***Разветвление стрелок (Fan-out Junction)***
	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
	Synchronous OR	Один или несколько предшествующих процессов завершаются одновременно	Один или несколько следующих процессов запускаются одновременно
	XOR Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Каждый функциональный блок UOB может иметь последовательность декомпозиций, и, следовательно, может быть детализирован с любой необходимой точностью. Под декомпозицией понимается представление каждого UOB с помощью отдельной IDEF3 диаграммы.

Например, можно декомпозировать UOB «Окрасить деталь», представив его отдельным процессом и построив для него свою PFDD диаграмму. При этом эта диаграмма будет называться дочерней, по отношению к изображенной на рисунке, а та, соответственно, родительской. Номера UOB дочерних диаграмм имеют сквозную нумерацию, т.е., если родительский UOB имеет номер «1», то блоки UOB на его декомпозиции будут соответственно иметь номера «1.1», «1.2» и т.д.

Если диаграммы PFDD представляет технологический процесс «с точки зрения наблюдателя», то другой класс диаграмм IDEF3 – OSTN позволяет рассматривать тот же самый процесс «с точки зрения объекта». Состояния объекта (в данном случае детали, см. рис. 3.17) и Изменение состояния являются ключевыми понятиями OSTN-диаграммы. Состояния объекта отображаются окружностями, а их изменения направленными одинарными стрелками. Каждая стрелка имеет ссылку на соответствующий функциональный блок UOB, в результате которого произошло отображаемое ею изменение состояния объекта. Таким образом, OSTN-диаграмма представляет собой аналог STD-диаграммы (переходов состояний) технологии 3VM.

С точки зрения примера, рассматриваемого в данном разделе (ресторана), стандарт IDEF3 является удобным средством анализа и документирования процессов (сценариев) приема гостей в различных ситуациях (например: ресторан «fast food» или вечерний ресторан; обычный обед/ужин, заказанное мероприятие, праздничный ужин, завтрак), а также технологических процессов приготовления различных блюд. На рисунках 3.18 и 3.19 представлен пример сценария приема гостей в вечернем ресторане при обслуживании обычного обеда/ужина (подготовлен с помощью BPwin (IDEF3)).

2. Объектно-ориентированное моделирование

2.1. Язык UML

Объектно-ориентированное моделирование, как и методы системно-структурного моделирования и анализа, предполагает использование некоторой нормативной системы, т.е. языка, состоящего из набора символов, имеющих определенное значение (семантику), и правил манипулирования ими (синтаксиса).

В настоящее время благодаря усилиям концерна ***Object Management Group*** (***OMG***) создан единый стандарт языка объектного моделирования – Unified Modelling Language (UML).

Язык UML – это, в первую очередь, стандартное средство для составления «чертежей» программного обеспечения (ПО). Однако, сфера его применения не ограничивается моделированием программ. Он предназначен для визуализации, специфицирования, конструирования и документирования различных аспектов анализируемых и проектируемых систем произвольной природы. При этом, в дальнейшем, обеспечивается возможность компьютерного моделирования этих систем с помощью объектно-ориентированных языков программирования.

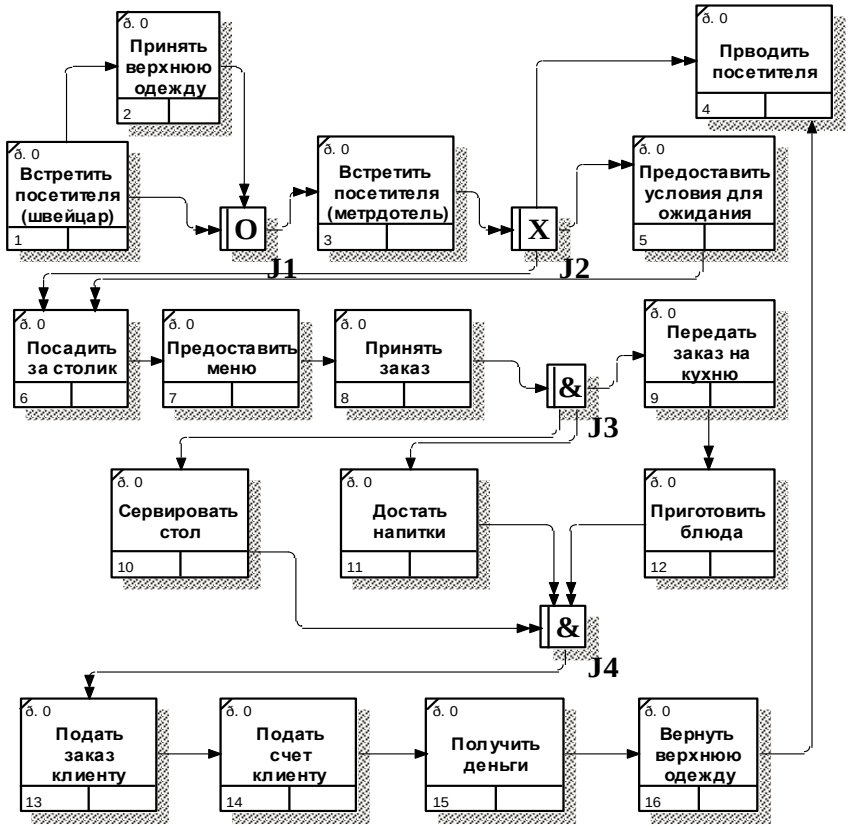


Рис. 3.18. Пример PFDD-диаграммы сценария приема гостей в вечернем ресторане

Возможности этого языка учитывать общесистемные закономерности представлены в Приложении №1.

Нормативная система языка UML включает два вида символов: ***сущности*** и ***отношения***, а также правила создания комбинаций из этих символов, т.е. ***диаграммы***. Кроме того, существуют ***механизмы расширения*** языка для уточнения семантики символов сущностей и отношений при моделировании конкретной предметной области.

Рассмотрим основные элементы нормативной системы языка UML, используя работы [Буч, 2000; 2010; Фаулер, 1999; Вендров, 2000; Кватрани, 2001; Ларман, 2001; Леоненков, 2001; Трофимов, 2001; Мацяшек, 2002; Франс, 2006].

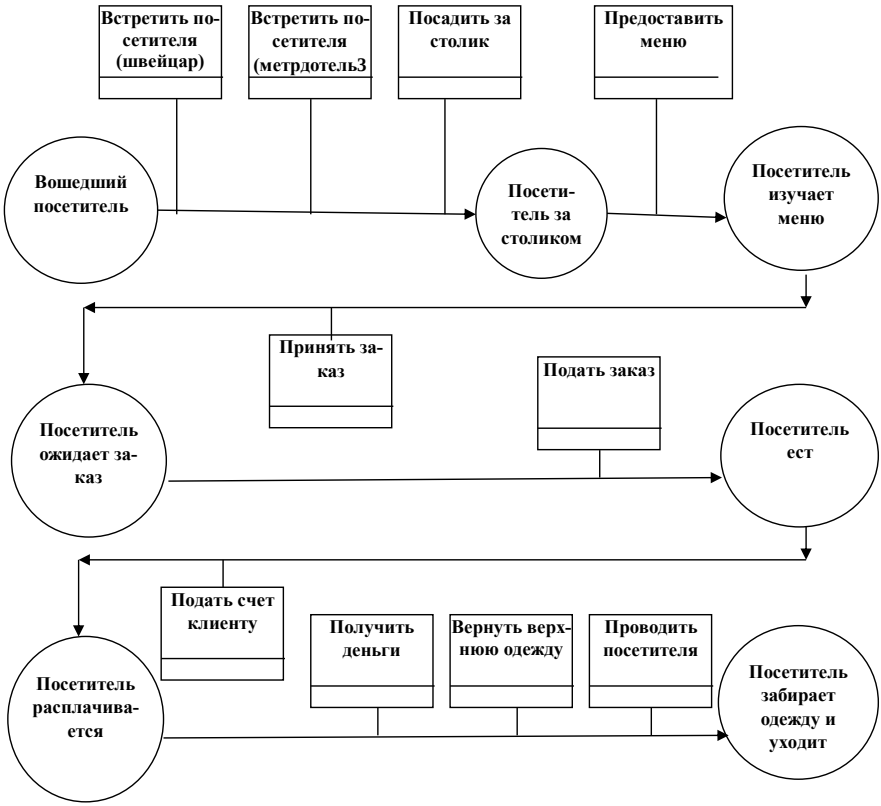


Рис. 3.19. Пример OSTN-диаграммы сценария приема гостей в вечернем ресторане

2.1.1. Сущности: структурные; поведенческие; группирующие; аннотационные

Сущности – это символы, являющиеся основными элементами объектной модели. Принято использовать четыре вида сущностей при построении моделей: ***структурные, поведенческие, группирующие, аннотационные***.

Структурные сущности.

Структурные сущности (Structural things) представляют собой символы статических частей объектной модели, соответствующие концептуальным или физическим элементам системы.

Класс (***Class***) – это описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой (рис. 3.20).

При этом атрибут – это именованное свойство всех экземпляров данного класса. Он имеет тип, обычно указываемый в классе, и значение, которое указывается в экземпляре класса. Операция – это описание поведения всех экземпляров данного класса, или функция, которую могут выполнять его экземпляры, или услуга, которую могут у них запросить.

Экземпляр (***Instance***) – конкретная реализация абстракции (класса, прецедента, компонента и т.д.) или объект, графически представляемый как класс с подчеркнутым именем (рис. 3.20), после которого может следовать двоеточие (:) и имя класса, от которого производится данный объект. Методы объекта представляют собой реализации соответствующих операций класса и, как правило, не указываются.

Активный класс (***Active class***) – класс, объекты которого могут инициировать управляющее воздействие, и вовлечены в один или несколько процессов или потоков. Изображается также как класс, но толстыми линиями.

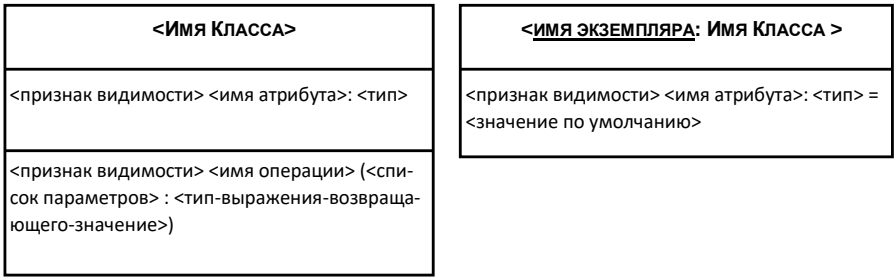


Рис. 3.20. Графическое изображение класса и экземпляра

Признак видимости может принимать одно из трех значений:

- + – открытый (public) – может использовать любой класс или объект;
- # – защищенный (protected) – может использовать любой потомок класса;
- - – закрытый (private) – может использовать только данный класс.

Интерфейс (***Interface***) – совокупность операций, которые определяют набор услуг, предоставляемых классом или компонентом, т.е. описание видимого извне поведения элемента (рис. 3.21). Определяет спецификации операций (сигнатуры), но не их реализацию и присоединяется к реализующему его классу или компоненту.

Кооперация (***Collaboration***) – представляет взаимодействия элементов, которые, работая совместно, производят некоторый кооперативный эффект, не сводящийся к простой сумме слагаемых (рис. 3.21).

Прецедент / Вариант использования (***Use case***) – описание последовательности выполняемых системой действий, в результате которых образуется наблюдаемый результат, значимый для какого-нибудь определенного актора (рис. 3.21).

Рис. 3.21. Графическое изображение интерфейса, кооперации и прецедента

Компонент (***Component***) – физически заменяемая часть системы, соответствующая некоторому набору интерфейсов и обеспечивающая их реализацию (рис. 3.22). Как правило, это физическая упаковка логических элементов (файлы), таких как классы, интерфейсы и кооперации.

Узел (***Node***) – элемент реальной (физической) системы, существующей во время функционирования программного комплекса и представляющий собой вычислительные ресурсы (объем памяти, скорость обработки) (рис. 3.22).

Рис. 3.22. Графическое изображение компонента и узла

Поведенческие сущности.

Поведенческие сущности (Behavioral things) представляют собой символы динамических составляющих объектной модели.

Взаимодействие (***Interaction***) – обмен сообщениями между объектами для достижения определенной цели (рис. 3.23).

Состояние (***Statechart***) – алгоритм поведения, определяющий последовательность состояний, через которые объект или взаимодействие проходят на протяжении своего жизненного цикла в ответ на различные события (рис. 3.23).

Активность (***Activity***) – алгоритм поведения, определяющий последовательность процессов, осуществляемых объектом или происходящих с объектом на протяжении его жизненного цикла (рис. 3.23).

Рис. 3.23. Графическое изображение сообщения, состояния и активности

Группирующие сущности.

Группирующие сущности (Grouping things) представляют собой символы, организующие различные компоненты объектной модели. Это блоки, на которые можно разложить модель.

Пакет (***Package***) – механизм организации элементов (структурных, поведенческих и группирующих сущностей) модели в группы (рис. 3.24). В отличие от компонентов носят чисто концептуальный характер.

Аннотационные сущности.

Аннотационные сущности (Summary things) представляют собой пояснительные символы объектной модели.

Примечание (***Note***) – символ для изображения комментариев или ограничений, присоединенных к элементу или группе элементов (рис. 3.24).

Рис. 3.24. Графическое изображение пакета и примечания

2.1.2. Отношения

Отношения – это символы, связывающие различные сущности. В языке определено четыре вида отношений: ***обобщение***, ***ассоциация***, ***зависимость*** и ***реализация***.

Обобщение (***Generalization***) – отношение (потомок-родитель), при котором специализированный элемент (потомок) может быть подставлен вместо обобщенного элемента (родителя) (рис. 3.25). Потомок наследует структуру и поведение своего родителя.

Ассоциация (***Association***) – отношение связи (соединения) между объектами (рис. 3.25). На графическом изображении ассоциации могут присутствовать дополнительные обозначения кратности, ролей или меток.

Агрегирование – разновидность ассоциации, т.е. ассоциация между целым и его частями (рис. 3.25).

Композиция – такое отношение агрегирования, при котором часть является неотъемлемой составляющей целого (рис. 3.25).

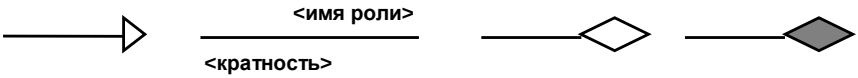


Рис. 3.25. Графическое изображение обобщения, ассоциации, агрегирования и композиции

Зависимость (***Dependency***) – отношение между двумя сущностями, при котором изменение одной из них, независимой, может повлиять на семантику другой, зависимой (см. рис. 3.26).

Реализация (***Realization***) – отношение между элементами, при котором один элемент определяет «контракт», а другой гарантирует его выполнение (см. рис. 3.26). Используется между интерфейсами и реализующими их классами или компонентами, а также между прецедентами и реализующими их кооперациями.

Рис. 3.26. Графическое изображение зависимости и реализации

2.1.3. Диаграммы

Диаграммы – это совокупности правил соединения различных символов языка моделирования. Диаграмма представляет собой связанный граф, вершинами которого являются сущности, а ребрами –

отношения. Любая диаграмма представляет собой модель некоторого аспекта разрабатываемой системы. Предусмотрены следующие виды диаграмм (правил соединения символов):

Диаграмма вариантов использования/прецедентов (***Use case diagram***) – диаграмма поведения, показывающая функции моделируемой системы и ее связи с другими системами, т.е., в случае разработки программного обеспечения, требования пользователей. Диаграмма состоит из множества прецедентов, классов (акторов: пользователей или любых других внешних систем) и отношений между ними (рис. 3.27).

Проектирование системы осуществляется путем реализации на каждой итерации одного или нескольких вариантов использования. Данная диаграмма является основой для моделирования поведения и тестирования системы.

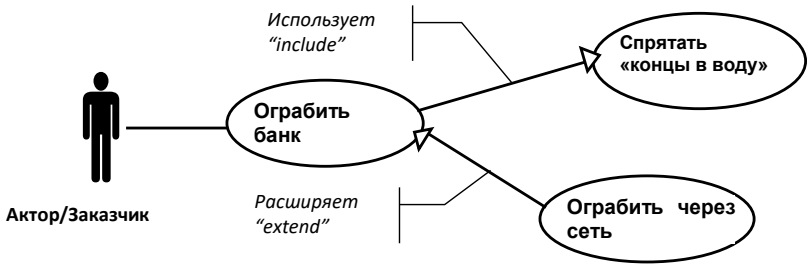


Рис. 3.27. Диаграмма вариантов использования

На данной диаграмме показаны следующие виды отношения между прецедентами: ***расширение*** и ***использование***.

Связь типа «***расширение***» применяется в тех случаях, когда один вариант использования подобен другому, но несет несколько другую нагрузку и, в связи с этим, необходимо описать изменение в обычном поведении системы.

Связь типа «***использование***» применяется в тех случаях, когда имеется какой-либо фрагмент поведения системы, который повторяется более чем в одном варианте использования и нет смысла копировать его описание.

Диаграмма классов (***Class diagram***) – структурная диаграмма, показывающая структуру классов предметной области. Диаграмма состоит из классов, интерфейсов, кооперации и отношений между ними (рис. 3.28). Абстрактные классы, от которых не производятся экземпляры, помечаются курсивом.

Классы на диаграмме отражают понятия и представления пользователей о соответствующей предметной области, т.е. названная диаграмма является ее концептуальной моделью.

Диаграмма объектов (***Object diagram***) – структурная диаграмма, показывающая объектную модель системы, состоящую из множества экземпляров классов (объектов) и отношений между ними.

Рис. 3.28. Диаграмма классов

Диаграмма взаимодействия – диаграмма поведения, показывающая взаимодействие нескольких объектов, например, в рамках одного варианта использования. Взаимодействия изображаются с помощью ***диаграммы последовательностей*** (***Sequence diagram***), подчеркивающей временную последовательность событий (рис. 3.29), или ***диаграммы кооперации*** (***Collaboration diagram***), подчеркивающей структурную организацию объектов, посылающих и принимающих сообщения (рис. 3.30).

Рис. 3.29. Диаграмма последовательности

Диаграмма состояний (***Statechart diagram***) – диаграмма поведения, показывает поведение одного объекта в нескольких вариантах использования. Диаграмма представляет автомат, включающий в себя состояния, переходы, события и виды действий. Данная диаграмма акцентирует внимание на поведении объекта, зависящем от последовательности событий, и особенно важна при описании поведения классов, интерфейсов и кооперации.

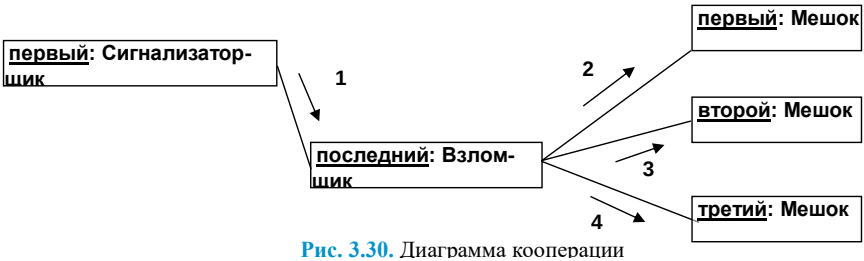


Рис. 3.30. Диаграмма кооперации

Сообщения: 1 – открыть сейф; 2 – деньги в первый мешок;
3 – деньги во второй мешок; 4 – деньги в третий мешок.

Диаграмма деятельности (***Activity diagram***) – диаграмма поведения, показывающая поведение разрабатываемой системы вместе с управляющей структурой. Диаграмма представляет автомат и подчеркивает переходы потока управления от одной деятельности к другой (рис. 3.31). Может отображать несколько объектов в нескольких вариантах использования или реализацию метода. Позволяет определять параллельные процессы.

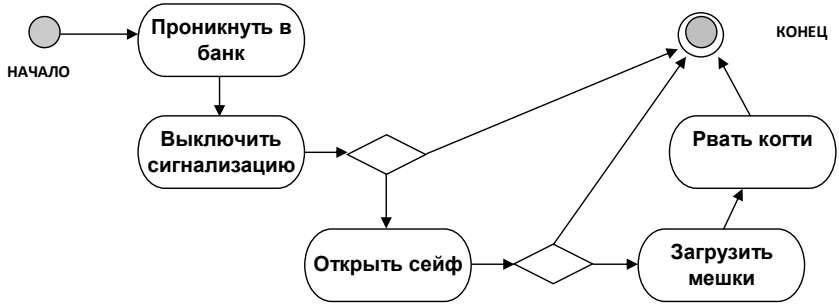


Рис. 3.31. Диаграмма деятельности (активности)

Диаграмма компонентов (***Component diagram***) – диаграмма, показывающая организацию совокупности компонент (файлов) и их зависимости.

Диаграмма развертывания (***Deployment diagram***) – диаграмма, показывающая узлы и отношения между ними (физическое размещение компонент в аппаратных средствах).

Применение языка UML существенно облегчается за счет следующих механизмов:

- ***спецификаций*** (***specifications***) – текстовых описаний синтаксиса и семантики соответствующих элементов модели (диаграммы);
- ***принятых делений*** (***common divisions***) – в соответствии с дихотомией «класс/объект» и дихотомией «интерфейс/реализация»;
- ***механизмов расширения*** (***extensibility mechanisms***) – обеспечивающих расширение синтаксиса и семантики языка.
- Предусмотрены следующие механизмы расширения языка:
- ***стереотипы*** (***stereotypes***), позволяющие расширять словарь языка и представлять на основе существующих элементов новые специфичные для конкретных проблем элементы модели;
- ***помеченные значения*** (***tagged values***), позволяющие расширять свойства элементов языка и представлять новые атрибуты и новую информацию в спецификацию элемента;
- ***ограничения*** (***constraint***), позволяющие расширять семантику элементов языка и определить новые или изменить старые правила.

UML не зависит от процесса его использования или метода моделирования. Однако, лучше всего он поддерживает ***Рациональный Унифицированный Процесс*** (***Rational Unified Process – RUP***) изготовления программных продуктов.

В общих чертах процесс анализа и моделирования системы в терминах UML представляет собой последовательность следующих шагов:

- Анализ внешних по отношению к данной систем и формулирование требований к моделируемой системе с их точки зрения. Результаты этого шага представляются в виде диаграммы прецедентов.
- Уточнение потоков событий в прецедентах, например, в виде диаграммы деятельности или текстового документа.
- Определение типов объектов (классов) и их свойств, с помощью которых можно будет реализовать деятельность в прецедентах. Результаты этого шага представляются в виде диаграммы классов.
- Декомпозиция моделируемой системы и представление ее модели в виде диаграммы кооперации (взаимодействия объектов).

Консорциум OMG начал разрабатывать UML 2.0, чтобы преодолеть существенные недостатки ранних версий. В число проблем, которые пытаются решить архитекторы стандарта UML 2.0, входит проблема разбухания ранних версий и недостатки в определении семантики. В процессе работы над новым стандартом возникла необходимость включить в него поддержку ***разработки на базе моделей*** (***Model-Driven Development – MDD***) и, в частности, подхода к такой разработке с позиций ***архитектуры на базе моделей*** (***Model-Driven Architecture – MDA***). В результате стандартизации языка моделирования общественными усилиями был разработан очень громоздкий и сложный вариант стандарта. По мнению известных экспертов [Франс, 2006, с. 34] «Кажется, конечный результат полностью противоречит благим намерениям, породившим радикальный пересмотр стандарта. Многочисленные концепции моделирования, плохо определенная семантика и облегченные механизмы расширения UML 2.0 затрудняют его изучение и применение в MDD» ... «Язык UML 2.0 в некоторых отношениях превосходит предыдущие версии, но его громоздкость и сложность могут создать проблемы для пользователей, разработчиков инструментальных средств и рабочих групп OMG, развивающих этот стандарт».

2.2. Требования к объектному моделированию организационных систем

Рассмотрим требования к моделированию бизнес-систем и бизнес-процессов, описанные, например, в [Ойхман, 1997].

Главной задачей, решаемой в рамках моделирования бизнеса, является анализ окружающей предприятие среды и того, как предприятие взаимодействует с этой средой. Под окружающей средой, в данном случае, понимается все, с чем предприятие взаимодействует в ходе выполнения своих бизнес-процессов, т.е. клиентов, партнеров, субподрядчиков, конкурентов и т.д. Полная модель бизнеса показывает работникам всех уровней и подразделений предприятия, что должно быть сделано, когда и как именно.

Наиболее известная модель бизнеса – организационная (иерархическая) структура предприятия. Эта модель совершенно недостаточна для того, чтобы проанализировать, спроектировать и (или) изменить предприятие (оптимизировать деловую активность, решить проблемы). Вместо этого нужны модели, показывающие предприятие в

контексте его клиентов, поставщиков, партнеров и т.д., т.е. модели, которые представляют бизнес-процессы предприятия и то, как оно производит товары и услуги для внешнего мира (рынка).

Методы и средства, используемые для моделирования бизнеса, должны обеспечивать [Ойхман, 1997]:

- Визуализацию образа существующего и будущего предприятия и окружающего его мира (рынка).
- Возможность работы с альтернативными архитектурами бизнеса и моделирования их воздействие на деятельность предприятия.
- Описание продукции предприятия в контексте того, как, когда и в ходе какого процесса она обрабатывается.
- Адаптацию выбранного архитектурного решения к существующему предприятию для его последующего внедрения.
- Описание реализации конечного проекта с учетом как человеческих, так и технических ресурсов.
- Представление реконструированного предприятия таким образом, чтобы каждый участник понял новую организацию работ, свои новые задачи и способы их выполнения, т.е. модель должна быть понятна персоналу без длительного обучения и серьезного вмешательства в его работу.

Моделирование бизнеса принято осуществлять в двух направлениях. Во-первых, в направлении обратного инжиниринга, что позволяет понять, как в настоящее время работает предприятие. Обычно это делается для того, чтобы получить прочную основу для кардинального улучшения различных аспектов предприятия в будущем или тогда, когда требуется понять и объяснить, как функционирует предприятие или некоторый его процесс. Во-вторых, в направлении прямого инжиниринга, что обеспечивает описание нового предприятия. Эта работа начинается с формулирования целей и образа (vision) будущего предприятия. После этого набрасываются различные сценарии. Для каждого сценария создается общее описание процесса, включающее заказчиков, поставщиков и т.д., а также сам процесс. Далее проводится имитационное моделирование различных процессов при помощи деловой игры или компьютерной модели. Наконец, выбранная альтернатива реализуется.

В ***бизнес-модели*** необходимо уметь выразить [Ойхман, 1997]:

1. ***Процессы***, т.е. структурированный, измеряемый набор действий, осуществляемый для того, чтобы произвести определенный выход для конкретного клиента на рынке. При моделировании бизнеса

модели процессов должны показывать, как клиент может использовать бизнес, и включать в себя полный поток событий в системе, описывающий, как клиент начинает, ведет и завершает бизнес-процесс. Язык моделирования должен описывать различные типы задач или внутренних процессов, из которых состоит почти каждый бизнес-процесс, а также способ взаимодействия этих внутренних процессов при обслуживании клиента.

2. ***Ресурсы***, т.е. как внутренний процесс реализуется при помощи человеческих или технических ресурсов и откуда эти ресурсы будут взяты. Особенно важно уметь показать, как процесс может поддерживаться информационной системой. Должна существовать согласованность между внутренним процессом в организации бизнеса и требованиями, предъявляемыми к поддерживающей бизнес информационной системе.

3. ***Продукцию***. Понятия «товар» и «услуга» или их обобщение – понятие «продукция». Продукция должна иметь характеристики, которые описывают, что с ней можно делать.

4. ***Потоки событий***. Необходимо описывать входные данные из внешнего мира; действия (операции) процесса, которые он производит над исходными данными; потребляемые ресурсы; решения, которые будут приняты, и выход (результат), который процесс отправит во внешний мир. Так как «вход» и «выход» – это способы общения процесса с клиентом, они должны быть ориентированы на клиента.

5. Наконец, существуют ***общие требования к моделированию бизнеса*** в целом. При моделировании предприятия обязательно необходимо иметь в виду, что бизнес будет подвержен изменениям. Следовательно, важно, чтобы язык был гибким. Должна существовать возможность менять процессы, но при этом производить продукцию для клиентов бизнеса. Необходимо также иметь возможность адаптировать процессы к новым ситуациям. Например, если предприятие открывает филиалы, должна быть предусмотрена возможность специализировать процесс продаж так, чтобы он работал и в филиале.

При разработке модели бизнеса как в ходе прямого, так и в ходе обратного инжиниринга, согласно рассматриваемой методологии, рекомендуется создавать две модели организации: ***внешнюю*** и ***внутреннюю***.

2.2.1. Внешняя модель бизнес-системы

Внешняя модель – прецедент-модель (***П-модель***). Она описывает бизнес и его окружение, предприятие в целом и его внешний

мир (сегмент рынка), а также процессы, которые удовлетворяют интересы клиентов и интересы вне предприятия. Процессы моделируются при помощи прецедентов (вариантов использования), а окружение моделируется при помощи так называемых действующих лиц или ***субъектов***.

П-модель показывает, как внешнее окружение взаимодействует с бизнесом, т.е. как отдельные субъекты общаются с бизнесом посредством прецедентов. П-модель описывает бизнес так, как он виден извне, т.е. так, как он воспринимается теми, кто хочет его использовать. Следовательно, структуры внутри бизнеса, которые невидимы для субъектов, не следует описывать в П-модели. П-модель представляет собой, по сути дела, диаграмму прецедентов UML. Чтобы создать корректную модель бизнеса, важно очертить его окружение, в котором присутствуют клиенты, и именно они накладывают на бизнес наиболее существенные требования.

Субъект обозначает роль, которую кто-то или что-то может играть по отношению к бизнесу. В рассматриваемых моделях окружение представлено субъектами. Субъекты обозначают все то в окружении, что взаимодействует с бизнесом, и подлежит моделированию. Например, это клиенты, поставщики или партнеры. Субъектов можно разделить на две группы: человеческие и технические. Например, компьютерная система другой компании может быть представлена как технический субъект. С другой стороны, работников моделируемого предприятия или его технику не следует рассматривать как субъекты, так как они являются неотъемлемой частью самого бизнеса, т.е. они представляют собой ресурсы, которые используются для выполнения задач системой.

Важно понимать, что субъект представляет собой абстракцию кого-либо или чего-либо, использующего бизнес. Субъект может представлять различные виды явлений в окружении предприятия. Не следует путать реальных людей с субъектами. Реальная личность, в отличие от субъекта, может играть несколько ролей в бизнес-системе: конкретный человек может быть и посетителем, и одним из поставщиков ресторана.

Прецедент, в данном случае – это последовательность ***транзакций*** в системе, выполняемых для получения ***измеримой потребительской ценности*** для некоторого ***индивидуального субъекта*** бизнес-системы.

Индивидуальный субъект. Это понятие упрощает нахождение правильных прецедентов, т.е. позволяет избегать слишком сложных

прецедентов. Рассмотрение прецедента следует начинать с индивидуальных субъектов – с экземпляров действующих лиц. Например, при моделировании предприятия, специализирующегося на продаже некоторой продукции, необходимо осознавать, что субъект, названный «клиент», на самом деле представляет трех различных клиентов. Во-первых, простого потребителя продукции (каких много); во-вторых, покупателя продукции, т.е. кого-то, кто разбирается в закупках, но не обязательно знает, для чего используется продукция; и, в-третьих, того, кто может компетентно судить о продукции в сравнении с конкурирующими продуктами. Каждый из этих случаев требует отдельного прецедента, так как субъекты играют в них различные роли по отношению к системе.

Измеримая потребительская ценность. Эти слова служат ключом для выбора не слишком детального уровня прецедента. Должна быть предусмотрена возможность оценить эффективность прецедента в терминах цены или трудоемкости. Например, обращение в банк за кредитом имеет ценность для клиента банка.

Транзакция – это неделимое множество действий, которые или выполняются все целиком, или не выполняются вообще. Она запускается при получении системой стимула от субъекта или при наступлении определенного момента времени. Транзакция состоит из набора действий, решений и передачи стимулов некоторому субъекту (субъектам).

Следует различать ***класс прецедентов*** и ***экземпляр прецедента***. Класс прецедентов определяется его описанием, а экземпляр прецедента – действиями, которые выполняются, когда поток событий, соответствующий описанию прецедента, проходит через систему. Класс прецедентов может содержать несколько альтернативных путей через систему, но экземпляр следует только одному из них.

При описании модели нужно иметь возможность говорить, как об общих характеристиках некоторого типа субъектов, так и об уникальных характеристиках конкретного субъекта. Другими словами, надо иметь возможность различать класс субъектов и экземпляр субъекта. В модели конкретные люди являются не классами, а экземплярами некоторого класса. Субъект (класс) представляет собой роль, которую реальные люди могут играть. Один человек может играть несколько ролей, т.е. реализовывать несколько классов. Субъекты взаимодействуют с системой, посылая стимулы. Чтобы лучше понять субъект, необходимо знать, в каком прецеденте он участвует. В П-модели это указывается отношениями между субъектом и прецедентом.

Чтобы придать П-модели большую глубину, составляются детальные описания каждого прецедента. Следует не просто описывать поток событий в прецеденте, но и то, как он взаимодействуют с окружением, т.е. с субъектами. Таким образом, прецедент это такая «машина состояний-событий», в которой состояние экземпляра прецедента представляет, какие стимулы могут быть получены следующими прецедентами и какие стимулы могут вызвать переход в другое состояние, после завершения транзакции. Экземпляр прецедента, будучи инициированным, может пройти через некоторое количество состояний до того, как он завершится.

Цель П-модели – внешнее представление системы. Это означает, что модель предназначена в первую очередь для заказчиков и пользователей системы, а не для тех, кто ее реализует. Учитывая вышеизложенное, необходимо помнить, что надо и чего не надо показывать в П-модели [Ойхман, 1997]:

- В модели не показывается коммуникация между потоками событий. Таким образом, экземпляры прецедента не могут общаться друг с другом, иначе пришлось бы описывать интерфейс между ними.
- Экземпляры одного или различных классов прецедентов, очевидно, будут влиять в бизнесе друг на друга. Конечно, такие отношения очень важны, но они показывают детали внутренней работы бизнеса, не интересные людям, для которых предназначена П-модель. Следовательно, эти влияния будут показаны во внутренней модели.
- В П-модели не показывается параллельность хода событий. Предполагается, что прецедент интерпретируется как один экземпляр и одна транзакция в каждый промежуток времени. Следовательно, транзакции прецедентов описываются так, как если бы они были неделимыми и последовательными.
- В соответствии со всем сказанным выше, в П-модели показываются только отношения классов.

П-модель – хорошее средство как для описания требований к бизнесу, так и для представления наглядной картины того, что бизнес выполняет. Тем не менее, П-модель не дает ясной картины о внутреннем устройстве бизнеса. Она ничего не говорит о том, как различные виды деятельности реализуют бизнес-процессы, как эти виды деятельности связываются вместе в цепочки процессов, какой вид ресурсов должен использоваться для реализации того или иного вида деятельности и т.д.

П-модель иллюстрирует функции бизнеса, его окружение и бизнес-процессы, которые он предлагает внешнему миру. Для более

полного понимания бизнеса требуется описание его более полное чем то, которое дается П-моделью.

2.2.2. Внутренняя модель бизнес-системы

Внутренняя модель – объект-модель (***О-модель***). Она описывает, как строится каждый бизнес-процесс предприятия из различных рабочих задач (внутренних процессов) и какие ресурсы он использует. Внутренняя модель использует объекты, соответствующие рабочим задачам, и объекты, соответствующие предметам бизнеса, например, продукцию. Если прецеденты выражают, что бизнес должен делать, то внутренняя модель описывает, как бизнес работает. Таким образом, П-модель – есть «что модель», а О-модель – «как модель», представляющая собой UML-диаграммы классов и взаимодействия.

Используются два вида внутренних моделей: идеальные и реальные. ***Идеальная О-модель*** – это модель, не учитывающая, как бизнес должен реализоваться на практике. Такая модель не учитывает, например, что предприятие (фирма) географически распределена на несколько филиалов. ***Реальная О-модель*** учитывает эти факторы. Она учитывает, что в настоящее время предприятие не располагает персоналом, имеющим уровень компетенции, предполагаемый идеальной моделью. Во многих случаях достаточно построить идеальную модель и предоставить предприятию возможность решать, как следует работать, чтобы приблизиться к реальной модели.

При описании ***О-модели*** объекты представляют участников процессов и различного рода сущности (продукция, предметы, задачи), используемые в ходе их выполнения. Следует различать класс объектов и экземпляр класса (объект). Имя, которое дается классу объектов, должно как можно более ясно выражать суть свойственную экземплярам этого класса, т.е. объектам. Классам часто дают имена, состоящие из нескольких слов (как правило, существительных), так как ясность важнее, чем краткость.

Объект может соответствовать задачам, продукции или сущностям в бизнесе. Задачи могут быть подразделены на два типа: те, что обеспечивают взаимодействие субъектов с бизнесом, и те, которые являются чисто внутренними. Удобно определить различные типы объектов, для того чтобы сделать более ясными задачи, которые они выполняют в модели. Например, в работе [Ойхман, 1997] предлагает различать следующие типы объектов: ***объекты-сущности***, ***управляющие объекты*** и ***интерфейсные объекты***. При этом интерфейсные и управляющие объекты представляют задачи, а не типы ресурсов. С другой стороны, эти задачи

выполняются людьми, относящимися к определенной категории ресурсов. Часто, однако, несколько задач помещаются вместе в один объект, который реализуется одним конкретным экземпляром ресурса.

Интерфейсные объекты представляют в бизнесе операции, каждая из которых должна выполняться одним и тем же ресурсом. Эта задача включает взаимодействие с окружением бизнеса. Следовательно, к коммуникабельности людей, выполняющих эту задачу, предъявляются определенные требования.

Интерфейсный объект может участвовать в нескольких прецедентах. Часто объекты этого типа несут ответственность за координацию процесса в той его части, которая касается непосредственного контакта с клиентами. Каждый конкретный клиент должен иметь как можно меньше контактов с предприятием. Это, впрочем, не противоречит теориям, которые рекомендуют работникам предприятия быть ближе к клиенту, а просто означает, что в интересах клиента его контакты с предприятием должны быть как можно проще и реже. Можно сказать, что часть бизнеса, имеющая непосредственный контакт с внешним миром, видима как интерфейсные объекты, в то время как объекты-сущности и управляющие объекты более независимы от окружения.

Управляющие объекты, как и интерфейсные объекты, представляют операции в бизнесе. Различие заключается в том, что эти операции не имеют непосредственного контакта с окружением бизнеса. Название «управляющий объект» используется потому, что эти объекты активны, они управляют или принимают участие в потоке управления при обработке продукции. Следовательно, экземпляр управляющего объекта часто имеет то же время жизни, что и экземпляр прецедента, в котором он участвует. Типичные примеры управляющих объектов на предприятии – это ***Разработчик*** и ***Менеджер Проекта***.

Объекты-сущности представляют такие сущности, как продукция и предметы, которые обрабатываются бизнесом. Объект-сущность участвует не только в одном прецеденте. Один и тот же экземпляр может принимать участие во многих событиях в бизнесе. В отличие от интерфейсных и управляющих объектов, объекты-сущности представляют сущности, не являющиеся человеческими или техническими ресурсами. Типичные примеры объектов-сущностей: ***Продукция***, ***Извещение*** (***Invoice***) и ***Заказ***.

О-модель с помощью устанавливаемых между объектами отношений показывает, каким образом реализуются прецеденты, описанные в П-модели. Данные отношение связывают два объекта (экземпляра или класса) и изображаются стрелками (дугами). Изучая

прецеденты, в которых участвует объект, можно получить представление об обязательствах объекта по отношению к его окружению. Поведение, описываемое в классе объекта, должно подтверждать эти обязательства. Если необходимо более подробно показать, как объекты взаимодействуют при выполнении прецедентов, то для этого привлекают диаграммы взаимодействий.

П-модель и О-модель – это разные типы описания бизнеса. О-модель, состоящая из сотен объектов и более, неудобна для работы и малопонятна, так как слишком сложна. Она содержит детали того, как объекты связаны друг с другом, как они общаются, какой интерфейс используют, что они знают друг о друге и т.д. Информация такого рода делает любое описание слишком детальным, и есть риск, что больше энергии будет тратиться на объяснение внутреннего взаимодействия, чем на понимание бизнес-процессов. Конечно, информация такого рода очень важна при создании подробных моделей бизнес-систем, однако она не обязательна для понимания прецедентов.

Как указывают многие специалисты, выявлять объекты для модели не сложно, сложно находить правильные объекты. Рекомендуется для каждого прецедента определить объекты, необходимые для его выполнения. Если в объектную модель включать объекты, принимающие участие в различных прецедентах, то будет больше шансов определить действительно нужные объекты. Рассмотрев все прецеденты, в которых некоторый объект играет какие-либо роли, можно понять назначение этого объекта в системе.

2.2.3. Пример UML-модели бизнес-системы

В качестве примера анализируемой и моделируемой бизнес-системы рассмотрим систему «***Ресторан***», аналогично примеру, использованному в работе [Ойхман, 1997]. В целях конкретизации задачи предполагается, что анализ и моделирование осуществляются с целью проектирования информационной системы поддержки реинжиниринга подобного вида организаций, которая должна использовать модель существующего бизнеса. Диаграммы выполнены с помощью инструментального пакета Rational Rose.

На рисунке 3.32 представлена внешняя модель (П-модель) бизнес-системы «Ресторан», описывающая функции ресторана по отношению к его клиентам и поставщикам. Данная модель является UML-диаграммой прецедентов или вариантов использования, выполненной с учетом отношений расширения и использования между прецедентами.

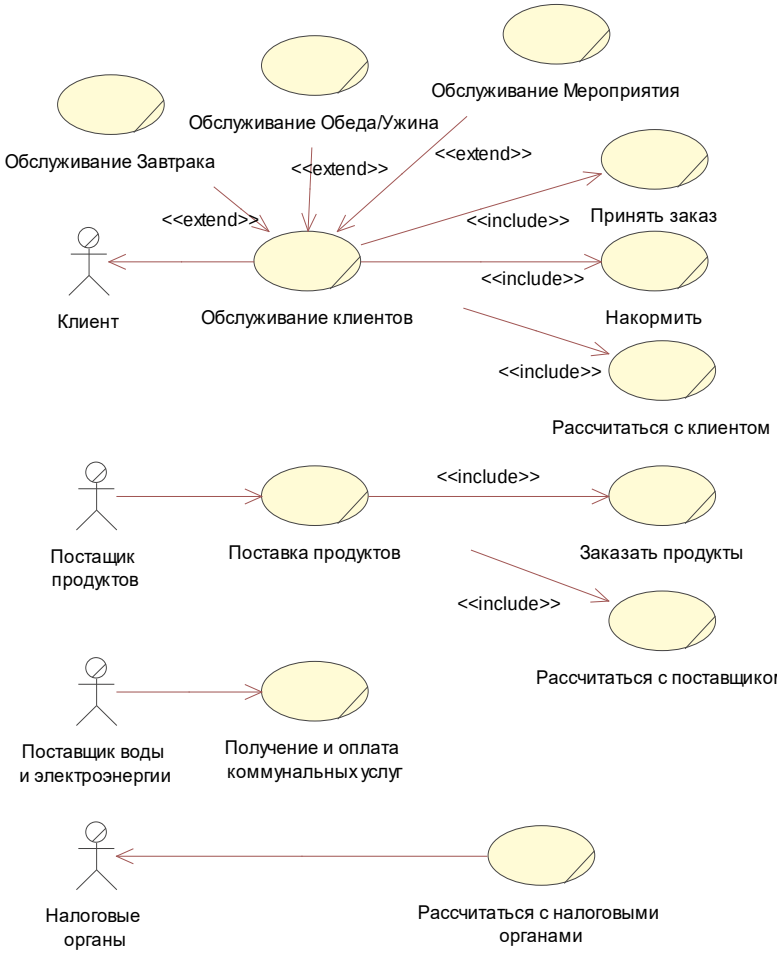


Рис. 3.32. П-модель (внешняя модель) бизнес-системы «Ресторан»

На рисунках 3.33 и 3.34 представлена внутренняя модель (О-модель) бизнес-системы «Ресторан», описывающая типы объектов (классы) ресторана и взаимодействия между ними. Модель на рисунке 3.33 является UML-диаграммой классов для одного из вариантов использования бизнес-системы «Ресторан» (для прецедента «Обслуживание обеда/ужина»). Модель на рисунке 3.34 является UML-диаграммой кооперации между экземплярами (объектами) классов, представленных на диаграмме 3.33, т.е. для того же прецедента «Обслуживание обеда/ужина».

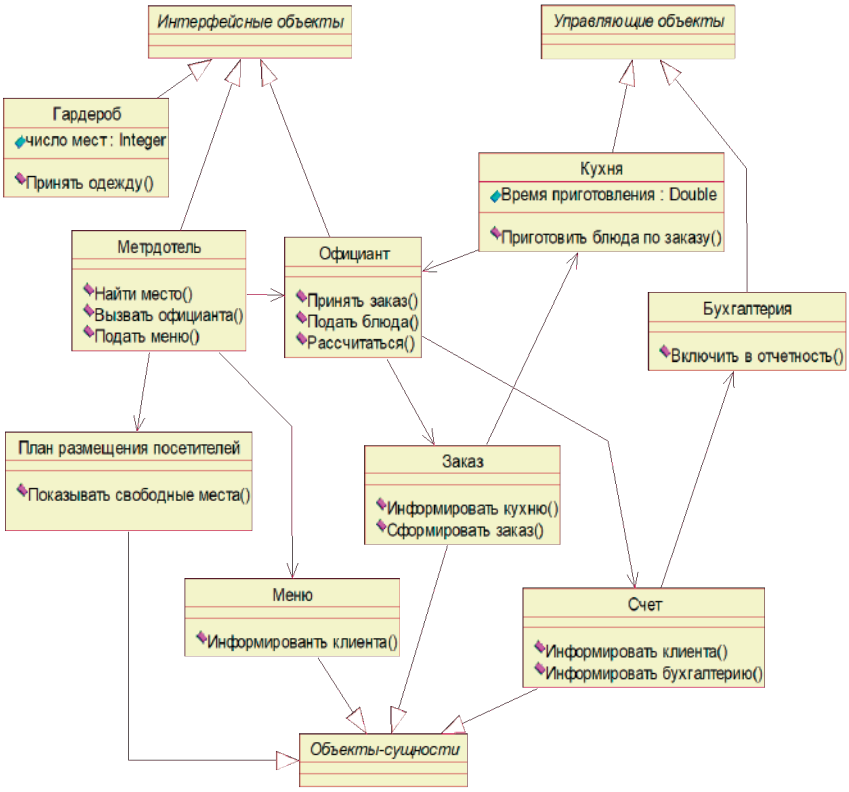


Рис. 3.33. О-модель (внутренняя модель) бизнес-системы «Ресторан». Диаграмма классов для прецедента «Обслуживание обеда/ужина»

3. Нотация BPMN

«Нотация моделирования бизнес-процессов (Business Process Modeling Notation (BPMN))» разработана инициативной группой по управлению бизнес-процессами (Business Process Management Initiative (BMPI)), которая проанализировала опыт многих существующих нотаций и попробовала объединить лучшие концепции разных нотаций в одну стандартную. Основное назначение получившегося стандарта – обеспечение пользователей нотацией, понятной всем участникам бизнес-сферы, от бизнес-аналитиков, создающих перво-

начальные эскизы процессов, технических разработчиков, ответственных за внедрение технологии, в которой будут представлены данные процессы, и, наконец, до бизнесменов, которые будут управлять этими процессами. Таким образом, BMPN является стандартизованным связующим звеном между разработкой бизнес-процессов и их реализацией.

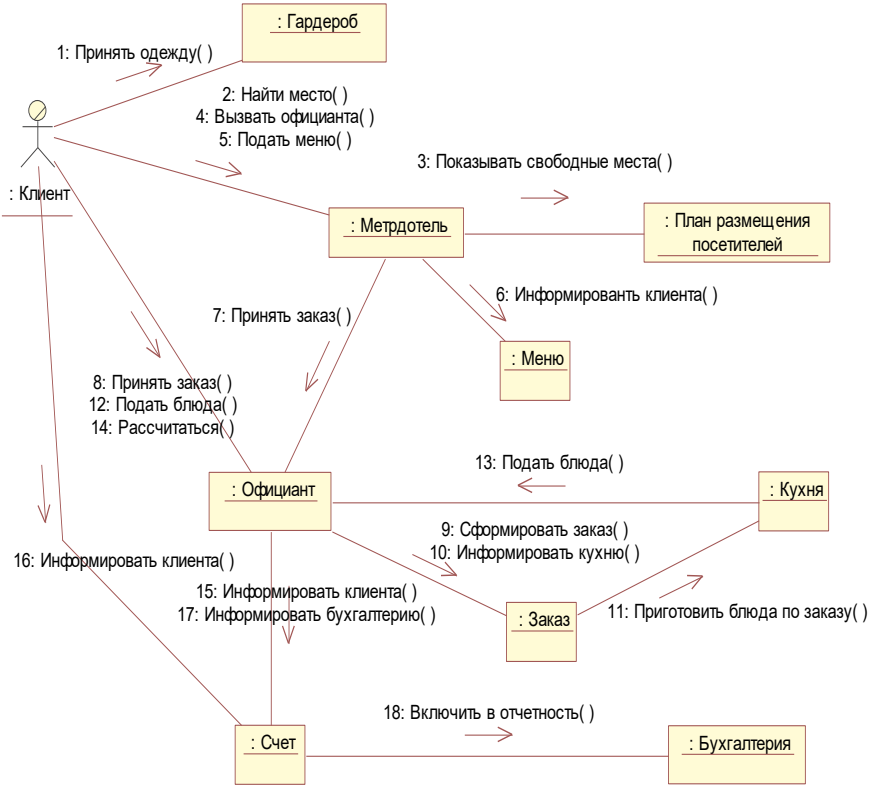


Рис. 3.34. О-модель (внутренняя модель) бизнес-системы «Ресторан». Диаграмма кооперации объектов для прецедента «Обслуживание обеда/ужина»

Другой не менее важной целью создания стандарта бизнес-моделирования является визуализация бизнес-процессов посредством бизнес-ориентированного подмножества языков XML (например, такого как BPEL (Business Process Execution Language – язык выполнения бизнес процессов)), разработанных для их исполнения.

Слияние концерна OMG с группой BMPI и его активная поддержка графической нотации BPMN свидетельствуют о признании объектно-ориентированном сообществом фактической непригодности использования языка UML для моделирования бизнес-процессов.

3.1. Диаграммы бизнес-процессов (BPD)

Одной из причин создания BPMN явилась необходимость использования единого простого механизма для проектирования как простых, так и сложных моделей бизнес-процессов. Для удовлетворения этих двух противоречащих друг другу требований была осуществлена систематизация графических элементов по категориям. Результатом явился небольшой перечень категорий элементов графической нотаций, позволивший людям, работающим с диаграммами BPMN, без труда распознавать основные типы элементов и осуществлять корректное чтение схем (BPD-диаграмм). Основные категории элементов допускают внутренние вариации, а также добавление информации для удовлетворения требований сложности без внесения значительных изменений в общую структуру диаграммы для легкости её понимания.

Возможности этой нотации учитывать общесистемные закономерности представлены в Приложении №1.

Существуют четыре основные категории элементов BPMN [Выдержки]:

- *Элементы потока* (Flow Objects);
- *Соединяющие элементы* (Connecting Objects);
- *Зоны ответственности* (Swimlanes);
- *Артефакты* (Artifacts).

Элементы потока являются важнейшими графическими элементами, определяющими ход бизнес-процесса. Они, в свою очередь, делятся на:

- *События* (Events);
- *Действия* (Activities);
- *Шлюзы* (Gateways).

Выделяют три вида Соединяющих элементов:

- *Поток операций* (Sequence Flow);
- *Поток сообщений* (Message Flow);
- *Ассоциация* (Association).

Используются два способа группировки основных элементов моделирования с помощью Зон ответственности:

- Группировка с помощью Пула (Pool);
- Группировка с помощью Дорожки (Lane).

Артефакты используются для добавления дополнительной информации о процессе. Выделяют три типовых Артефакта, что, однако, не запрещает разработчикам моделей бизнес-процессов либо программам моделирования добавлять необходимое количество Артефактов. Для широкого круга пользователей, а также для вертикальных рынков существует возможность стандартизации более полного перечня Артефактов. Таким образом, текущий перечень Артефактов включает в себя следующие элементы:

- *Объект данных* (Data object);
- *Группа* (Gruop);
- *Аннотация* (Annotation).

3.2. Элементы потока

В таблице 3.5 представлены графические элементы BPMN, относящиеся к категории «Элементы потока», которые обозначают различные события.

Таблица 3.5

Графические элементы BPMN «Событие».

Описание элемента	***Основные графические элементы***
Событие (Event) Событие – это то, что происходит в течение бизнес-процесса и оказывает влияние на его ход. Событие имеет причину (триггер) или воздействие (результат), маркеры которых представлены на рисунке 3.35 Согласно влиянию Событий на ход бизнес-процесса, выделяют три типа: Стартовое событие (Start), Промежуточное событие (Intermediate) и Конечное событие (End)	Start Intermediate End

Следующий рисунок поясняет смысл маркеров (триггеров) событий.

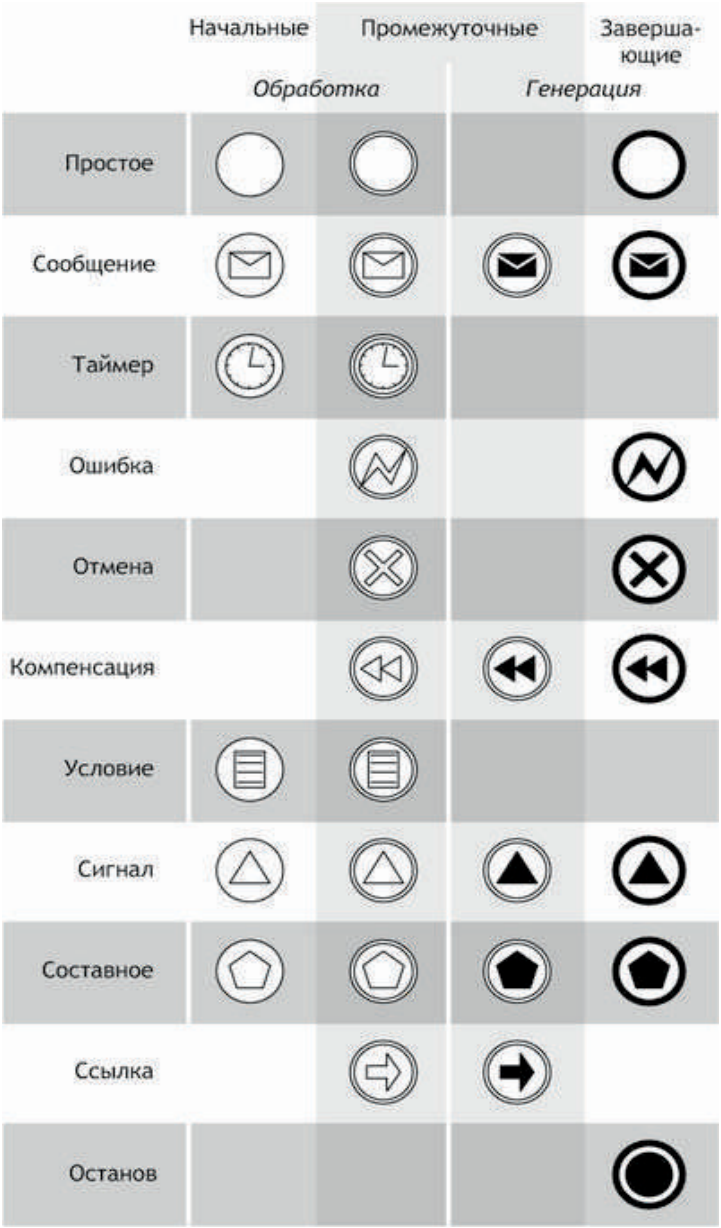
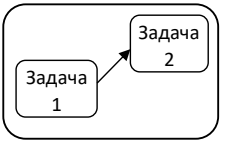
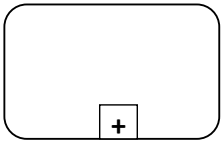


Рис. 3.35. Маркеры событий

В таблице 3.6 представлены графические элементы BPMN, относящиеся к категории «Элементы потока», которые обозначают различные действия и операции с потоками.

Таблица 3.6

Графические элементы BPMN «Действие» и «Шлюз».

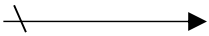
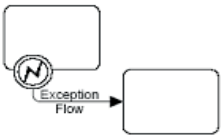
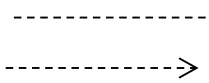
Описание элемента	*Основные графические элементы*
Действие (Activity) Действие – общий термин, обозначающий работу, выполняемую исполнителем. Действия могут быть либо элементарными, либо неэлементарными (составными). Выделяют следующие виды действий, являющихся частью модели Процесса: Процесс (Process), Подпроцесс (Sub-Process) и Задача (Task). Задача и Подпроцесс изображаются в виде прямоугольника с закругленными углами. Процесс либо не имеют границ, либо находятся внутри Пула. Для свернутых подпроцессов могут применяться следующие маркеры: ↻- циклический подпроцесс; || – многоэкземплярный подпроцесс; ~ – подпроцесс ad-hoc; ⏪ – компенсирующий подпроцесс	 Развернутый подпроцесс  Свернутый подпроцесс
Шлюз (Gateway) Шлюзы используются для контроля расхождений и схождений потока операций. Таким образом, данный термин подразумевает ветвление, раздвоение, слияние и соединение маршрутов. Внутренние маркеры указывают тип контроля развития бизнес-процесса: ✖ – эксклюзивное ИЛИ (XOR); ○ – ИЛИ (OR); ✚ – И (AND); ✱ – Комплексные/сложные (Complex); ✸ – основано на событиях (Event-based). Шлюзы каждого из типов оказывают влияние, как на входящие, так и на исходящие потоки.	

3.3. Соединяющие элементы

В таблице 3.7 представлены графические элементы BPMN, относящиеся к категории «Соединяющие элементы», которые обозначают различные потоки, связывающие между собой события, действия и шлюзы.

Таблица 3.7

Соединяющие графические элементы BPMN.

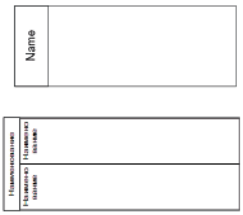
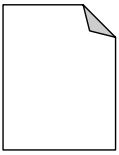
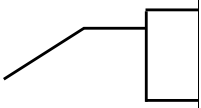
Описание элемента	*Основные графические элементы*
Поток операций (Sequence Flow) Поток операций служит для отображения того порядка, в котором организованы действия Процесса:	**См. ниже**
Стандартный поток (Normal Flow)	
Неконтролируемый поток (Uncontrolled Flow) Неконтролируемый поток операций относится либо к потокам, на которые не воздействую никакие условия, либо к потокам, не проходящим через Шлюзы	
Условный поток (Conditional Flow) Поток операций может зависеть от условных выражений, оценивающихся согласно времени выполнения для того, чтобы определить, будет ли использоваться поток или нет	
Поток по умолчанию (Default Flow) Поток операций данного типа используется в том случае, если все остальные исходящие Условные потоки операций не являются верными во время выполнения действия	
Исключающий поток (Exception Flow) Поток исключений встречается за пределами Стандартного потока операций. Основывается на Промежуточных событиях, возникающих в ходе Процесса	
Поток сообщений (Message Flow) Поток сообщений служит для отображения обмена сообщениями между двумя участниками, готовыми эти сообщения отсылать и принимать. На диаграмме BPMN два отдельно взятых Пула представляют собой двух участников процесса (бизнес-объекты или бизнес-роли)	
Ассоциация (Association) Ассоциация служит для установления связи между информацией и элементами потока. Текстовые объекты, а также графические объекты, не относящиеся к элементам потока, могут соотноситься с Элементами потока	

3.4. Зоны ответственности и артефакты

В таблице 3.8 представлены графические элементы BPMN, относящиеся к категориям «Зоны ответственности» и «Артефакты».

Таблица 3.8

Элементы BPMN «Зоны ответственности» и «Артефакты»

Описание элемента	*Основные графические элементы*
Пул (Pool) Пул представляет собой Участника Процесса. Он также может выступать в качестве Зоны ответственности или графического контейнера, отвечающего за разделение определенного набора действий. **Дорожка (Lane).** Дорожка – это подраздел в пределах пула, его протяженность равна длине пула, как по вертикали, так и по горизонтали. Дорожки организовывают и классифицируют действия	
Объект данных (Data object) Объект данных относится к Артефактам, т.к. не оказывает непосредственного влияния ни на Поток операций, ни на Поток сообщений. Однако Объект данных предоставляет информацию о том, какие действия необходимо выполнить и/или каков результат этих действий	
Группа (Gruop) Группировка действий, не оказывающая влияние на последовательный поток. Группировка может использоваться для документирования или анализа. Группы могут так же использоваться для обозначения действий распределенных групповых операций, расположенных по ширине областей	
Аннотация (Annotation) Текстовая аннотация – это возможность для разработчика привести дополнительную информацию для читателя схемы BPMN	

Текстовые аннотации элементов BPD-диаграм в BPMN-нотации могут отражать дополнительную информацию о процессе или атрибутах элементов в рамках процесса.

- У элементов и потоков МОГУТ быть признаки (например, название и/или другие атрибуты), находящиеся внутри формы, а также над, либо под формой, в любом направлении и месте, в зависимости от пожеланий разработчика или продавца инструмента моделирования.
- Заливка, предназначенная для графических объектов, МОЖЕТ быть белой или прозрачной.

- МОГУТ использоваться другие цвета заливки в соответствии с целями разработчика или продавца инструмента моделирования (например, в целях подчеркивания значимости атрибута объекта).
- Элементы схемы и маркеры МОГУТ быть любого размера, в соответствии с целями разработчика или инструмента моделирования.
- Линии, используемые для рисования графических объектов, МОГУТ быть черными.
- МОГУТ использоваться другие цвета линии в соответствии с целями разработчика или инструмента (например, в целях подчеркивания значимости атрибута объекта).
- МОГУТ использоваться другие стили линии в соответствии с целями разработчика или инструмента (например, в целях подчеркивания значимости атрибута объекта) при условии, что стиль линии НЕ ДОЛЖЕН вступать в конфликт ни с одним стилем линии, определенным BPMN. Таким образом, НЕ ДОЛЖНЫ изменяться стили линий последовательного потока, потока сообщений и ассоциаций.

3.5. Правила соединения элементов потока

Входящий Поток операций может быть присоединен к любой точкой Элемента потока (слева, справа, сверху, снизу). Подобно ему, Исходящий поток операций может брать начало из любой точки Элемента потока (слева, справа, сверху, снизу). Поток сообщений обладает теми же свойствами, что и Поток операций.

Для соединения Элементов потока разработчикам моделей настоятельно рекомендуется использовать имеющийся опыт, что облегчит понимание создаваемых диаграмм и сделает ход изображаемого бизнес-процесса прозрачным и доступным для понимания. Это особенно важно в том случае, если в диаграмме присутствуют такие графические элементы, как Поток операций или Поток сообщений. В данном случае оптимальным вариантом является выбор направления Потока сообщений, который необходимо расположить под углом в 90° по отношению к уже выбранному Потоку операций.

Таблица 3.9 содержит изображения Элементов потока, используемые языком BPMN, и указывает, каким образом данные графические элементы соединяются друг с другом посредством Потока операций [Выдержки].

Таблица. 3.9

Правила соединения с помощью Потока операций.

От/К						
		↗	↗	↗	↗	↗
		↗	↗	↗	↗	↗
		↗	↗	↗	↗	↗
		↗	↗	↗	↗	↗
		↗	↗	↗	↗	↗

Символ ↗ обозначает, что графический элемент, изображенный в одной из строк таблицы, может соединяться с графическим элементом, изображенным в соответствующей колонке. В таблице не указывается количество входящих и исходящих соединений графического элемента, зависящее от различных конфигураций.

Таблица 3.10 содержит изображения объектов моделирования BPMN, а также указывает, каким образом данные объекты соединяются друг с другом посредством Потока сообщений. Символ ⇗ обозначает, что графический элемент, изображенный в одной из строк таблицы, может соединяться с графическим элементом, изображенным в соответствующей колонке. В таблице не указывается количество входящих и исходящих соединений графического элемента, зависящее от различных конфигураций [Выдержки].

3.6. Примеры моделей в нотации BPMN

Моделирование бизнес-процессов в нотации BPMN используется для донесения широкого спектра информации до различных категорий пользователей. BPD-диаграммы бизнес-процессов позволяют создавать три типа моделей для описания бизнес-процессов:

- Частные (внутренние) бизнес-процессы.

- Абстрактные (открытые) бизнес-процессы.
- Процессы взаимодействия (глобальные).

Частные бизнес-процессы описывают внутреннюю деятельность организации. Они представляют бизнес-процессы в общепринятом понимании (business processes или workflows). При использовании ролей частный бизнес-процесс помещается в отдельный пул. Поэтому поток управления находится внутри одного пула и не может пересекать его границ. Поток сообщений, напротив, пересекает границы пулов для отображения взаимодействия между разными частными бизнес-процессами.

Таблица 3.10

Правила соединения с помощью Потока сообщений.

От/К		(Pool)				
(Pool)	⇗	⇗	⇗	⇗	⇗	
	⇗	⇗	⇗	⇗	⇗	
	⇗	⇗	⇗	⇗	⇗	
	⇗	⇗	⇗	⇗	⇗	

Абстрактные (открытые) бизнес-процессы служат для отображения взаимодействия между двумя частным бизнес-процессами (то есть между двумя участниками взаимодействия) В открытом бизнес-процессе показываются только те действия, которые участвуют в коммуникации с другими процессами. Все другие, «внутренние», действия частного бизнес-процесса не показываются в абстрактном процессе. Таким образом, абстрактный процесс показывает окружающим последовательность событий, с помощью которой можно взаимодействовать с данным бизнес-процессом. Абстрактные процессы помещаются в пулы и могут моделироваться как отдельно, так и внутри большей

диаграммы для отображения потока сообщений между действиями абстрактного процесса с другими элементами. Если абстрактный процесс и соответствующий частный процесс находятся в одной диаграмме, то действия, отображённые в обоих процессах, могут быть связаны ассоциациями.

Процессы взаимодействия (глобальные) отображает взаимодействия между двумя и более сущностями. Эти взаимодействия определяются последовательностью действий, обрабатывающих сообщения между участниками. Процессы взаимодействия могут помещаться в пул. Эти процессы могут моделироваться как отдельно, так и внутри большей диаграммы для отображения ассоциаций между действиями и другими сущностями. Если процесс взаимодействия и соответствующий частный процесс находятся в одной диаграмме, то действия, отображённые в обоих процессах, могут быть связаны ассоциациями.

Рассмотрим далее пример бизнес-процесса «Регистрация на рейс».

Сначала приводится словесное описание процесса, а потом один из вариантов его представления в BPMN. Данный пример не гарантирует максимального приближёния к реальному процессу, а ставит целью показать использование конструкций нотации BPMN.

Когда пассажир прибывает в аэропорт, его приоритетной задачей является регистрация на рейс. Сотрудник на стойке регистрации приветствует клиента и берёт у него документы: билет на рейс и паспорт. Если документы клиента не в порядке (например, истёк срок действия паспорта), он не может быть зарегистрирован на рейс и процесс завершается. При этом клиент получает документы обратно.

Если паспорт и билет в порядке, то сотрудник авиакомпании регистрирует клиента на рейс и распечатывает посадочный талон. При этом он взаимодействует с информационной системой авиакомпании. Сотрудник отдаёт пассажиру посадочный талон и паспорт, после чего уточняет, нет ли в багаже пассажира запрещённых грузов (например, воспламеняющихся веществ). Если таковые есть, то они изымаются из багажа. Сотрудник авиакомпании забирает багаж и ручную кладь пассажира и регистрирует её. При этом сотрудник снова взаимодействует с информационной системой авиакомпании. Если выясняется, что есть перевес, то сотрудник уведомляет об этом пассажира и сообщает сколько необходимо заплатить. После получения денег от пассажира, сотрудник регистрирует оплату в системе.

В итоге, пассажир получает багажную квитанцию. Сотрудник желает пассажиру приятного полёта, и процесс завершается.

На рисунке 3.36 приведен пример представления упомянутого выше словесного описания бизнес-процесса в виде BPD-диаграммы.

Далее на рисунках 3.37–3.39 приведены примеры BPD-диаграмм других бизнес-процессов.

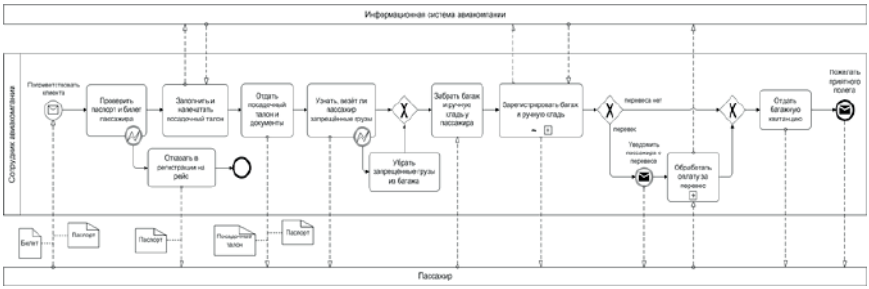


Рис. 3.36. Пример диаграммы в нотации BPMN. Регистрация на рейс

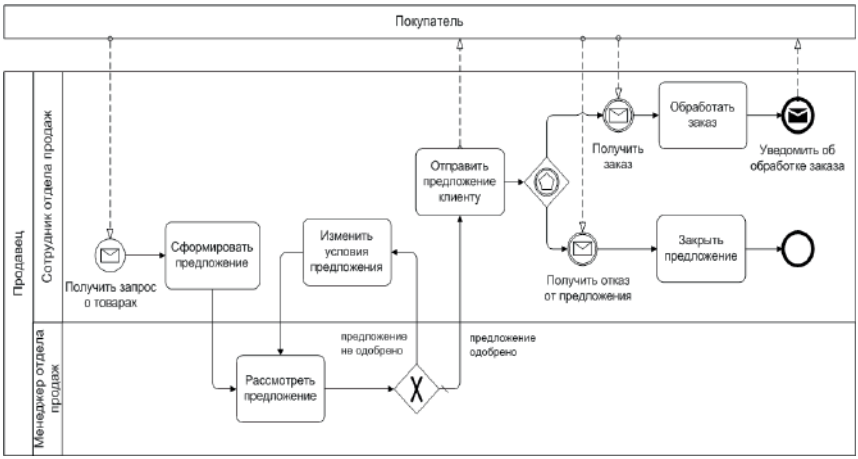


Рис. 3.37. Пример диаграммы в нотации BPMN. Обработка запроса о товарах

3.7. Недостатки моделирования в нотации BPMN

Анализируя графический язык моделирования бизнес-процессов в BPMN-нотации, можно обратить внимание на следующие недостатки:

1. Для освоения данной нотации требуются курсы, консультации и время для ее изучения на уровне достаточном для практического

использования. Маловероятно, что эта нотация будет хорошо понимаема управленцами.

2. Очень сложно моделировать большие иерархические системы, к которым относятся серьезные организации. А ведь именно для их моделирования такие средства и нужны в первую очередь.

3. Соединяющие элементы предназначены только для отображения порядка выполнения действий или появления событий и не отображают материальные и информационные потоки. Бизнес-процессы же, по сути своей, всегда являются проточными элементами, пропускающими через себя потоки материи и информации.

4. Введение элементов «Событие» и «Объект данных», представляющих, по сути дела, некоторые специфические виды связей. Использование, таким образом, избыточных сущностей, затрудняющих понимание диаграмм.

5. Авторы нотации BPMN утверждают, что данная нотация не предназначена для построения функциональных диаграмм и представления бизнес-правил. Но это может означать только то, что данная нотация вообще не предназначена для моделирования бизнес-процессов (хотя она, вроде бы, для этого и сделана), так как бизнес-процессы, по сути своей, всегда функциональны, а удачные модели бизнес-процессов представляют собой бизнес-правила.

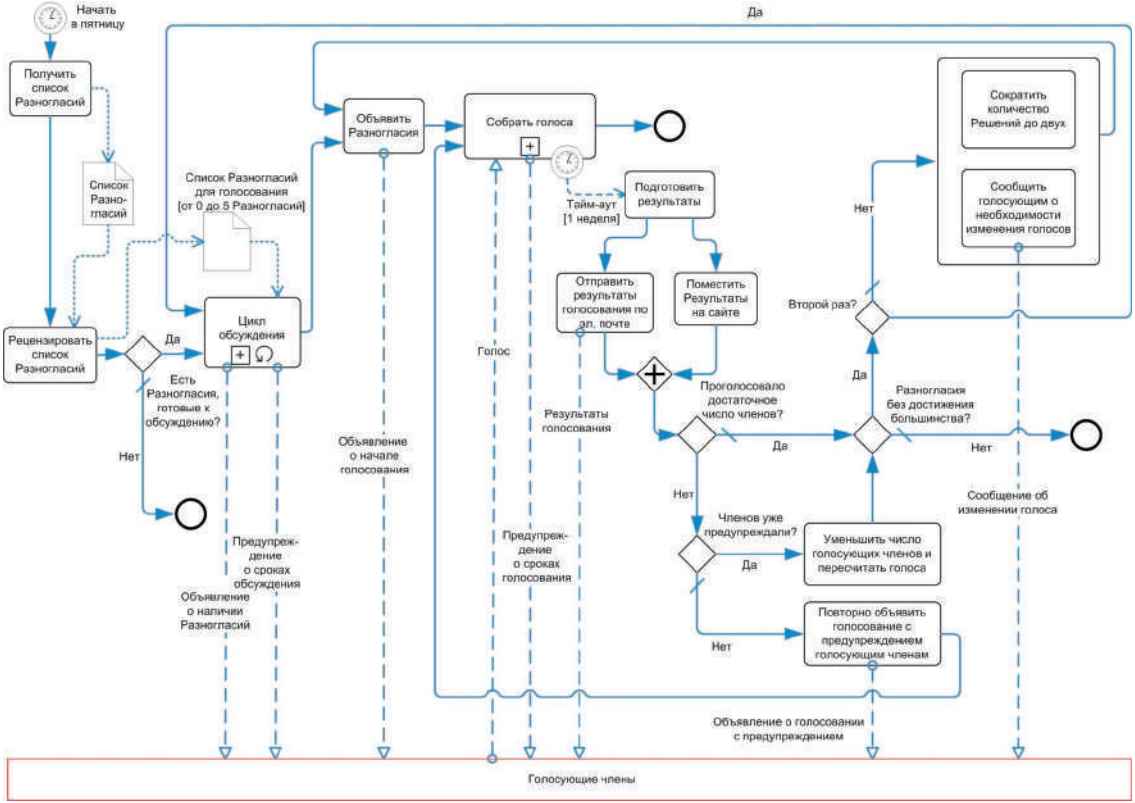


Рис. 3.38. Пример диаграммы в нотации BPMN. Голосование по электронной почте

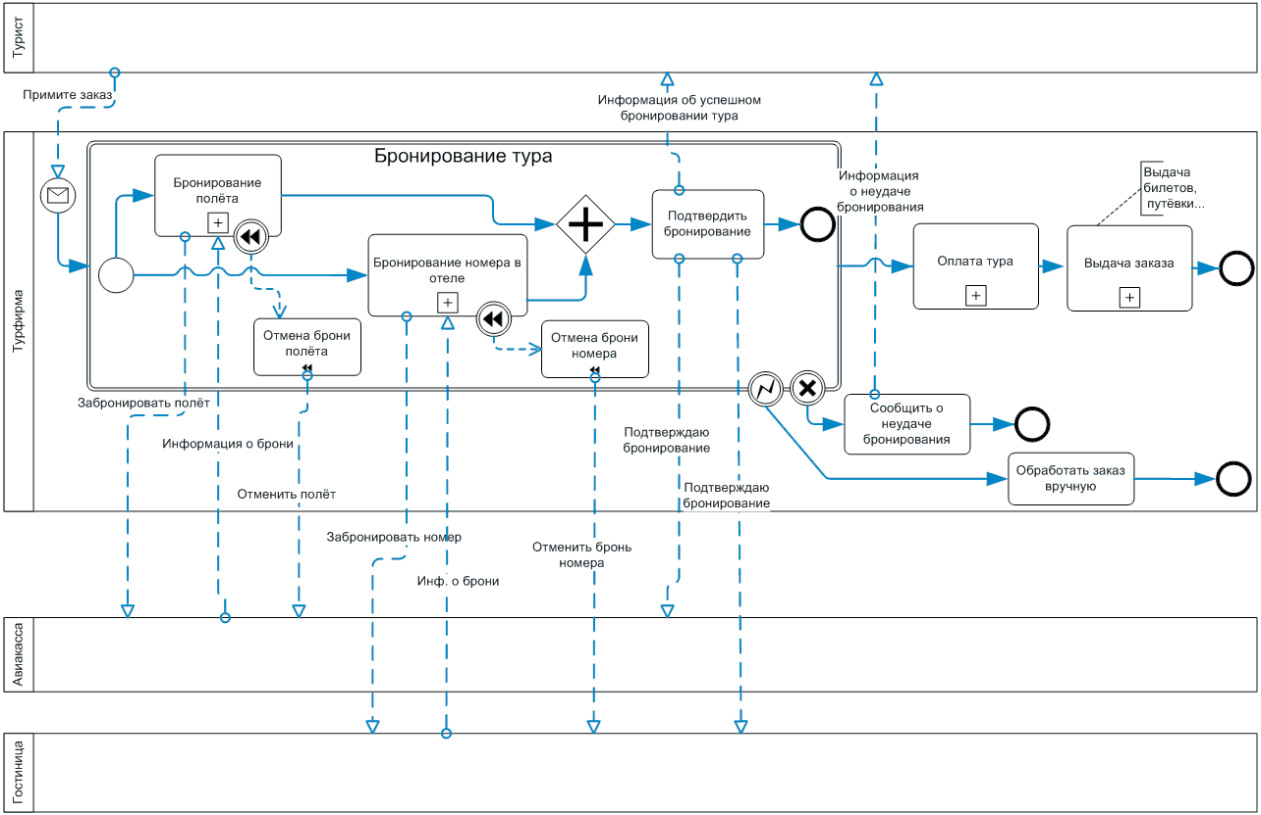


Рис. 3.39. Пример диаграммы в нотации BPMN. Деятельность турфирмы

4. Системно-объектное моделирование

4.1. Структурное системно-объектное моделирование

4.1.1. Нотация и методика структурного системно-объектного моделирования.

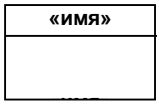
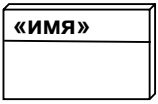
Диаграммы Узел-Функция-Объект (***УФО-диаграммы***) являются средством моделирования структурных, функциональных и субстанциальных характеристик любых систем. Возможности этих диаграмм учитывать общесистемные закономерности представлены в Приложении №1.

Нормативная система.

Для изображения УФО-диаграмм используются символы, представленные в таблице 3.11.

Таблица 3.11

Нормативная система (нотация) Узел-Функция-Объект.

Элемент алфавита	***Нотация УФО***
Связь/Поток Используется для моделирования передачи материальных или информационных потоков, соответствующих определенной классификации (см. далее) из одной части системы в другую. На диаграммах изображаются именованными стрелками, ориентация которых указывает направление потока	
Узел Используется для моделирования системы как структурного элемента надсистемы в виде перекрестка входящих и выходящих потоков (связей) материальных или информационных, выбранных из классификации (см. далее).	
Функция/Процесс Используется для моделирования системы как процесса преобразования системой в заданном узле входного потока в выходной, т.е. для представления системы, у которой кроме структурной определена и функциональная ее характеристика	
Объект Используется для моделирования системы как объекта, выполняющего заданную функцию в заданном узле, т.е. для представления системы, у которой кроме структурной и функциональной характеристик определена ее субстанциальная характеристика	

Построение модели.

При построении модели в виде иерархического множества УФО-диаграмм можно использовать упомянутые выше при описании DFD-диаграмм рекомендации [Калянов, 1997]. Кроме указанных рекомендаций необходимо учитывать следующие моменты, являющиеся следствием системно-объектного подхода и приведенных выше теоретических построений.

1. Перед построением контекстной УФО-диаграммы должны быть определены функциональные связи моделируемой системы с другими системами. Данные связи должны относится к четырем классам: «Вещественная связь (**V**)», «Энергетическая связь (**E**)»; «Связь по данным (**D**)» и «Управляющая связь (**C**)» в соответствии с базовой классификацией связей.

2. Построение диаграммы декомпозиции любого уровня должно предваряться расширением классификации связей, действующей на предыдущем уровне, и начинаться с выявления интерфейсных узлов, т.е. узлов, представляющих собой перекрестки входных и выходных функциональных связей данного уровня иерархии.

3. Процесс моделирования структуры анализируемой системы при необходимости может сопровождаться определением функциональности всех (или некоторых) структурных узлов и их субстанциальных (объектных) характеристик.

Суть процесса системно-объектного моделирования можно проследить на простом примере. Рассмотрим модель некоторой мифической организации, занимающейся торгово-закупочной деятельностью. Пусть это будет АООЗТ (акционерное общество очень закрытого типа) «*Рога и копыта*», классификация связей которой представлена на рисунках 3.40 и 3.41.

В целом (на контекстном уровне), как **узел** (с определенной функциональностью и объектными характеристиками)**,** АООЗТ является перекрестком связей/потоков представленных на рисунке 3.42 (в наименованиях связей через буквенно-цифровые обозначения показано, как эти связи вписываются в базовую классификацию связей).

Рис. 3.40. Материальные связи АООЗТ «Рога и копыта»

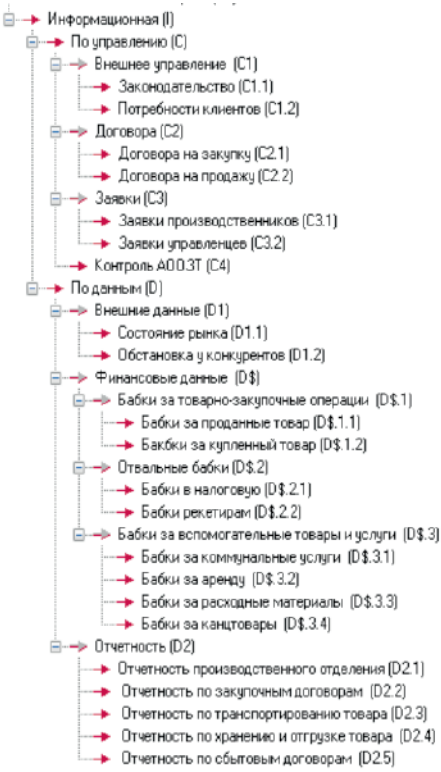


Рис. 3.41. Информационные связи АООЗТ «Рога и копыта»

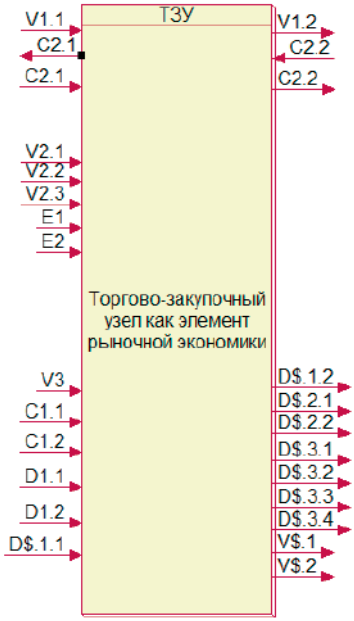


Рис. 3.42. АООЗТ «Рога и копыта» как перекресток входных и выходных связей, т.е. Узел (с определенной функциональностью и объектными характеристиками)

АООЗТ «*Рога и копыта*» в целом, как **функция**, в самом общем виде представляет собой торгово-закупочную деятельность, преобразующую представленные выше входя в выходы. Это представление может и должно быть дополнено описанием этой деятельности как процесса такой степени подробности и формальности, которые соответствуют имеющейся информации и целям анализа.

Как **объект** АООЗТ «*Рога и копыта*» в целом может иметь, например, следующие общие характеристики, которые также могут изменяться и дополняться: адрес, телефон, e-mail, web-страница, Ф.И.О. руководства, контактная информация, объем товарооборота (закупок и сбыта), длительность и стоимость производственного цикла, цена товара, рыночная цена фирмы, рейтинг, численность сотрудников, и т.д. и т.п.

Декомпозиция АООЗТ «*Рога и копыта*» на узлы с определенной функциональностью нижнего уровня может быть осуществлена следующим образом (см. рис. 3.43).

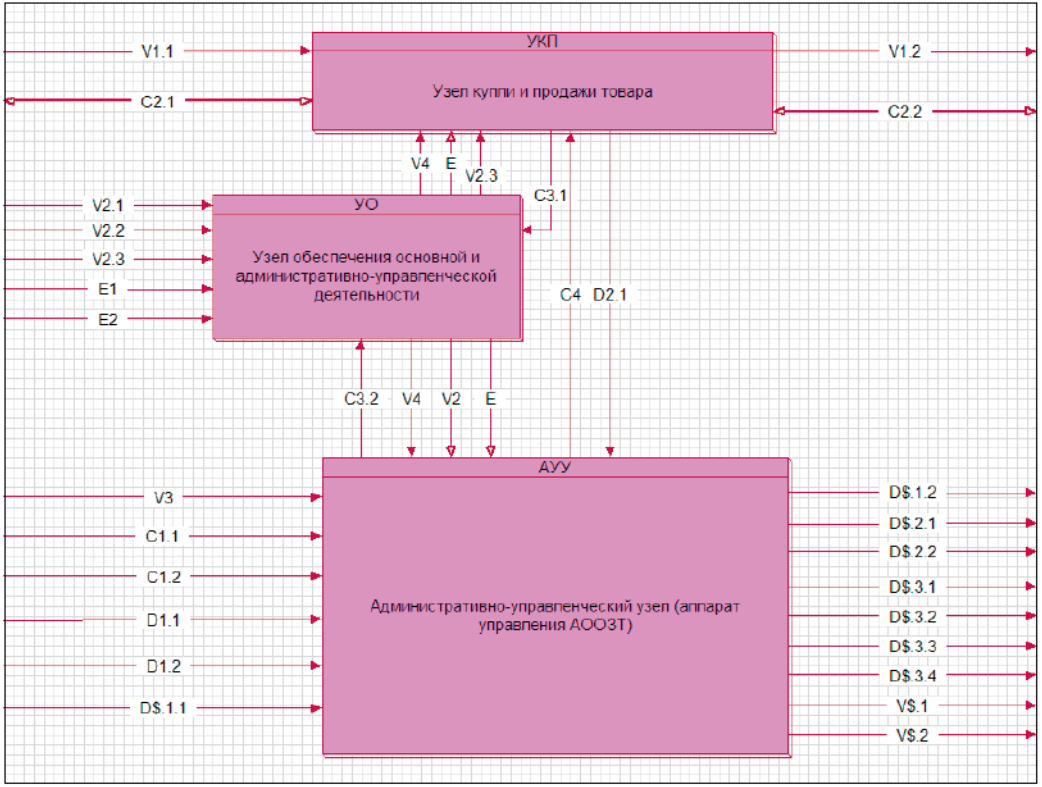


Рис. 3.43. Декомпозиция узла, занимаемого АООЗТ «Рога и копыта»

Естественно, на данном уровне декомпозиции появляются новые связи, не использовавшиеся на уровне общего представления АООЗТ, но также включающиеся теперь в классификацию. В данном случае это: C3.1 – Заявки производственников на расходные материалы и канцтовары; C3.2 – Заявки управленцев на расходные материалы, канцтовары и коммунальные услуги; C4 – Контроль со стороны управления за основной и вспомогательной деятельностью; D2.1 – Отчетность производственного отделения; V4 – Ремонтно-технический персонал для обслуживания. Кроме того, поток E представляет собой объединение потоков E1 и E2 (т.е. родовой по отношению к ним поток), а поток V2 – объединение потоков V2.1 – V2.3 (так же родовой поток).

При этом функция в узле **УКП**, соответствующая **основной деятельности** АООЗТ, может быть описана, например, как «деятельность по приобретению, транспортировке, хранению и сбыту товара». Объектом, который осуществляет эту деятельность фактически, может быть, например, **производственное отделение** (подразделения и должностные лица, занятые куплей, транспортировкой, хранением и сбытом товара).

Функция в узле **УО**, соответствующая **вспомогательной деятельности** АООЗТ, может быть описана, например, как «работа по материально-техническому обеспечению основной и административно-управленческой деятельности АООЗТ». Объектом, который осуществляет эту деятельность фактически, может быть, например, **вспомогательное отделение** (подразделения и должностные лица, занятые вспомогательной, обеспечивающей деятельностью).

Функция в узле **АУУ**, соответствующая **управленческой деятельности** АОЗТ, может быть описана, например, как «работа по координации и организации деятельности торгово-закупочного предприятия». Объектом, который осуществляет эту деятельность фактически, может быть, например, **управление** (подразделения и должностные лица, занимающиеся организацией, планированием, контролем, учетом, отчетностью, а также кадровыми вопросами).

Дальнейшая декомпозиция на УФО-элементы более глубоких ярусов (на узлы функциональные и функциональные объекты) осуществляется до тех пор, пока не будут выявлены узлы, функции в которых имеют четкое (по возможности формализованное) описание, позволяющее сформировать либо положения о конкретных подразделениях, либо инструкции конкретным должностным лицам. При этом должна быть гарантирована возможность существования объектов (людей, подразделений или технических средств), способных выполнить упомянутые функции.

Далее на рисунке 3.44 показан вариант декомпозиции узла купли и продажи товара (УКП) АООЗТ «*Рога и копыта*» на узлы с определенной функциональностью еще более нижнего уровня, выполненный средствами специального программного инструментария UFO-toolkit, поддерживающего структурное системно-объектное моделирование и который будет описан ниже.

Приведенный выше подход к построению моделей может быть усовершенствован в случае необходимости.

С практической точки зрения, может оказаться некорректно и неточно использование «информационных» связей без указания конкретного вида материального носителя данного вида информации, «энергетических» связей без указания конкретного вида вещественного носителя данного вида энергии, связей «по управлению» без указания конкретного вида данных, являющихся носителями этого вида управления. При этом, для построения моделей целесообразно использовать связи, являющиеся комбинациями следующих потоков: **V** – поток вещества, **VE** – поток энергии (с учетом вещественного носителя), **VD** – поток данных (с учетом вещественного носителя, например, бумажный документ), **VED** – поток данных (с учетом энергетического носителя, например, электронный документ), **VDC** – поток управления (с учетом вещественного носителя потока данных), **VEDC** – поток управления (с учетом энергетического носителя потока данных).

Кроме того, эффективность анализа и моделирования любой системы, как известно, в значительной степени зависит от правильности формулирования требований к ней, что выражается в необходимости построения, в первую очередь, контекстной модели системы. Для управления процессом контекстного моделирования организационных или бизнес-систем в рамках системно-объектного структурного моделирования вводится в рассмотрение «образ» узла, занимаемого системой, изображенный на рисунке 3.45. Использование данного образа при контекстном моделировании обеспечивает учет большего числа видов реально существующих взаимодействий бизнес-систем.

На данном рисунке показано, что любая система для производства своей выходной продукции должна получать на вход «предметы труда» (то из чего будет делаться продукция):

- или в виде вещества $\mathbf{V_{in}}$;
- или в виде энергии $\mathbf{VE_{in}}$ на некотором носителе;
- или в виде данных $\mathbf{VD_{in}}$ или $\mathbf{VED_{in}}$ опять же на некотором (вещественном или энергетическом) носителе.

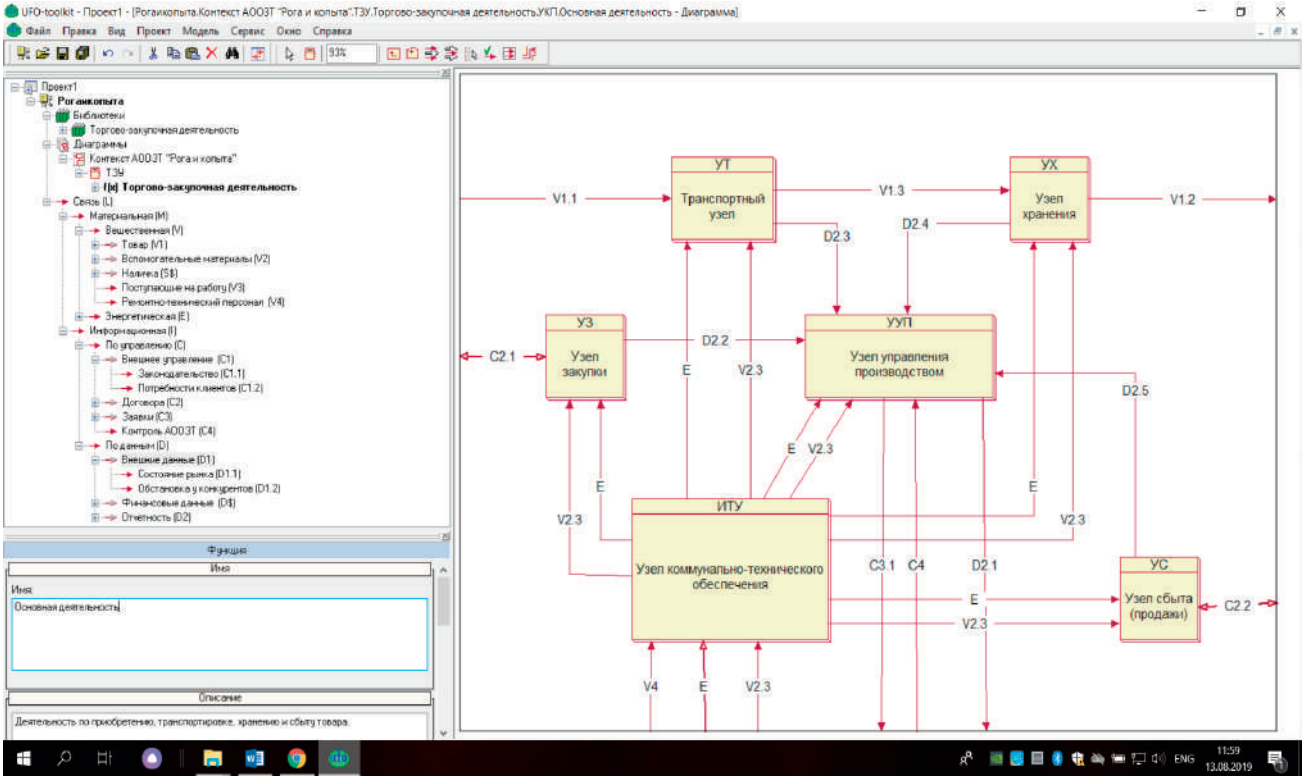


Рис. 3.44. Декомпозиция узла купли-продажи АООЗТ в среде UFO-toolkit

Эта связь соответствует связи «input» стандарта моделирования IDEF0.

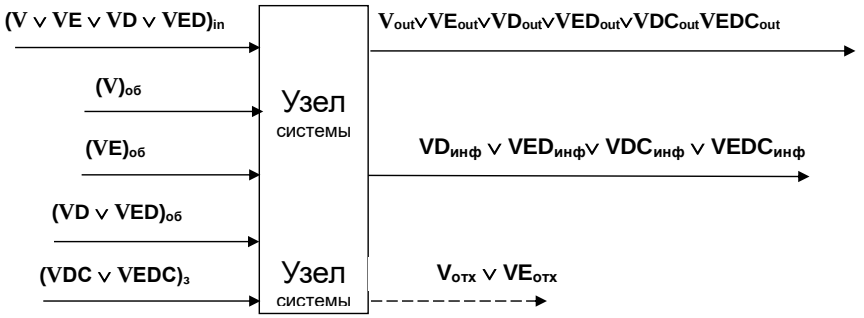


Рис. 3.45. «Образ» бизнес-системы

Для своего нормального функционирования система должна также получать:

- материально-техническое обеспечение **$S_{об}$**, например, оборудование;
- энергетическое обеспечение **$VE_{об}$**, например, электроэнергию;
- информационное обеспечение **$VD_{об}$** или **$VED_{об}$**, например, описания технологических процессов.

Эти связи соответствуют связи «mechanism» в стандарте IDEF0.

Кроме того, система должна получать управляющие воздействия **$VDC_з$** или **$VEDC_з$**, которые, в первую очередь, являются запросами (потребностями) тех систем, для которых данная система вырабатывает свои товары или услуги.

Эта связь соответствует связи «control» в стандарте IDEF0.

При этом допускается отсутствие у системы отдельных входов при наличии у нее достаточных собственных внутренних ресурсов данного вида.

На выходе системы в зависимости от отрасли деятельности могут быть:

- или **V_{out}** – вещество;
- или **V** – энергия на некотором носителе;
- или **VD_{out}** (**VED_{out}**) – данные на некотором носителе;
- или **VDC_{out}** (**$VEDC_{out}$**) – управляющая информация на носителе.

Эти связи соответствуют связи «output» в стандарте IDEF0.

Кроме того, на выходе системы может иметь место информация (данные – $VD_{инф}$ и/или $VED_{инф}$, либо управляющая информация – $VDC_{инф}$ и/или $VSEDC_{инф}$) о ее функционировании. Это могут быть, например, заявки другим системам на материалы и комплектующие или отчеты в налоговые органы, а также вещество или энергия, представляющие собой отходы производства ($V_{отх}$ и/или $VE_{отх}$), например, макулатура.

4.1.2. Программный инструментарий структурного системно-объектного моделирования

Рассмотрим инструментарий структурного системно-объектного моделирования «UFO-toolkit». Хотя название данного инструмента легко ассоциируется с аббревиатурой УФО (как и было задумано), в данном случае «UFO» – есть сокращение словосочетания «*User Familiar Object*» из английского компьютерного словаря, означающего «знакомый пользователю объект» (что также полностью соответствует сути дела).

В самых общих чертах инструмент обеспечивает представление любой системы (а также любой ее подсистемы и т.д.) в виде трехэлементной конструкции: «Узел – Функция – Объект» (УФО-элемента). Для выполнения названной функции UFO-toolkit поддерживает классификацию (библиотеку) УФО-элементов, основанную на классификации связей, пересечения которых и образуют «узлы».

Использование UFO-toolkit для моделирования предполагает предварительную специализацию классификации связей и УФО-библиотеки с учетом конкретной предметной области. Это позволяет создавать шаблоны классификаций и библиотеки для различных предметных областей, которые будут обеспечивать процесс моделирования множествами алфавитных УФО-элементов.

Использование инструмента непосредственно для системно-объектного моделирования организационных систем осуществляется следующим образом:

- Построение контекстной модели (самого верхнего уровня иерархии) анализируемой/проектируемой системы в виде «черного ящика» с указанием входных и выходных связей (функциональных), которые должны быть представлены в классификации связей. Процесс может начинаться либо со стороны классификации связей, либо со стороны контекстной модели.

- Выявление функциональных узлов в структуре моделируемой системы, т.е. узлов, функция которых либо уже известна, либо может быть сформулирована в результате проектирования. Для этого может использоваться таблица, в которой строки обозначаются входными связями, а столбцы – выходными. Эти связи также должны быть представлены в классификации связей. При этом процесс начинается с функциональных связей, показанных на контекстной модели. Если для моделирования используются шаблоны (алфавит) УФО-элементов, то обеспечивается возможность автоматической идентификации известных Инструменту узлов. Если шаблоны не используются или в них нет необходимых в данном случае узлов, то в таблицу добавляются внутренние связи, поддерживающие функциональные, показанные на контекстной модели. Этот процесс продолжается до тех пор, пока не будет обеспечен баланс «втекающих» и «вытекающих» потоков/связей контекстной модели. При этом ручная идентификация узлов должна сопровождаться модификацией классификации связей и УФО-библиотеки.
- Построение иерархической системно-объектной анализируемой или проектируемой системы. Данная модель в процессе анализа/проектирования представляет собой (на каждом уровне иерархии) совокупность взаимосвязанных функциональных узлов, идентифицированных с помощью таблицы. При этом каждому узлу должна назначаться функция (из всех известных и хранимых в УФО-библиотеке вариантов) в максимально возможной степени точно балансирующая данный узел. Для каждой же функции, в конце концов, должен быть указан объект (из всех известных и хранимых в УФО-библиотеке вариантов), реализующий ее оптимальным с точки зрения данного проекта образом. В результате, следовательно, системно-объектная модель представляет собой (на каждом уровне иерархии) совокупность взаимосвязанных функциональных объектов.

Рассматриваемое программное средство поддержки системно-объектного анализа и моделирования представляет собой CASE-инструмент категории toolkit, использующий базу знаний специальной конфигурации. Эта база знаний включает в себя библиотеку УФО-элементов и классификацию связей, имеет сетевую структуру и хранится в формате XML. Единицей хранения в этой базе является УФО-элемент, который, в свою очередь, является основой как для декомпозиции сложной системы на составные части, так и для процедуры синтеза сложной системы из более простых частей. При этом УФО-элементами

моделируются не только функциональные компоненты системы, но и компоненты, выполняющие роль связей. Формально УФО-элемент может быть представлен как класс языка объектного моделирования UML.

1. Назначение и условия выполнения программы

Программный пакет UFO-toolkit представляет собой современный CASE-инструментарий, основанный на знаниях. Программа предназначена для моделирования и проектирования сложных систем, в том числе организационных, информационных и технических. В основе UFO-toolkit лежит современный системно-объектный подход, который является первым и единственным вариантом системного подхода, согласованным с объектно-ориентированным подходом.

Программа предназначена для специалистов по консалтингу, информационным и CASE технологиям, реинжинирингу бизнеса, проектированию и моделированию, менеджеров, руководителей различными проектами.

Программа может выполняться на любом компьютере, на котором установлена операционная система Microsoft Windows 98 или более новая версия. Требования к аппаратной части компьютера соответствуют требованиям к операционной системе.

2. Описание интерфейса

2.1. Активация UFO-toolkit

Программа свободно скачивается с сайта www.ufo-toolkit.ru вместе с генератором ключа. При первом запуске после установки программы открывается мастер регистрации. Регистрация необходима для предотвращения несанкционированного использования программы и доступа пользователя к технической поддержке. Стартовое окно мастера регистрации показано на рисунке 3.46.

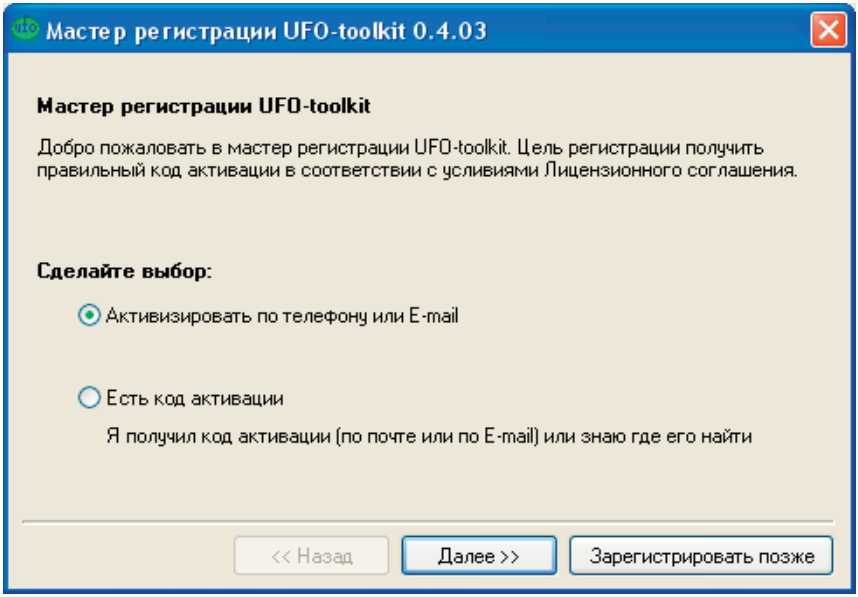


Рис. 3.46. Стартовое окно мастера регистрации UFO-toolkit

При нажатии на кнопку «Далее>>» появляется диалоговое окно, представленное на рисунке 3.47. Программа предлагает ввести активационный код. Для его получения необходимо ввести полученный при установке программы регистрационный номер в генератор ключа, скачиваемый с того же сайта. Работу можно начать только после того, как введен правильный активационный код. После активации программа привязывается к конфигурации компьютера, на котором установлена. На одном компьютере может быть одновременно установлено и активировано несколько версий UFO-toolkit.

В случае успешной регистрации и ввода корректного кода активации процедура регистрации завершается (рисунок 3.48), и программа начинает работу.

2.2. Окно приветствия

При запуске UFO-toolkit появляется диалог приветствия. С помощью него можно открыть одну из предыдущих моделей или проектов, или же создать новую модель. Окно приветствия представлено на рисунке 3.49. Окно появляется каждый раз, начиная с первого запуска. Если в появлении этого диалога нет необходимости, на нём можно установить опцию «Больше не показывать этот диалог». В дальнейшем появление окна может быть возобновлено при помощи опций.

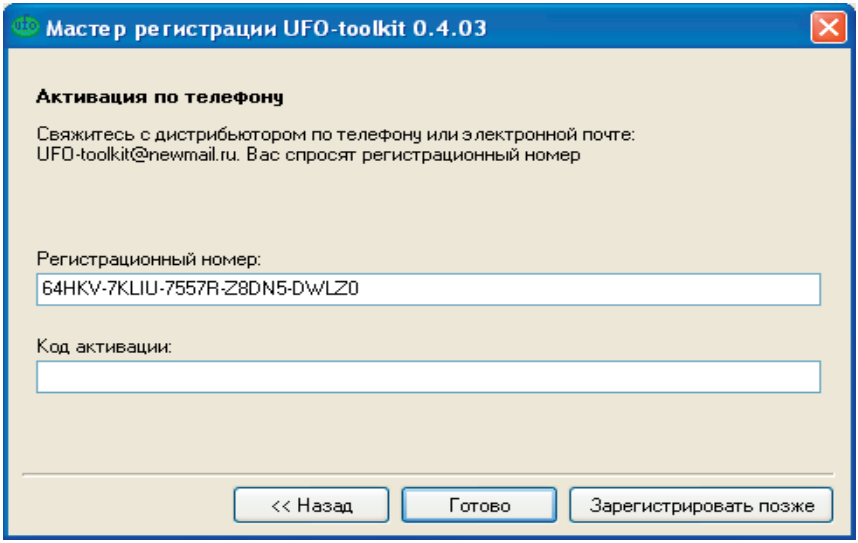


Рис. 3.47. Ввод кода активации UFO-toolkit

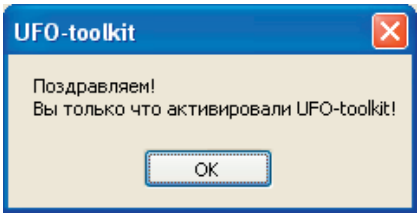


Рис. 3.48. Поздравления при успешной активации UFO-toolkit

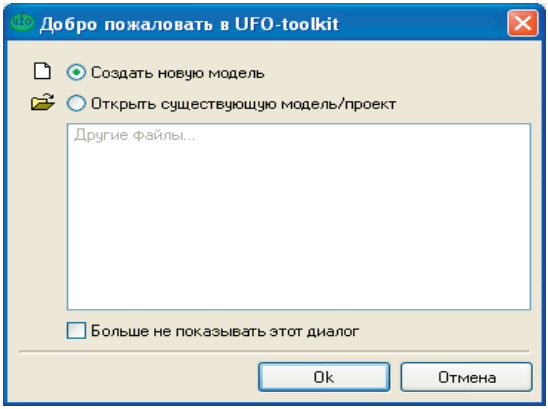


Рис. 3.49. Окно приветствия UFO-toolkit

2.3. Внешний вид программы

Основные элементы интерфейса программы – это браузер, панель свойств, панель инструментов и редактор узлов и библиотек. Кратко, назначение этих элементов заключается в следующем:

- браузер используется для быстрой навигации по моделям;
- панель свойств необходима для отображения параметров выделенных узлов и связей и для работы с ними;
- панель инструментов применяется для быстрого доступа к наиболее распространенным командам;
- редактор узлов и библиотек используется для редактирования одной или нескольких диаграмм или библиотек.

На рисунке 3.50 показан внешний вид программы.

2.4. Браузер

Браузер – это иерархическая структура, позволяющая легко осуществлять навигацию по модели. Всё; что существует в проекте, демонстрируется в окне браузера. С помощью браузера можно:

- добавлять к модели новые диаграммы и связи;
- просматривать существующие элементы модели;
- перемещать существующие в модели элементы;
- переименовывать эти элементы;
- добавлять элементы модели к диаграмме;
- вызывать просмотр и настройку свойств элементов;
- открывать диаграммы и библиотеки.

Браузер организован в древовидном стиле. Одни элементы модели (библиотеки, диаграммы, связи, узлы, функции, объекты) могут содержать другие элементы, находящиеся ниже их в иерархии. Знак «-» около элемента означает, что его ветка раскрыта. Знак «+» – что его ветка свернута. Браузер находиться в левой верхней части экрана.

2.5. Панель инструментов

Панель инструментов обеспечивает быстрый доступ к наиболее распространенным командам. На основной панели расположены общие команды. Динамическая панель инструментов автоматически выводит соответствующие кнопки, в зависимости от выполняемого в данный момент задания. Функции кнопок показаны в таблице 3.12.

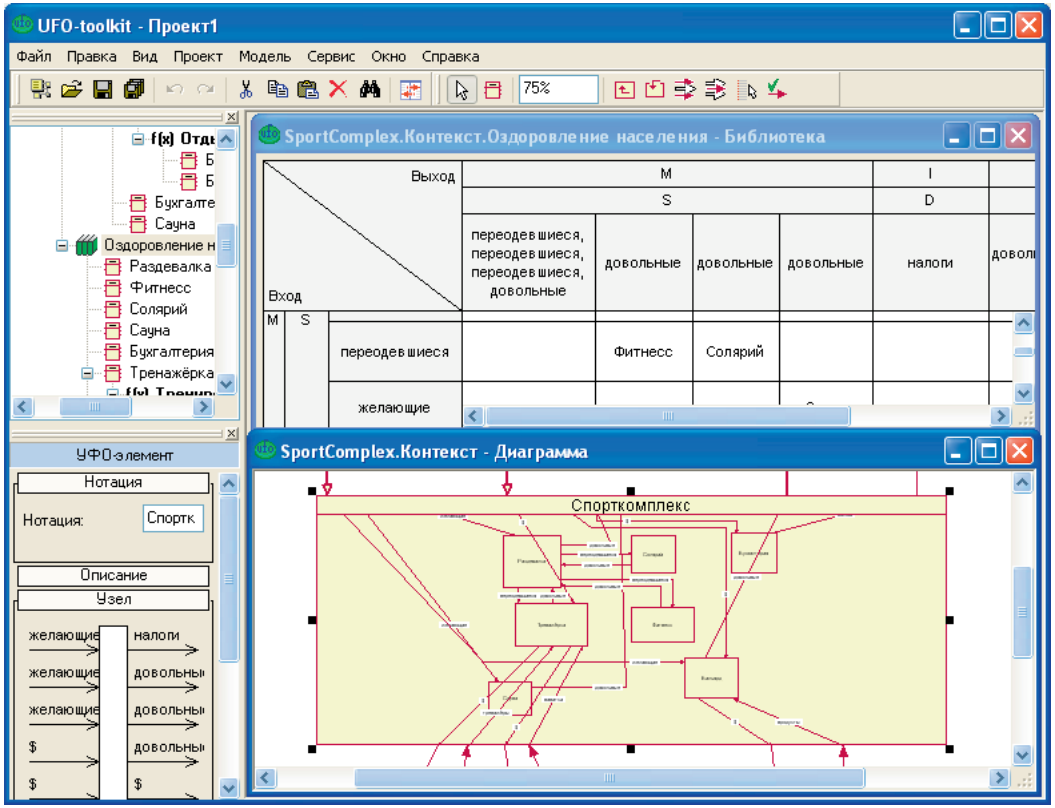


Рис. 3.50. Главное окно программы UFO-toolkit

Таблица 3.12

Кнопки на панели инструментов

Кнопка	***Назначение***
	Основная панель
Создать модель	Создает новую UFO модель
Открыть	Открывает существующую UFO модель
Сохранить	Сохраняет в файл текущую UFO модель
Сохранить всё	Сохраняет проект и все открытые модели
Отменить	Отменяет последнее выполненное действие (работает только для тех действий, отмена которых возможна)
Вернуть	Если какое-то действие было отменено, позволяет снова повторить его
Вырезать	Помещает выделенные объекты в буфер обмена и при этом удаляет их из текущей модели
Копировать	Помещает выделенные объекты в буфер обмена и при этом оставляет их без изменения в текущей модели (делает копию)
Вставить	Помещает объекты, находящиеся в буфере обмена, в активную область текущей модели (если такая вставка возможна)
Удалить	Удаляет выделенные объекты из текущей модели
Найти…	Производит поиск элементов модели
Определить местонахождение	Показывает выбранный элемент в браузере и на панели свойств
	Динамическая панель
Создать связь…	Позволяет создать в браузере новую связь
Удалить	Удаляет выделенные объекты из текущей модели
Создать библиотеку…	Создаёт новую библиотеку
Создать диаграмму…	Создаёт новую диаграмму
Построить библиотеку…	Построение новой библиотеки по выбранной диаграмме
Открыть	Показывает содержимое выделенного элемента
Создать функцию	Создаёт новую функцию для выбранного UFO-элемента
Сделать текущим	Делает выбранную функцию текущей
Создать объект…	Создаёт новый объект для выбранной функции

Кнопка	*Назначение*
Добавить UFO-элемент	Добавляет UFO элемент
Показывать пустые линии	Показывает или скрывает незаполненные строки и столбцы в библиотеке
Опции…	Вызывает диалог настройки опций
Выбор объектов	Позволяет выбирать объекты
Создание UFO-элементов	Переходит в режим создания UFO-элементов
Масштаб	Позволяет изменять масштаб диаграммы
На один уровень вверх	Показывает диаграмму верхнего уровня
Сгруппировать связи	Группирует выделенные связи в одну и даёт ей название ближайшего общего элемента в иерархии
Разгруппировать связи	Разгруппирует сгруппированные ранее связи
Выделить объекты…	Вызывает диалог выделения объектов
Выделять порты	Позволяет выделять порты связей при выделении объектов на диаграмме
Найти пути для связей	Для выделенных связей находит оптимальный путь между UFO-элементами
Выделить свободные порты	Выделяет все свободные порты на текущей диаграмме

2.6. Панель свойств

Панель свойств расположена в левой нижней части экрана (по умолчанию). На ней отображаются параметры выделенных элементов и связей для их просмотра и модификации. При выделении элемента могут отображаться вкладки «имя», «узел», «нотация», «описание», «стиль», «имя и нотация», «диаграмма», «группа связей», «UFO-сборщик». Отображённые вкладки могут быть свёрнуты или развёрнуты пользователем.

Вкладки «имя», «нотация» и «имя и нотация» позволяют просмотреть и изменить имя и нотацию выбранного элемента.

Во вкладке «узел» находиться редактор узлов. В нём к текущему элементу могут быть добавлены входные и выходные связи. Это делается путём переноса нужной связи из браузера на вкладку.

При помощи вкладки «описание» к любому элементу модели (имеющему данную возможность) может быть добавлено его описание.

Вкладку «стиль» имеют элементы, расположенные на диаграмме. Вкладка позволяет изменить стиль выбранного элемента (цветовое оформление), создать новый стиль и назначить стиль для будущих элементов.

Диаграмма имеет вкладку «диаграмма», на которой можно изменить ширину и высоту диаграммы, а также задать привязку к сетке и размер сетки, для элементов, расположенных на диаграмме.

При выделении группы связей отображается вкладка «группа связей», позволяющая просмотреть состав выделенной группы.

Вкладка «UFO-сборщик» отображается, когда выделены свободные порты. Она позволяет подбирать наиболее подходящие узлы для выделенных свободных портов и вставлять их на диаграмму.

2.7 Редактор узлов и библиотек

Используется для редактирования одной или нескольких диаграмм, или библиотек. Диаграммы и библиотеки имеют вид самостоятельных окон, которые могут быть свёрнуты или развёрнуты в правой части основного окна программы.

3 Работа в среде UFO-toolkit

3.1 Работа с меню

Здесь описываются все пункты меню UFO-toolkit. В конкретный момент времени в программе отображаются только те пункты меню, которые необходимы для текущей работы.

Таблица 3.13

Пункты меню UFO-toolkit

Пункт меню	***Описание***
Файл > Создать	Создает новую UFO модель или новое рабочее пространство
Файл > Открыть	Открывает существующую UFO модель или проект
Файл > Открыть проект...	Открывает существующее рабочее пространство
Файл > Открыть последний	Открывает последние из использовавшихся моделей и рабочих пространств
Файл > Закрыть проект	Закрывает рабочее пространство
Файл > Сохранить	Сохраняет текущую модель
Файл > Сохранить как…	Сохраняет текущую UFO модель под другим именем

Пункт меню	*Описание*
Файл > Сохранить проект как…	Сохраняет текущее рабочее пространство под другим именем
Файл > Сохранить всё	Сохраняет все открытые модели вместе с рабочим пространством
Файл > Выход	Завершает работу с UFO-toolkit
Правка > Отменить	Отменяет последнее выполненное действие (работает только для тех действий, отмена которых возможна)
Правка > Вернуть	Если какое-то действие было отменено, позволяет снова повторить его
Правка > Вырезать	Помещает выделенные объекты в буфер обмена и при этом удаляет их из текущей модели
Правка > Копировать	Помещает выделенные объекты в буфер обмена и при этом оставляет их без изменения в текущей модели (делает копию)
Правка > Вставить	Помещает объекты, находящиеся в буфере обмена, в активную область текущей модели (если такая вставка возможна)
Правка > Удалить	Удаляет выделенные объекты из текущей модели
Правка > Выделить всё	Выделяет всё…
Правка > Найти...	Находит необходимый элемент, связь или текст в текущем рабочем пространстве
Вид > Браузер	Позволяет показать или скрыть браузер
Вид > Редактор свойств	Позволяет показать или скрыть редактор (панель) свойств
Вид > Панели инструментов	Позволяет показать или скрыть панели инструментов
Проект > Добавить новую модель	Добавляет новую модель к рабочему пространству
Проект > Добавить существующую модель	Добавляет новую существующую модель к рабочему пространству
Модель > Удалить из проекта	Удаляет текущую модель из рабочего пространства

Пункт меню	*Описание*
Сервис > Опции...	Выводит на экран окно, позволяющее изменять параметры UFO-toolkit
Окно > Каскадом	Отображает открытые окна диаграмм и библиотек каскадом
Окно > Упорядочить	Поровну разделяет пространство между открытыми окнами
Окно > Минимизировать всё	Сворачивает все окна
Окно > Закрыть всё	Закрывает все окна
Окно > [Список текущих окон]	Пункт меню «Окна»содержит список открытых окон с помощью которого можно изменит активное окно
Окно > Окна…	Выводит список открытых окон для их просмотра и обработки
Справка > Документация…	Показывает документацию и справку по UFO-toolkit
Справка > О программе…	Выводит информацию об авторах и версии UFO-toolkit

С помощью некоторых команд меню в программе вызываются дополнительные диалоговые окна. Они позволяют пользователю уточнить различные системные настройки и параметры и предоставляют дополнительные функции.

3.2 Диалоговые окна

В UFO-toolkit существует ряд диалоговых окон. Это поиск, опции, выделение объектов на диаграмме и работа с открытыми окнами в редакторе библиотек и диаграмм. Они реализуют ряд функций программы. Ниже описаны их расположение и функции.

3.2.1 Поиск

С помощью пункта меню «Правка > Найти...» вызывается диалоговое окно, позволяющее произвести поиск необходимой информации в текущем рабочем пространстве. Внешний вид этого окна представлен на рисунке 3.51.

В поле вверху этого окна вводится искомое выражение. Если это выражение уже искалось раньше, то его можно найти во всплывающем списке. Можно также модифицировать опции поиска для определения места и типа поиска. В данном окне могут быть отображены не все

опции. Часть опций можно скрыть, а потом снова вывести нажатием клавиши «Опции<<«. Поиск осуществляется по нажатию на кнопку «Найти Далее».

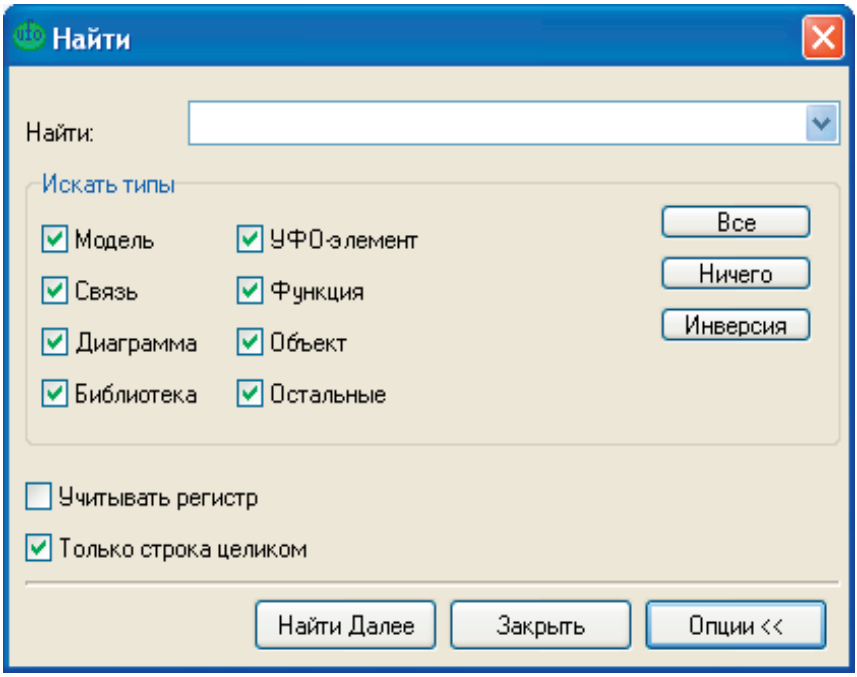


Рис. 3.51. Диалог поиска

3.2.2 Опции

С помощью пункта меню «Сервис > Опции...» вызывается диалоговое окно, позволяющее настроить опции UFO-toolkit. Внешний вид этого окна на рисунке 3.52.

В левой части окна расположен браузер, позволяющий выбирать вид настраиваемых опций. В правой части отображаются настраиваемые опции.

Заголовок «Среда» содержит несколько общих опций: установка подтверждения удаления, включение и отключение окна приветствия при старте, возможность открывать один объект в нескольких окнах.

Пункт «Интернациональные настройки» позволяет выбрать язык, на котором будут отображаться все надписи в программе.

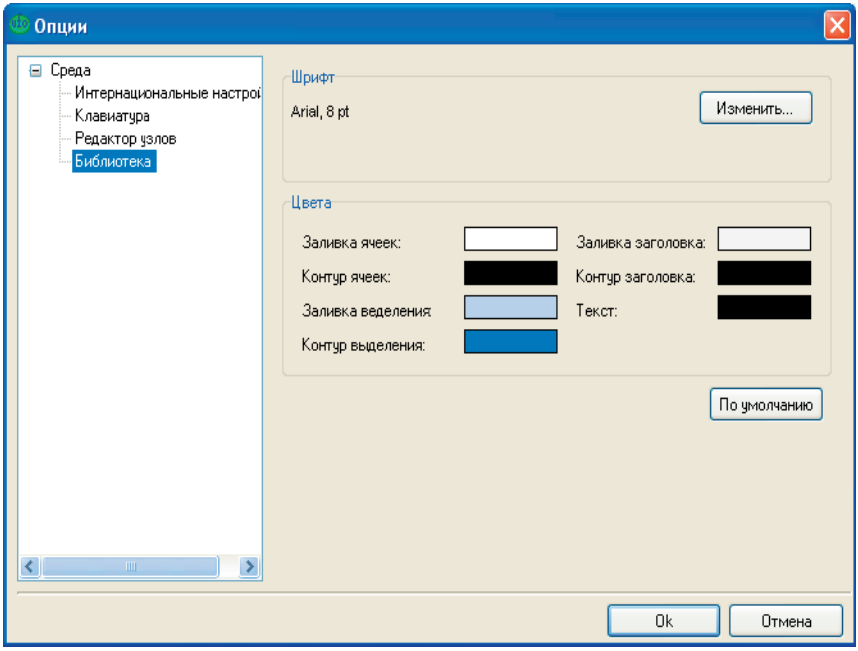


Рис. 3.52. Внешний вид диалогового окна настройки опций

Пункт «Клавиатура» позволяет задать комбинацию горячих клавиш для любой из команд программы.

Пункт «Редактор узлов» позволяет изменять стилистическое оформление редактора узлов (вкладка «узел» на панели свойств).

Пункт «Библиотека» позволяет изменять стилистическое оформление библиотек.

3.2.3 Выбор окон

С помощью пункта меню «Окно > Окна…» вызывается диалоговое окно, содержащее список открытых окон для их просмотра и обработки. Внешний вид этого окна представлен на рисунке 3.53.

С помощью данного окна пользователь может просмотреть, закрыть и активизировать любые окна из списка.

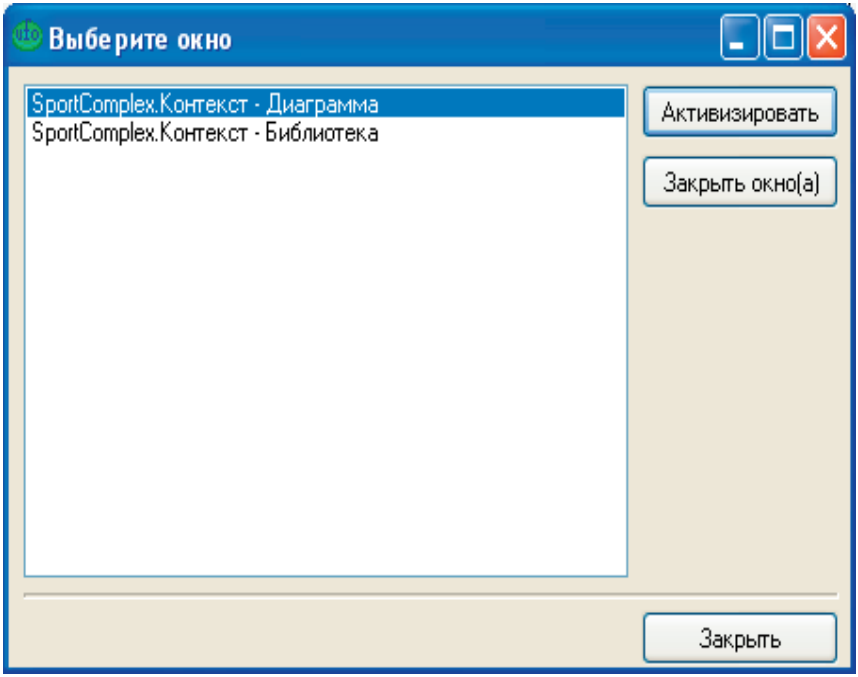


Рис. 3.53. Внешний вид диалогового окна, для работы с окнами диаграмм и библиотек

3.2.4 Выделить объекты

При работе с диаграммой на динамической панели инструментов появляется кнопка «Выделить объекты». При нажатии на неё вызывается диалоговое окно, представленное на рисунке 3.54.

Этот диалог позволяет просмотреть существующие на диаграмме элементы. Можно просматривать как все элементы, так и отдельно связи, порты или узлы. Можно просматривать только элементы, содержащие определённый текст и инвертировать выделение. В выведенном списке элементов можно выделить необходимые элементы. После этого по нажатию кнопки «Выделить» диалог закроется, и выбранные элементы будут выделены на диаграмме. Диалог может быть очень полезен при необходимости изменения стиля или переноса определённых элементов.

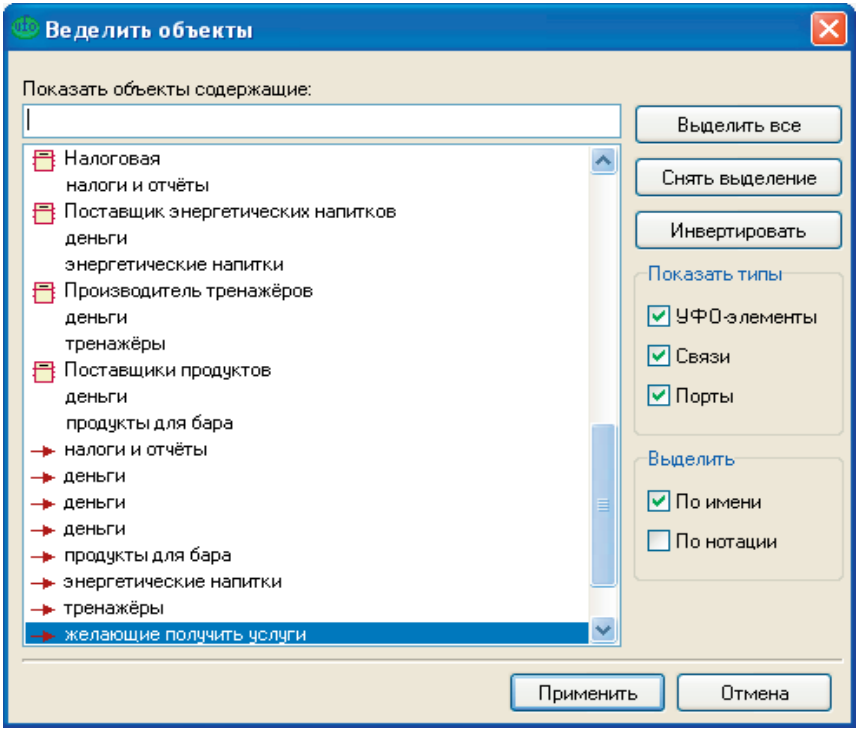


Рис. 3.54. Диалоговое окно для выделения объектов

3.3 Контекстные меню

В программе также существуют контекстные меню. Они вызываются нажатием правой кнопки мыши в определённом месте программы и служат для выполнения наиболее необходимых функций. Они позволяют пользователю в конкретной ситуации не искать команды в меню или кнопку на панели, а сразу вызвать список возможных действий и дополнительных функций.

3.3.1 Контекстные меню браузера

Браузер располагает рядом контекстных меню. Различные меню выводятся для различных элементов браузера.

Нажатие правой кнопкой мыши на названии проекта выводит меню, состоящее из следующих команд: Добавить новую модель, Добавить существующую модель, Закрыть проект, Сохранить, Сохранить как…, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии модели: Сделать текущей, Сохранить, Сохранить как…, Удалить из проекта, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На заголовке «Библиотеки: Создать библиотеку…, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии библиотеки: Создать библиотеку…, Открыть, Удалить, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На заголовке «Диаграммы»: Создать диаграмму…, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии диаграммы: Открыть, Удалить, Построить библиотеку…, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии Узла: Создать функцию..., Удалить, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии Функции: Сделать текущей, Создать объект..., Открыть, Удалить, Построить библиотеку…, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии объекта: Сделать текущим, Удалить, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

На названии Связи: Создать связь…, Удалить, Вырезать, Копировать, Вставить, Надписи, Развернуть всё, Свернуть всё.

Часть данных команд совпадает с командами основного меню. Функции других команд приведены в таблице 3.14.

Таблица 3.14

Функции команд контекстных меню браузера

Сделать текущим	Делает выбранный элемент текущим.
Открыть	Открывает данную диаграмму, функцию или библиотеку в новом окне
Надписи	Позволяет выбрать один из 3 вариантов надписей: только имя, только нотация и детальное описание
Развернуть всё	Открывает все элементы, находящиеся ниже в иерархии
Свернуть всё	Сворачивает все элементы, находящиеся ниже в иерархии
Создать библиотеку…	Создаёт новую библиотеку в отдельном окне. При этом предварительно запрашивается имя
Создать диаграмму…	Создаёт новую диаграмму в отдельном окне, При этом предварительно запрашивается имя
Создать функцию…	Создаёт новую функцию для выбранного узла
Создать объект…	Создаёт новый объект для выбранной функции

3.3.2 Контекстное меню диаграммы

При щелчке на самой диаграмме вызывается контекстное меню диаграммы. Это меню описывается в таблице 3.15.

Таблица 3.15

Контекстное меню диаграммы

Добавить UFO-элемент	Добавляет новый UFO-элемент в диаграмму
Удалить	Удаляет активный элемент
Открыть	Показывает содержимое выбранного UFO-элемента в этом же окне
Открыть в новом окне	Показывает содержимое выбранного UFO-элемента в новом окне
На уровень вверх	Показывает содержимое диаграммы верхнего уровня
Сгруппировать связи	Группирует выделенные связи в одну и даёт ей название ближайшего общего элемента в иерархии
Разгруппировать связи	Разгруппирует сгруппированные ранее связи
Экспортировать диаграмму	Позволяет экспортировать диаграмму в файл в формате bmp или jpg или поместить её в буфер обмена
Вырезать	Помещает выделенные объекты в буфер обмена и при этом удаляет их из текущей модели
Копировать	Помещает выделенные объекты в буфер обмена и при этом оставляет их без изменения в текущей модели (делает копию)
Вставить	Помещает объекты, находящиеся в буфере обмена, в активную область текущей модели (если такая вставка возможна)
Масштаб	Позволяет выбрать масштаб в котором показывается диаграмма в %
Надписи	Позволяет выбрать один из 3 вариантов надписей: только имя, только нотация и детальное описание
Содержимое узлов	Позволяет выбрать надпись на узле из следующих вариантов: текущие функции, описания и только имена

3.3.3 Контекстное меню библиотеки

При щелчке на самой библиотеке вызывается контекстное меню библиотеки. Это меню описывается в таблице 3.16.

Таблица 3.16

Контекстное меню библиотеки

Добавить UFO-элемент	Добавляет новый UFO-элемент в библиотеку
Открыть	Показывает содержимое выбранного UFO-элемента в текущем окне
Открыть в новом окне	Открывает новое окно диаграммы и показывает содержимое выбранного UFO-элемента в этом окне
Удалить функцию	Удаляет функцию
Удалить	Удаляет активный элемент
Экспортировать библиотеку	Позволяет экспортировать диаграмму в файл в формате bmp или jpg или поместить её в буфер обмена
Вырезать	Помещает выделенные объекты в буфер обмена и при этом удаляет их из текущей модели
Копировать	Помещает выделенные объекты в буфер обмена и при этом оставляет их без изменения в текущей модели (делает копию)
Вставить	Помещает объекты, находящиеся в буфере обмена, в активную область текущей модели (если такая вставка возможна)
Надписи	Позволяет выбрать один из 3 вариантов надписей: только имя, только нотация и детальное описание
Опции…	Открывает диалоговое окно опций, описанное выше
Показывать пустые линии	Позволяет скрыть в таблице библиотеки незаполненные поля

3.3.4 Контекстное меню редактора стилей

Каждый UFO-элемент имеет, изображённый на диаграмме, имеет свой стиль, Этот стиль может быть изменён при помощи редактора стилей, который расположен на панели свойств, Практически все его функции вызываются при помощи контекстного меню. Функции этого меню представлены в таблице 3.17.

Таблица 3.17

Контекстное меню редактора стилей

Создать стиль…	Создаёт новый стиль (цветовое и стилистическое оформление) на основе 1 из существующих стилей
Назначить	Назначает выбранный стиль текущим для выбранного UFO-элемента
Изменить…	Редактирование выбранного стиля
Удалить	Удаляет выбранный стиль
Сортировать по имени	Предоставляет возможность сортировать или не сортировать стили по алфавиту

4.1.3. Представление диаграмм в нотациях DFD, IDEF0 и BPMN с помощью системно-объектных моделей

Нотация и методика системно-объектного моделирования в терминах «Узел-Функция-Объект» позволяют создавать модели (УФО-модели), соответствующие диаграммам в нотациях DFD, IDEF0 и BPMN (см., например, работу [Зимовец, 2011]).

Для создания УФО-моделей, соответствующих диаграммам DFD, установим соответствие между графическими элементами DFD-нотации и УФО-моделей (см. таблицу 3.18).

Таблица 3.18

Соответствие элементов DFD и УФО

Описание элемента	*Графические элементы DFD в нотации Гейна-Сарсона*	*Элементы УФО-модели*
Поток данных Используется для моделирования передачи информации (или даже физических компонент) из одной части системы в другую	«имя»	Имя
Процесс Используется для моделирования процесса преобразования входного потока в выходной	«имя» «номер»	Номер Имя
Хранилище (накопитель) данных Используется для моделирования данных (или даже физических компонент), которые будут сохраняться между процессами	«имя»	Хранилище Имя
Внешняя сущность (терминатор) Используется для моделирования сущностей вне системы (контекстных сущностей), являющихся источником или приемником системных данных	«имя»	Имя

Рассмотрим пример модели в нотации DFD (см. рис. 3.55 и 3.56).

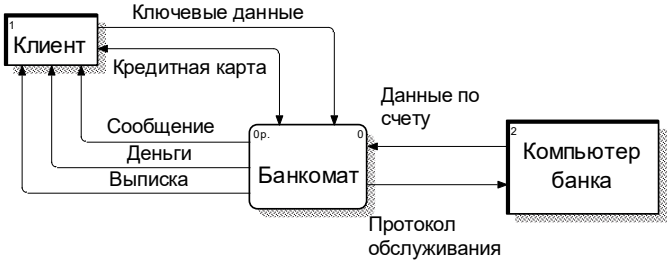


Рис. 3.55. Пример контекстной диаграммы в нотации DFD

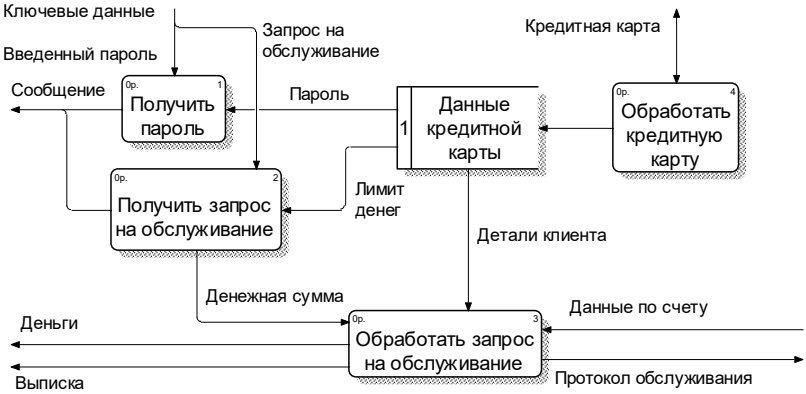


Рис. 3.56. Пример диаграммы декомпозиции в нотации DFD

Преобразуем представленные на рисунках 3.55 и 3.56 DFD-диаграммы в УФО-модели, используя соответствия между графическими элементами, показанные в таблице 3.18. Результаты представлены на рисунках 3.57 и 3.58 соответственно.

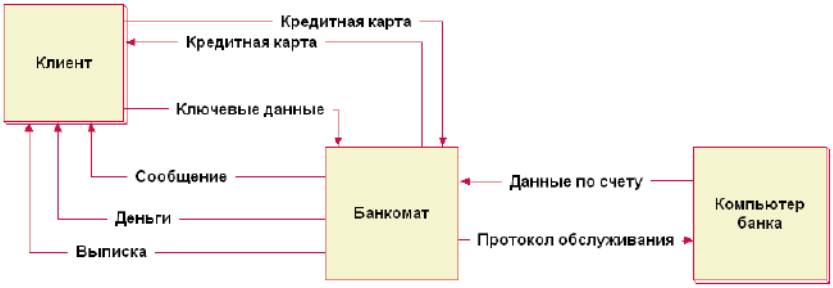


Рис. 3.57. Диаграмма на рис. 3.55 в виде модели «Узел-Функция-Объект»

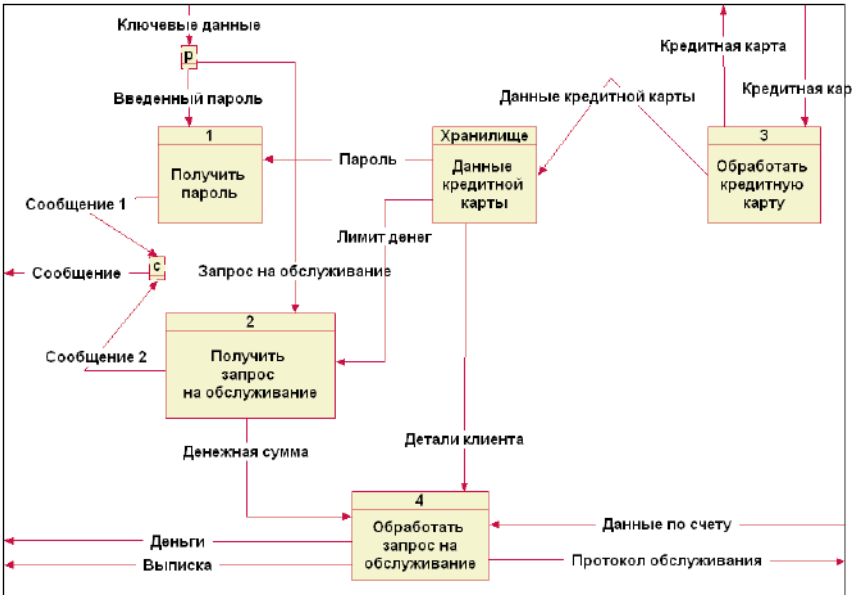


Рис. 3.58. Диаграмма на рис. 3.56 в виде модели «Узел-Функция-Объект»

В результате выполненного преобразования можно утверждать, что УФО-модель будет соответствовать DFD-диаграмме если в ней:

- для всех УФО-элементов определены функции;
- для контекстных УФО-элементов определены еще и объекты;
- выделен специфический УФО-элемент, представляющий собой функциональный узел для отображения какого-либо хранилища;
- введены служебные УФО-элементы, определенные только на уровне узлов, для обеспечения соединения и разветвления потоков.

Приведенный пример преобразования DFD-диаграмм в УФО-модели позволяет утверждать, что данное преобразование осуществляется просто и естественно. В результате данного преобразования снижается влияние упомянутых при описании DFD-диаграмм недостатков. Последнее обстоятельство обусловлено, во-первых, тем, что в УФО-элементах «Узлы» представляют собой абстрактные классы, «Функции» – конкретные классы, а «Объекты» – экземпляры этих классов. Это позволяет УФО-моделям поддерживать объектно-ориентированное проектирование. Во-вторых, тем, что в рамках УФО-подхода используется формально-семантическая нормативная система, семантика символов которой задается классификацией элементов

конкретной предметной области. Это обеспечивает «лишение разработчиков той части творческих возможностей, которые ведут к разнообразию представления организационных моделей».

Для создания УФО-моделей, соответствующих диаграммам IDEF0, установим соответствие между графическими элементами IDEF0-нотации и УФО-моделей (см. рис. 3.59 и 3.60).

Рассмотрим пример преобразования модели в нотации IDEF0 в УФО-модель. Воспользуемся моделью в нотации IDEF0, представленной на рисунке 3.11. Результат преобразования представлен на рис. 3.61 и 3.62.

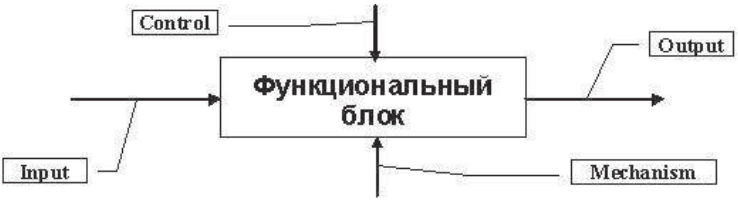


Рис. 3.59. Функциональный блок в нотации IDEF0

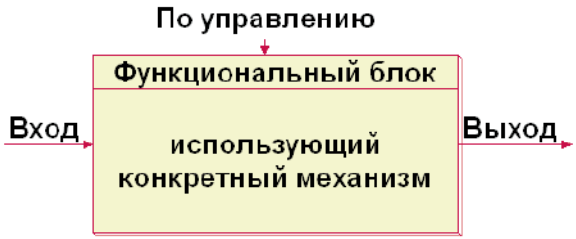


Рис. 3.60. Функциональный блок IDEF0 в виде модели «Узел-Функция-Объект»

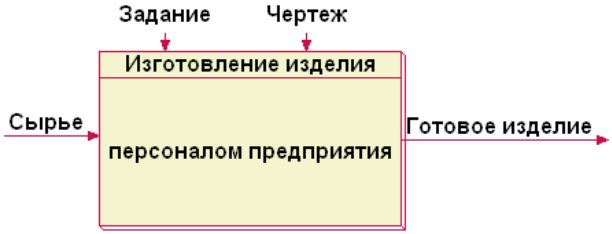


Рис. 3.61. Контекстная диаграмма на рис.3.11 в виде модели «Узел-Функция-Объект»

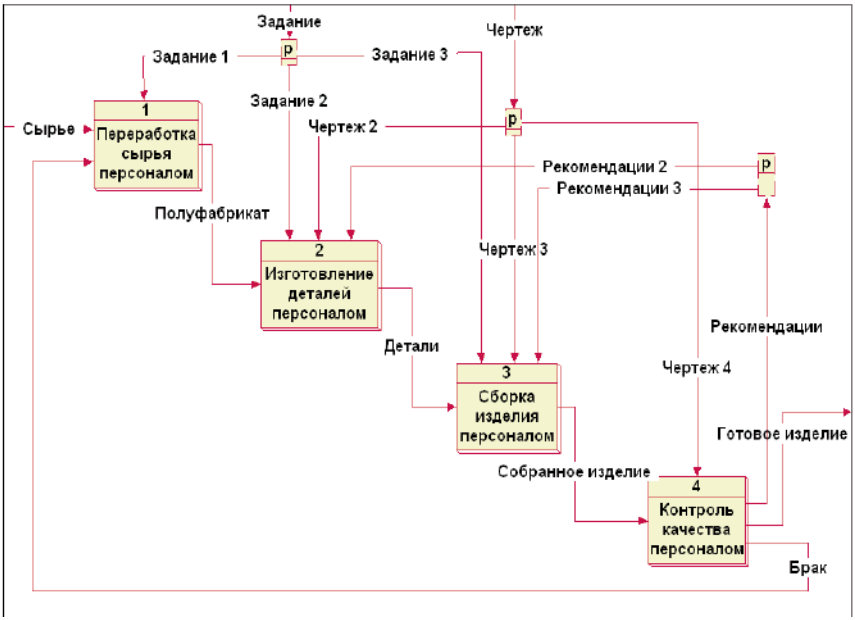


Рис. 3.62. Диаграмма декомпозиции на рис.3.11 в виде модели «Узел-Функция-Объект»

В результате выполненного преобразования можно утверждать, что УФО-модель будет соответствовать IDEF0-диаграмме если в ней:

- для всех УФО-элементов определены функции;
- для всех УФО-элементов определены объекты и их определения соответствуют связи «Механизм»;
- нижняя граница УФО-элемента для прикрепления связи не используется;
- все управляющие связи прикрепляются только к верхней границе УФО-элемента;
- для входов в УФО-элементы используется только левая граница;
- для выходов из УФО-элементов используется только правая граница;
- введены служебные УФО-элементы, определенные только на уровне узлов, для обеспечения соединения и разветвления потоков.

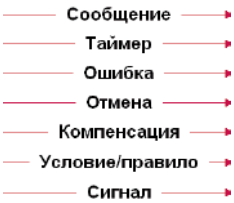
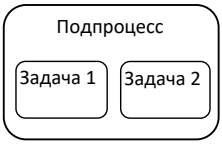
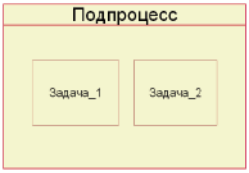
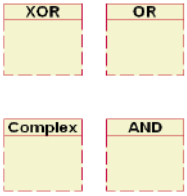
Приведенный выше пример преобразования IDEF0-диаграмм в УФО-модели позволяет утверждать, что данное преобразование также, как и преобразование DFD-диаграмм в УФО-модели осуществляется достаточно просто и естественно. В результате данного преобразования

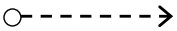
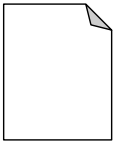
также снижается влияние упомянутых выше недостатков по тем же самым причинам. В дополнение к сказанному выше можно отметить возможность получать более «прозрачные» диаграммы за счет экономии на связях типа «механизм».

Для создания УФО-моделей, соответствующих диаграммам BPMN, установим соответствие между графическими элементами BPMN и УФО-моделей (см. таблицу 3.19).

Таблица 3.19

Соответствие элементов BPMN и УФО

Описание элемента	*Основные графические элементы BPMN*	*Элементы УФО-модели*
Событие (Event) Событие – это то, что происходит в течение бизнес-процесса и оказывает влияние на его ход. Чаще всего событие имеет причину (триггер) или воздействие (результат). Согласно влиянию Событий на ход бизнес-процесса, выделяют три типа: Стартовое событие (Start), Промежуточное событие (Intermediate) и Конечное событие (End)	***Маркеры (триггеры) событий:*** -сообщение, -таймер, -ошибка, -отмена, -компенсация, -условие\правило, -сигнал 	
Действие (Activity) Действие – общий термин, обозначающий работу, выполняемую исполнителем. Действия могут быть либо элементарными, либо неэлементарными (составными). Выделяют следующие виды действий, являющихся частью модели Процесса: Процесс (Process), Подпроцесс (Sub-Process) и Задача (Task)		
Шлюз (Gateway) Шлюзы используются для контроля расхождений и схождений потока операций. Таким образом, данный термин подразумевает ветвление, раздвоение, слияние и соединение маршрутов. Внутренние маркеры указывают тип контроля развития бизнес-процесса	 **Типы шлюзов:** -Эксклюзивные ИЛИ (XOR); -ИЛИ (OR); -Комплексные (Complex); -И (AND).	

Описание элемента	*Основные графические элементы BPMN*	*Элементы УФО-модели*
Поток операций (Sequence Flow) Поток операций служит для отображения того порядка, в котором организованы действия Процесса		
Поток сообщений (Message Flow) Поток сообщений служит для отображения обмена сообщениями между двумя участниками, готовыми эти сообщения отсылать и принимать. На диаграмме BPMN два отдельно взятых Пула представляют собой двух участников процесса		
Объект данных (Data Object) Объекты данных рассматриваются как артефакты, так как они не влияют непосредственно на последовательный поток или поток сообщений процесса, но они обеспечивают ввод информации о том, какие действия требуют выполнения и/или что они производят		

Используем пример модели в нотации BPMN, представленный на рис. 3.37. Преобразуем представленную на рисунке 3.37 BPMN-диаграмму в УФО-модель, используя соответствия между графическими элементами, показанное в таблице 3.19 Результаты представлены на рисунках 3.63–3.65.

Приведенный выше пример преобразования BPMN-диаграммы в УФО-модель позволяет утверждать, что данное преобразование вполне выполнимо с учетом особенностей соответствующего программного инструментария.

В результате выполненного преобразования можно утверждать, что УФО-модель будет соответствовать BPMN-диаграмме если в ней:

- в классификацию, в категорию связей «По управлению (С)» введен абстрактный класс связей «Событие», разделенный на подклассы связей, соответствующие маркерам (триггерам) событий (так как элемент «Событие» в нотации BPMN, по сути дела, представляет связи/потоки или поступающие на обработку (на вход процесса), или генерируемые процессом (поступающие на выход));

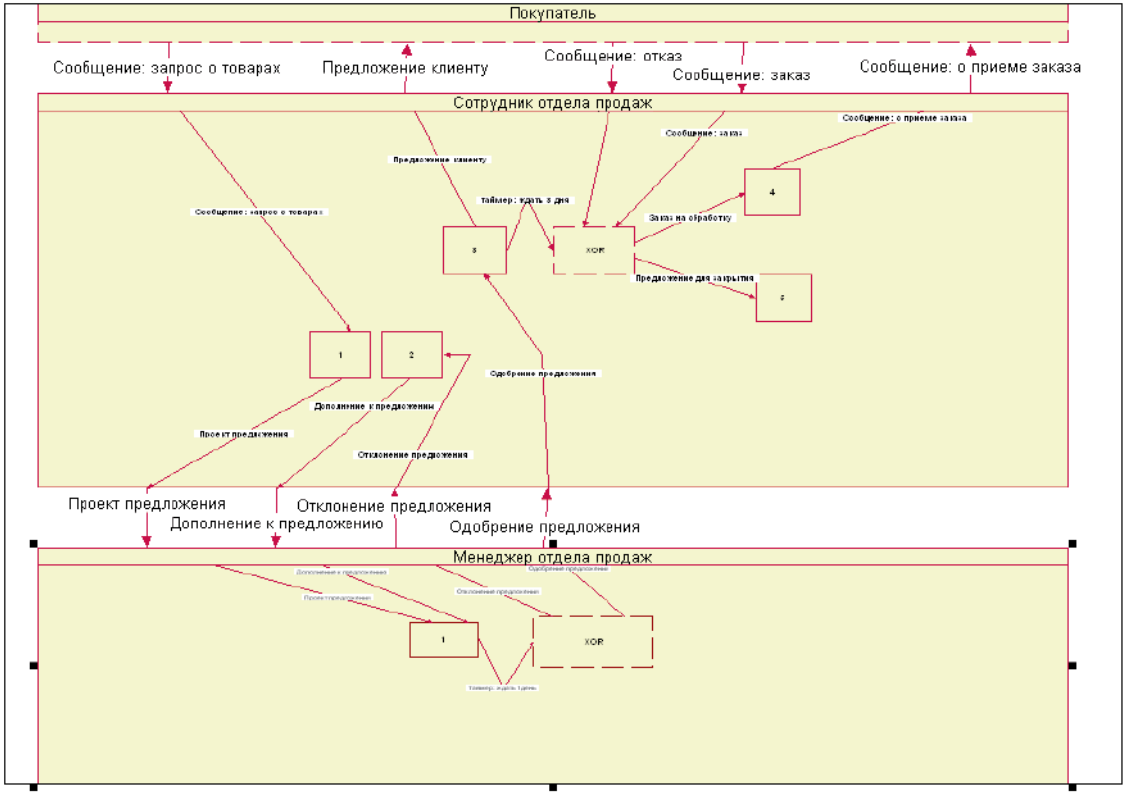


Рис. 3.63. Диаграмма на рис. 3.48 в виде модели «Узел-Функция-Объект»

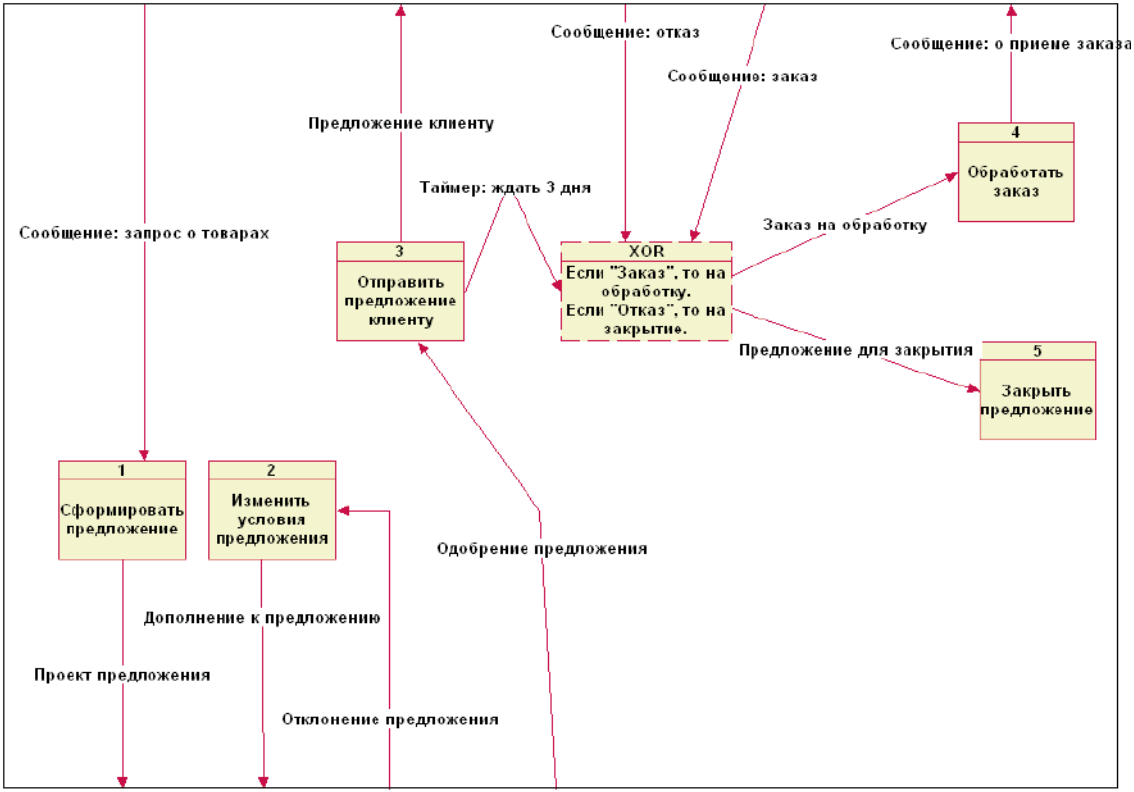


Рис. 3.64. Дополнение к диаграмме на рис.3.49: «Сотрудник отдела продаж»

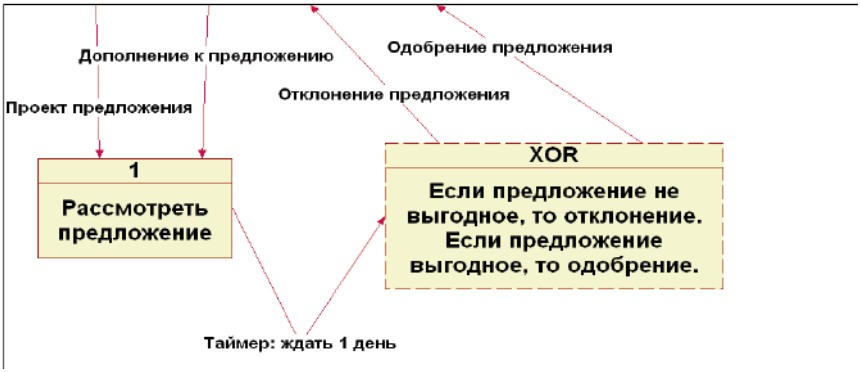


Рис. 3.65. – Дополнение к диаграмме на рис.3.49: «Менеджер отдела продаж»

- УФО-элементы в модели определены на уровне функций;
- введены специальные/служебные УФО-элементы, определенные на уровне узлов, обозначающие логические операции, обеспечивающие схождение и расхождение потоков;
- все потоки в BPMN-модели (операций и сообщений) представляются в УФО-модели связями из классификации (так как действия в процессах не могут просто так переходить одно в другое, они всегда обмениваются материей и информацией);
- элемент BPMN-модели «Объект данных» представляется в УФО-модели определенного вида связью из категории связей «По данным (D)»;
- пулы и дорожки BPMN-диаграммы представляются в УФО-модели УФО-элементами, определенными на функциональном уровне.

4.2. Имитационное системно-объектное моделирование

4.2.1. Нотация и методика имитационного системно-объектного моделирования

Рассмотрим подробнее процедуру имитационного системно-объектного моделирования [Жихарев, 2014–2016; Zhikharev, 2018; Маторин, 2015] на примере разработки имитационной модели распространения подземных вод в окрестностях горно-обогатительных комбинатов с использованием исчисления систем как функциональных объектов, а также программного инструментария UFOModeler.

При проведении анализа влияния горнодобывающей промышленности на подземные воды был применен метод имитационного системно-объектного моделирования, с помощью которого была построена модель знаний о динамике уровня подземных вод.

Рассмотрим иерархию потоковых объектов разрабатываемой имитационной модели. Выделим в модели вещественный потоковый объект, представляющий собой водную массу, протекающую между точками маршрута в единицу времени. Данный потоковый объект будет характеризоваться следующими параметрами:

$$Q=[q_1,h_1,h_2,v], \tag{3.1}$$

где: q_1–объем воды, в кубических метрах; h_1 – верхняя граница потока (расстояние от поверхности земли до водяного зеркала в скважине, в метрах); h_2 – нижняя граница потока (расстояние от поверхности земли до нижней границы водоносного горизонта, в метрах); v – скорость движения подземных вод (скорость фильтрации), в метрах в секунду. Данный потоковый объект имеет вещественный тип в иерархии потоковых объектов модели. Фактически это ключевой потоковый объект, который моделирует движение воды в определенной области. Для моделирования процессов водозабора из скважин, и дренажных шахт ГОКов, определим отдельный потоковый объект родитель следующего вида:

$$Водоотбор=[q,q_{max}], \tag{3.2}$$

где: q–объем воды, отобранный из скважины за сутки, в кубических метрах; q_{max} – максимальный возможный объем отбора воды из скважины в сутки, в кубических метрах. Для данного потокового объекта введем потоковые объекты, которые соответствуют технологическим объектам водозаборов, как показано на рисунке 3.66.

Для моделирования влияния климатических условий, введем в модель управляющий потоковый объект, моделирующий изменение климатических условий следующего вида:

$$Климат=[осадки,\ засуха,\ сезон], \tag{3.3}$$

Поля *Климат.осадки* и *Климат.засуха* хранят значение логического типа данных, а поле *Климат.сезон* принимает целое значение от 1 до 4, где 1- зима, 2-весна, 3-лето, 4- осень.

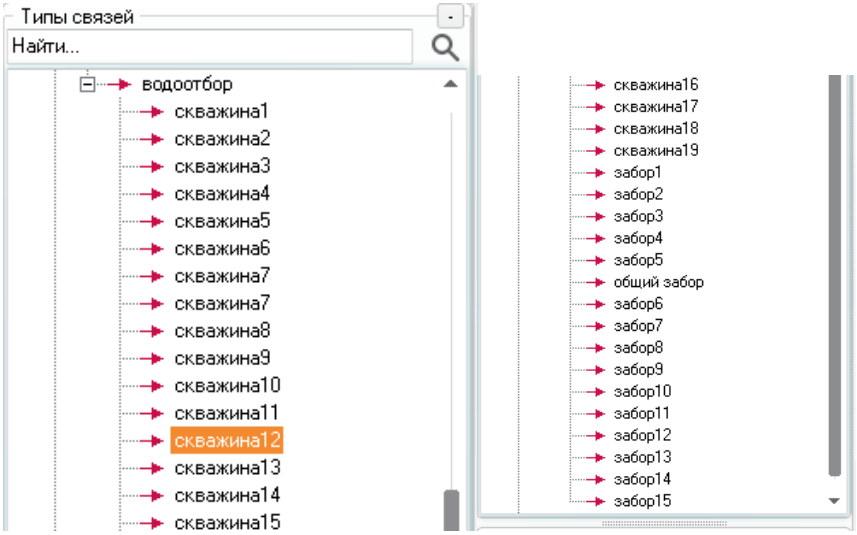


Рис. 3.66. Фрагмент иерархии потоковых объектов для Ильинского водозабора

Далее рассмотрим узловые объекты модели. В качестве узловых объектов будем рассматривать следующие виды техногенных сооружений:

1. Водозаборы, находящиеся на территории Старооскольского и Губкинского городского округа Белгородской области:
   1) Ильинский водозабор.
   2) Воротниковский водозабор.
   3) Незнамовский водозабор.
   4) Бор-Анпиловский водозабор.
   5) Гуменский водозабор.
   6) Водозабор севернойпромкомзоны.
   7) Водозабор МУП «ОЖКХ».
   8) Водозабор «Лебеди».
   9) Водозабор п. Троицкий.
   10) Водозабор «Теплый колодезь».
   11) Водозабор «Парковый».
   12) Водозабор «Городской парк».
   13) Водозабор «Яр-Кучугуры».
2. Техногенные сооружения, относящиеся к инфраструктуре Стойленского и Лебединского горно-обогатительных комбинатов:
   1) СГОК.

2) ЛГОК.
3) Хвостохранилище СГОКа.
4) Хвостохранилище ЛГОКа.

В качестве управляющих узловых объектов будем рассматривать:

1) Климатический фактор.
2) Забор воды.

Представленные узловые объекты, опишем с помощью формальных основ исчисления систем как функциональных объектов.

Рассмотрим узловой объект «Климатический фактор», который в модели играет роль генератора погодных условий. Формально, данный узловой объект представляет собою следующее выражение:

$$«Климатический\ фактор»=[L_?,L_!;f(L_?)L_!;O_?,\ O_!,\ O_f], \qquad (3.4)$$

где $L_?=\varnothing$; $L_!=[кл.усл.1,\ кл.усл.2,\ \dots\ ,\ кл.усл.n]$; $O_?=\varnothing$; $O_!=\varnothing$; $O_f=\varnothing$. Множество входных потоковых объектов данного узлового объекта равно пустому множеству. Множество выходных потоковых объектов содержит n выходящих потоковых объектов, где n – количество водозаборов. Т.е. узловой объект «*Климатический фактор*» генерирует погодные условия для каждого узлового объекта, моделирующего техногенные водозаборы. Объектные характеристики в данном случае не используются.

Рассмотрим подробнее метод данного узлового объекта.

$$f(t)L_! = \begin{cases} кл.усл.1.сезон: кл.усл.1.сезон = 1;\ delay = t; \\ \dots \\ кл.усл.n.сезон: кл.усл.n.сезон = 1;\ delay = t; \\ кл.усл.1.сезон: кл.усл.1.сезон = 2;\ delay = t; \\ \dots \\ кл.усл.n.сезон: кл.усл.n.сезон = 2;\ delay = t; \\ кл.усл.1.сезон: кл.усл.1.сезон = 3;\ delay = t; \\ \dots \\ кл.усл.n.сезон: кл.усл.n.сезон = 3;\ delay = t; \\ кл.усл.1.сезон: кл.усл.1.сезон = 4;\ delay = t; \\ \dots \\ кл.усл.n.сезон: кл.усл.n.сезон = 4;\ delay = t; \\ t = 7776000; n \in [1:15]. \end{cases} \qquad (3.5)$$

Таким образом, метод данного узлового объекта будет по истечении одного месяца (среднее количество дней – 30, в секундах – 7776000) изменять номер сезона в выходящих потоковых объектах от

1 до 4, что моделирует смену сезона, от которого зависит вероятность осадков или засухи.

Далее рассмотрим узловой объект соответствующий Ильинскому водозабору. Данный узловой объект на вход принимает управляющий потоковый объект *«кл.усл.1»* на выходе экземпляр потокового объекта – *«Q»* и *«Водоотбор»*.

Таким образом, формально, узловой объект, соответствующий Ильинскому водозабору имеет следующий вид:

$$«Ильинский\ водозабор»=[L_?,L_!;f(L_?)L_!;O_?,\ O_!,\ O_f], \quad (3.6)$$

где $L_?=[кл.усл.1];\ L_!=[Q,водоотбор\_1];\ O_?=\varnothing;\ O_!=\varnothing;\ O_f=\varnothing$.

Метод узлового объекта является композицией узловых объектов нижнего уровня. Как говорилось выше, каждый водозаборный комплекс состоит из нескольких скважин с различными параметрами, рассмотрим структуру узлового объекта, соответствующего скважине Ильинского водозабора. Формально такой узловой объект имеет следующий вид:

$$«Скважина\ 1»=[L_?,L_!;f(L_?)L_!;O_?,\ O_!,\ O_f], \quad (3.7)$$

где $L_?=[кл.усл.1];\ L_!=[Q,водоотбор\_1];\ O_?=\varnothing;\ O_!=\varnothing;\ O_f=[q,\ h1,\ h2,\ h1_{max},\ q_{max},\ n,\ k,\ l,\ debit_{max}]$.

Данный узловой объект на вход принимает управляющий потоковый объект «кл.усл.1» на выходе экземпляр потокового объекта – «Q» и «Водоотбор». Рассмотрим структуру узлового объекта, соответствующего скважине Ильинского водозабора.

Рассмотрим подробнее объектные характеристики данного узлового объекта:

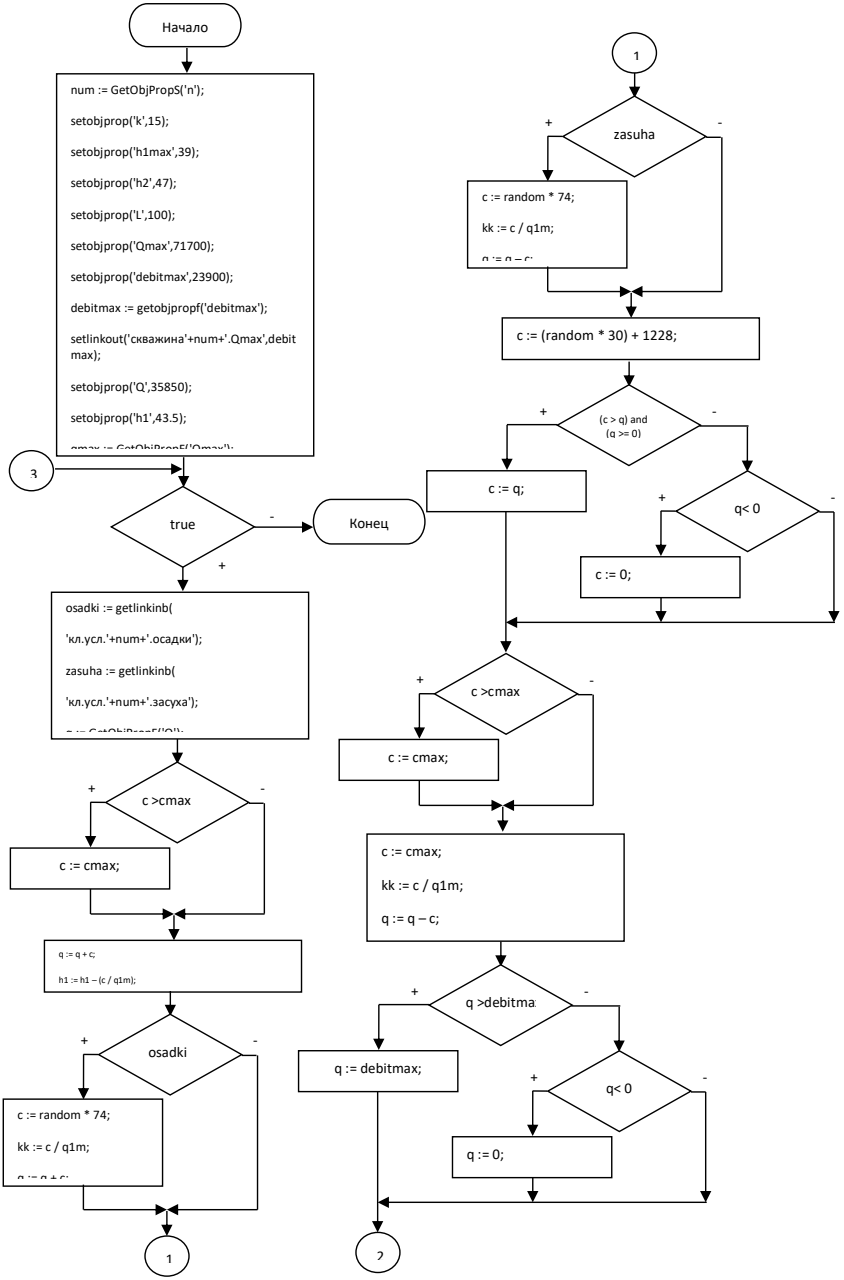
- *q* – количество воды, протекающей за сутки в текущей скважине (м3/сут);
- *h2* – нижняя граница водоносного горизонта в текущей скважине (м);
- *h1max* – максимальный уровень подземных вод (м);
- *qmax*– максимальный запас водоносного горизонта (м3);
- *n* – порядковый номер скважины текущего водозаборного комплекса;
- *k* – коэффициент фильтрации породы, сквозь которую протекает вода;
- *l* – расстояние между соседними скважинами;
- *debitmax* – максимальное количество откачиваемой воды;

- *currentDebit* – количество воды, откаченное за текущие сутки.

Значения, указанные в модели, могут быть изменены для показателей, перечисленных выше с целью осуществления различных экспериментов на модели.

Все представленные выше показатели характеризуют отдельную скважину. Рассмотрим подробнее показатель – коэффициент фильтрации. Данный коэффициент показывает скорость фильтрации при градиенте напора равном единице. Данный показатель зависит от типа почв, из которых состоит водоносный горизонт. Объектные характеристики скважины используются в описании метода данного узлового объекта, который в свою очередь, определяет процесс перетекания подземных вод через текущую точку.

Рассмотрим метод узлового объекта «Скважина 1» подробнее. Оператор random – генерирует случайное число в промежутке от нуля до единицы. По умолчанию уровень воды в скважине задается с помощью random в долях от 0,35 до 0,65 от максимального уровня воды в скважине. Переменная «Q» метода узлового объекта позволяет рассчитать суточный объем воды, откачиваемый из скважины в кубических метрах. Согласно исходным данным, значение показателя суточного объема откачиваемой воды «currentDebit» для данной скважины варьируется в пределах от 1228 до 1258 кубических метров, при условии, что значение показателя «currentDebit» установлено в нуль. Также имеется возможность принудительной установки объема откачиваемой воды «currentDebit» до необходимых значений, в таком случае количество откачиваемой воды по умолчанию будет игнорироваться. Переменная «h1» содержит рассчитанное значение верхней границы водоносного горизонта. Переменная «F» – представляет собой площадь поперечного сечения скважины в квадратных метрах. Переменная «h2» записывает нижнюю границу водоносного горизонта, причем для конкретной скважины данный показатель неизменен. Переменная «v» позволяет рассчитать, в соответствие с законом Дарси скорость движения подземной воды. Далее осуществляется задержка, равная одним суткам реального времени. Таким образом, данный метод будет срабатывать один раз в сутки модельного времени и осуществлять перерасчет интересующих показателей на основании текущих климатических условий и круглосуточного объема забора воды. Методы скважин всех водозаборов будут иметь подобную структуру за исключением значений объектных характеристик. На рисунке 3.67 представлена блок-схема алгоритма метода узлового объекта.


Начало
num := GetObjPropS('n');
setobjprop('k',15);
setobjprop('h1max',39);
setobjprop('h2',47);
setobjprop('L',100);
setobjprop('Qmax',71700);
setobjprop('debitmax',23900);
debitmax := getobjpropf('debitmax');
setlinkout('скважина'+num+'.Qmax',debitmax);
setobjprop('Q',35850);
setobjprop('h1',43.5);
3
true
Конец
osadki := getlinkinb(
'кл.усл.'+num+'.осадки');
zasuha := getlinkinb(
'кл.усл.'+num+'.засуха');
c >cmax
c := cmax;
q := q + c;
h1 := h1 – (c / q1m);
osadki
c := random * 74;
kk := c / q1m;
1
1
zasuha
c := random * 74;
kk := c / q1m;
c := (random * 30) + 1228;
(c > q) and (q >= 0)
c := q;
q< 0
c := 0;
c >cmax
c := cmax;
c := cmax;
kk := c / q1m;
q := q – c;
q >debitmax
q := debitmax;
q< 0
q := 0;
2

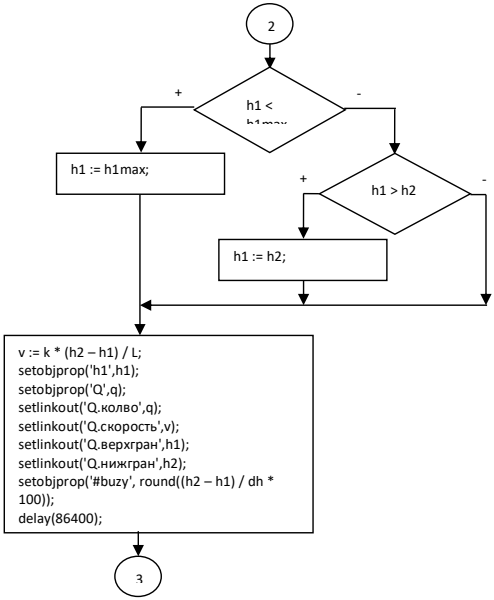


Рис. 3.67. Алгоритм метода узлового объекта «скважина» в терминах языка описания функциональных объектов

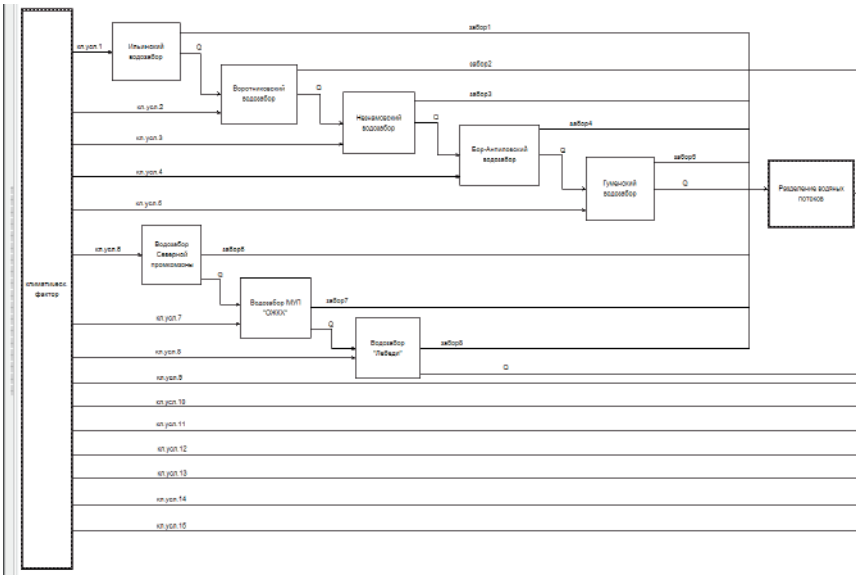


Рис. 3.68. Контекстная модель распространения подземных вод

Данный потоковый объект позволяет в модели учесть откачку воды из скважин в соответствие статистическими данными о средних отборах воды на отдельных участках водозаборов и горно-обогатительных комбинатов.

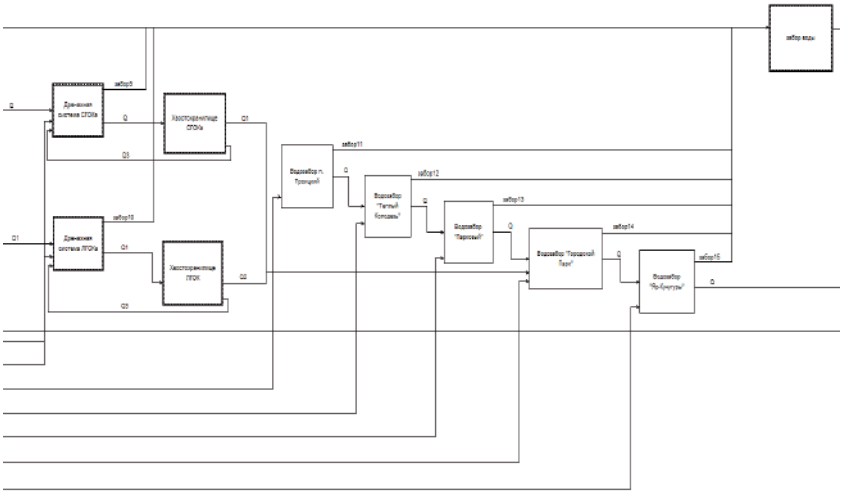


Рис. 3.69. Контекстная модель распространения подземных вод (продолжение)

Как видно из модели, представленной на рисунке 3.68 и рисунке 3.69, на контекстной диаграмме расположены: управляющий блок для генерации климатических условий, а также блоки, моделирующие техногенные сооружения, влияющие на уровень подземных вод. Блок *«Климатический фактор»* является генератором и одновременно генерирует на все водозаборы номер сезона. Климатические условия влияют на уровень подземных вод посредством соответствующего потокового объекта (см. рисунок 3.67). Так же данный блок моделирует естественное пополнение подземных вод осадками, просочившимися через почву. Последовательность расположения техногенных сооружений на модели отражает действительное расположение водозаборов, ГОКов и др. Поток подземных вод, как было описано выше, на модели представлен в виде потокового объекта *«Q»*. Данный потоковый объект моделирует направление движения подземных вод от Ильинского водозабора к Воротниковскому водозабору и т.д. Как видно из модели, из каждого блока, представляющего собой техногенный узел, выходит потоковый объект, моделирующий водозабор.

Каждый водозабор состоит из совокупности скважин, которые, собственно, и являются ключевыми техногенными сооружениями, влияющими на уровень подземных вод. Рассмотрим структуру Ильинского

водозабора, состоящего из 19 скважин. Причем все скважины расположены последовательно по ходу течения подземных вод в рассматриваемом водоносном горизонте. Для удобства скважины пронумерованы в порядке их монтажа по ходу движения подземных вод.

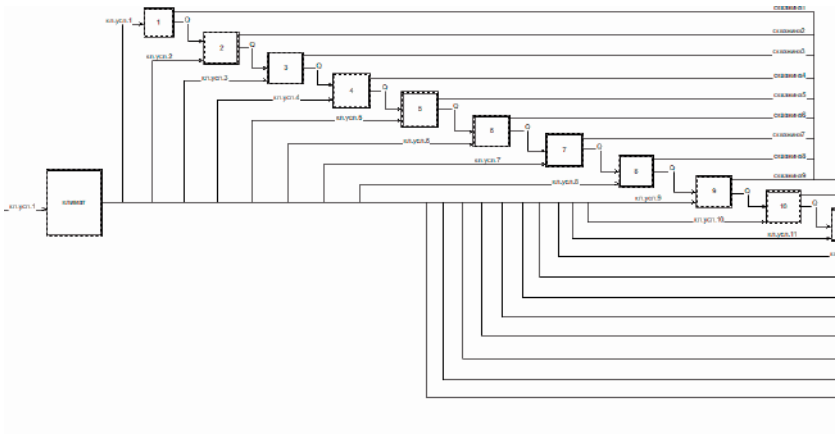


Рис. 3.70. Модель Ильинского водозабора

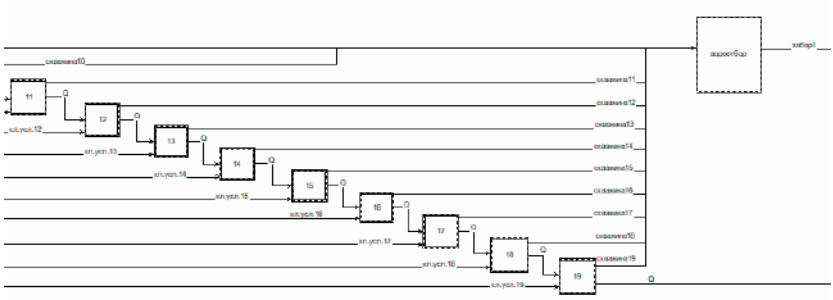


Рис 3.71. Модель Ильинского водозабора (продолжение)

Как видно из рисунка 3.70 и рисунка 3.71, все скважины имеют одинаковую структуру: в качестве входных потоковых объектов выступают водный поток и климатические условия, в качестве исходящих – водозабор и водный поток. Также на модели расположены два дополнительных блока. Первый блок – генерирует погодные условия в соответствие с текущим сезоном. Метод узлового объекта данного блока в соответствии со статистическими значениями вероятностей появления осадков или засухи в определенные сезоны, генерирует погодные условия в области Ильинского водозабора. Блок «Водоотбор» является блоком-сумматором, задача которого состоит в суммировании объемов откаченной воды в сутки со всех 19 скважин.

Далее рассмотрим систему откачки воды для осушения ГОКов. Модель откачки воды от горно-обогатительных комбинатов описана единым функциональным алгоритмом. Рассмотрим их работу на примере СГОКа. Данный узловой объект на вход принимает управляющий потоковый объект «*Q*» на выходе экземпляр потокового объекта – «*Q*» и «*Водоотбор*». Метод узлового объекта отражает работу дренажной системы ГОКа, производящую водозабор из всех водоносных слоев около комбината. На одном из этапов добычи из руды отделяют вскрышную (пустую) породу, смешивают с водой и получившуюся пульпу отправляют по пульпопроводу в хвостохранилище. Процесс забора воды и отправки пульпы в хвостохранилище отражен в методе объекта «*СГОК*».

Рассмотрим подробнее ключевые объектные характеристики данного узлового объекта:

- *qmax* – предельное количество воды поступающее из всех водоносных слоев в дренажную систему СГОКа (м3/сут);
- *q* – фактическое количество воды, откачиваемое дренажной системой СГОКа (м3/сут);
- *dolya_vozvrata* – доля воды, которая попадает обратно из хвостохранилища для осуществления тех. процессов СГОКа;
- *dolya_vody* – доля воды в пульпе, которая поступает в хвостохранилище;
- *dolya_goroda* – доля воды, забираемая на городские нужды;
- *q_pulpa* – количество пульпы, поступающее в хвостохранилище в сутки (м3/сут).

Значения, указанные в модели, могут быть изменены для показателей, перечисленных выше с целью осуществления различных экспериментов на модели, связанных с откачкой воды и использования технической воды для нужд в горно-обогатительном комбинате.

Рассмотрим метод узлового объекта *«СГОК»* подробнее. В основе метода лежит получение количества воды из соседнего водозабора альб-сеноманского водоносного горизонта. Также в методе задается значение откачки воды из всех водоносных слоев, стоит отметить, что при помощи оператора *random* имитируется количество отбираемой воды в диапазоне от 91000 до 92000 м^3/сут. На основе, отобранной воды, вычисляется количество, отбираемое на городские нужды (определено в объеме 85% от общего забора, остальные 15% отводятся на внутренние производственные процессы). Далее узловой объект «*СГОК*» обеспечивает перекачку в хвостохранилище пустой породы, разбавленной с водой в виде пульпы. Доля воды в пульпе определяется переменной «*dolya_vody*» и по умолчанию задана в размере 50%.

Метод узлового объекта «*ЛГОК*» функционально идентичен за исключением значений объектных характеристик. На рисунке 3.72 представлена блок-схема алгоритма метода узлового объекта.

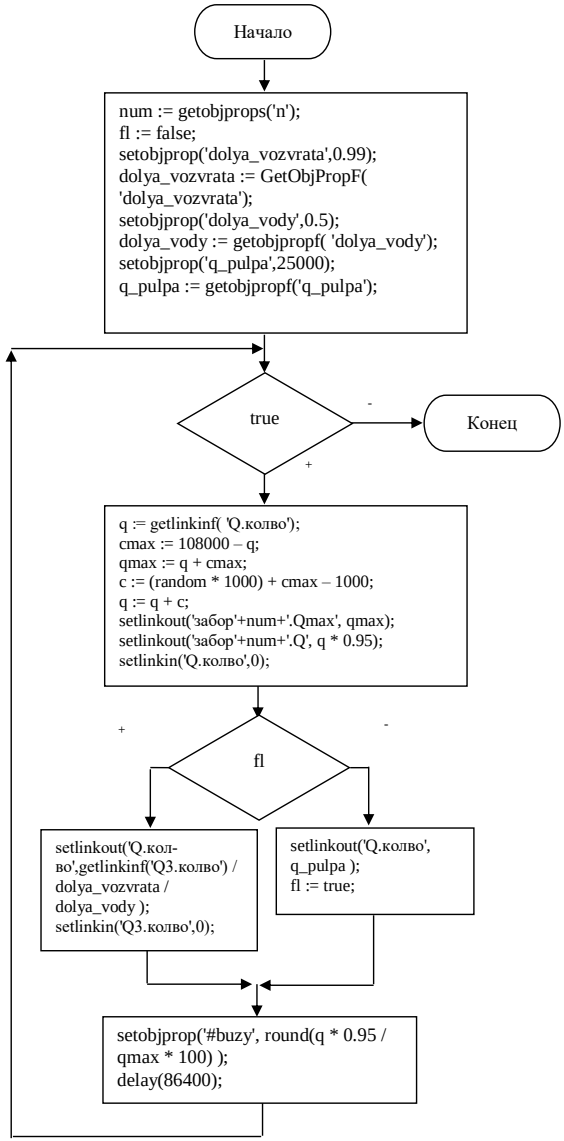


Рис. 3.72. Алгоритм метода узлового объекта «СГОК» в терминах языка описания функциональных объектов

На рисунке 3.73, на контекстной диаграмме расположены: блок «*СГОК*», функционально включающий дренажную систему и систему генерации пульпы. Блок *«Хвостохранилище СГОКа»* является приёмником пульпы от блока «*СГОК*».

Рассмотрим хвостохранилища ГОКов. Модель функционирования хвостохранилищ горно-обогатительных комбинатов описана единым функциональным алгоритмом. Рассмотрим их работу на примере хвостохранилища СГОКа. Данный узловой объект на вход принимает управляющий потоковый объект «*Q*» на выходе экземпляр потокового объекта – «*Q1*» и «*Q3*». Метод узлового объекта имитирует процесс отделения воды от пустой породы, складируемой в хвостохранилище, а также попадание очищенной от хвостов воды в производственный процесс комбината. Процесс функционирования хвостохранилища отражен в методе объекта «*Хвостохранилище СГОКа*».

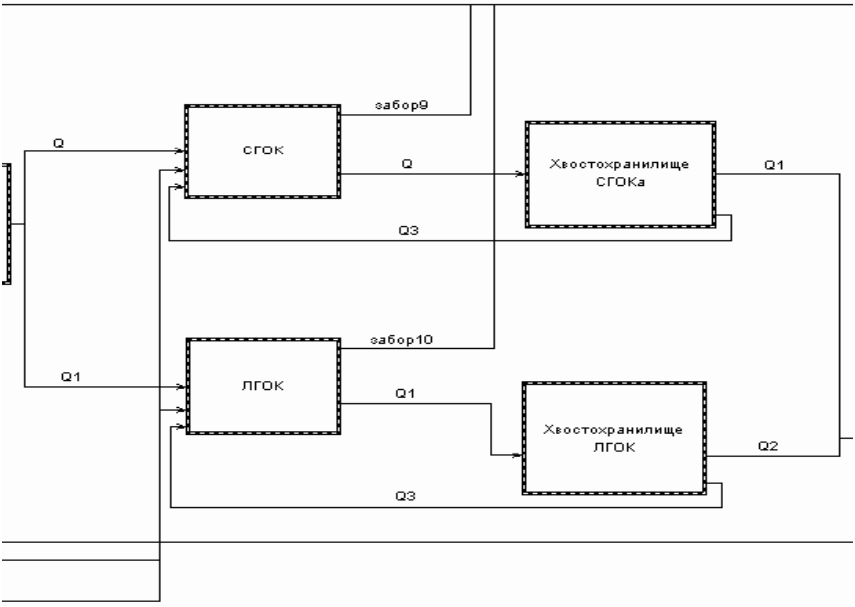


Рис. 3.73. Модель взаимодействия ГОКов и хвостохранилищ

Рассмотрим подробнее ключевые объектные характеристики данного узлового объекта:

- *shir* – ширина хвостохранилища (м);

- *dl* – длина хвостохранилища (м);
- *k* – коэффициент фильтрации породы сквозь которую протекает вода;
- *dolya_vozvrata* – доля воды, которая попадает обратно из хвостохранилища для осуществления тех. процессов СГОКа;
- *dolya_vody* – доля воды в пульпе, которая поступает в хвостохранилище;
- *h1max* – значение гидродинамического давления выше защитного слоя хвостохранилища (м);
- *h2* – значение гидродинамического давления ниже защитного слоя хвостохранилища (м);
- *l* – толщина защитного слоя хвостохранилища.

Значения, указанные в модели, могут быть изменены для показателей, перечисленных выше с целью осуществления различных экспериментов на модели, связанных с загрязнением почвы технической водой из хвостохранилища.

Все представленные выше показатели характеризуют отдельную скважину. Приведенные выше показатели являются компонентами формулы закона Дарси. В случае оценки водопроницаемости защитного слоя хвостохранилища закон Дарси стоит использовать, принимая *h1* и *h2* как гидродинамические давления выше и ниже оцениваемого защитного слоя, при этом параметр *l* отражает толщину оцениваемого слоя.

Рассмотрим метод узлового объекта *«Хвостохранилище СГОКа»* подробнее. В основе метода лежит получение пульпы из узлового объекта «*СГОК*». Затем имитируется отложение пустой породы в хвостохранилище согласно доле воды в пульпе (переменная *dolya_vody)* и возврат очищенной технической воды в узловой объект «*СГОК*». В переменной «*q*» осуществляется подсчет количества воды проникаемой в почву из хвостохранилища в сутки. В переменной «*v*» ведется подсчет скорости протекания воды. Метод узлового объекта «*Хвостохранилище ЛГОКа*» функционально идентичен за исключением значений объектных характеристик. На рисунке 3.74 представлена блок-схема алгоритма метода узлового объекта.

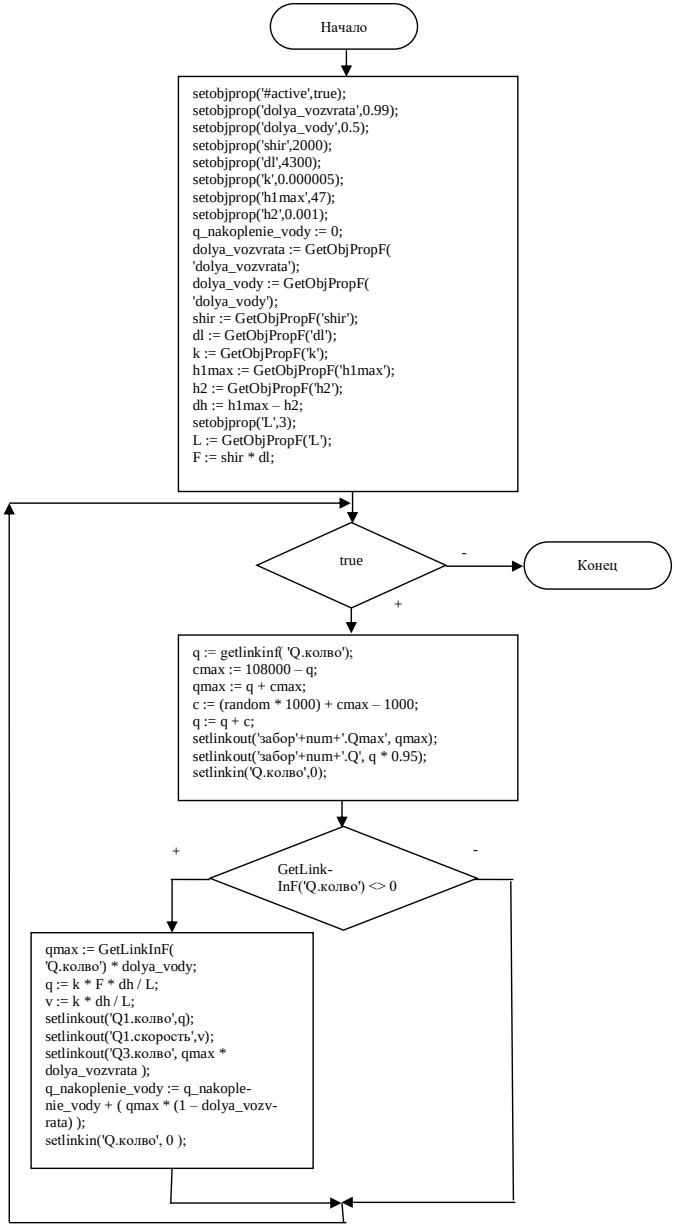


Рис. 3.74. Алгоритм метода узлового объекта «Хвостохранилище СГОКа» в терминах языка описания функциональных объектов

Рассмотрим имитацию процесса распространения подземных вод с применением разработанной модели. Для этого запустим модель с масштабом времени: 1 секунда реального времени равна 1 суткам модельного времени. На рисунке 3.75 показан процесс симуляции распространения подземных вод в заданных областях. Прямоугольные индикаторы на блоках – скважинах показывают текущую наполненность водоносного горизонта. Так же имеется возможность в режиме реального времени вывести промежуточные значения потоковых объектов модели.

Рассмотрим зависимость уровня подземных вод от сезонных климатических условий. Для этого мы запустили модель на исполнение в указанном выше варианте и вывели на график объем воды в 5 скважине за сутки и номер сезона.

Из графиков видно, что при наступлении лета (номер сезона = 3) вероятность засухи повышается, а, соответственно, уровень подземных вод снижается.

Проведем серию экспериментов на разработанной модели. В качестве первого эксперимента увеличим водозабор из скважины № 5 Ильинского водозабора. Для этого изменим значение показателя «*currentDebit*» с 1250 $м^3$/сут (то есть среднесуточный отбор) до 23330 $м^3$/сут (практически полное осушение). Рассмотрим изменения графиков количества и уровня воды в скважине при значительном увеличении количества забора воды (рис. 3.76).

Как видно из рисунка 3.76, на верхнем графике количество воды в скважине упало до минимальных значений, при этом на нижнем графике показано, что уровень воды упал (то есть расстояние уровня воды от поверхности земли стало больше).

Также увеличение количества откачиваемой воды из скважины №5 повлияло на уровень воды в соседних скважинах по направлению течения воды. На рисунке 3.77 отражено понижение уровня воды в соседних скважинах водозабора, о чем свидетельствуют индикаторы блоков, начиная со скважины №6.

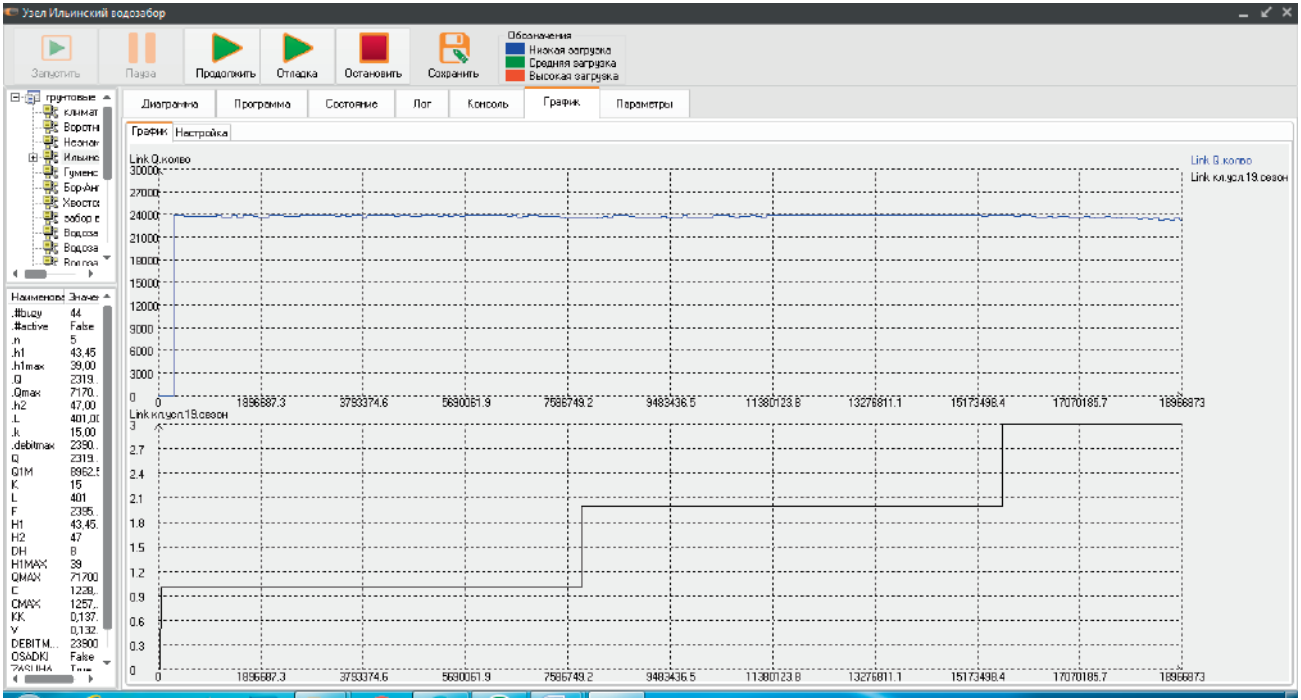


Рис. 3.75. График изменения объема подземных вод в зависимости от сезона

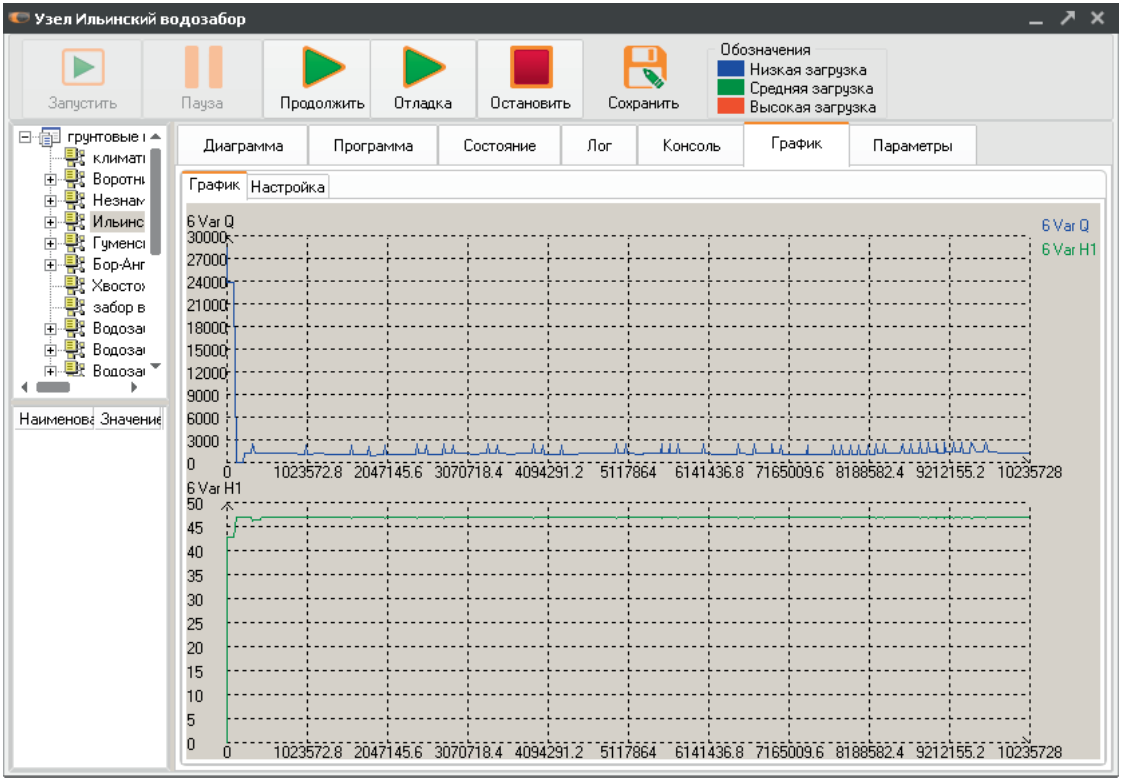


Рис. 3.76. График изменения уровня подземных вод
в зависимости от увеличения количества откачиваемой воды

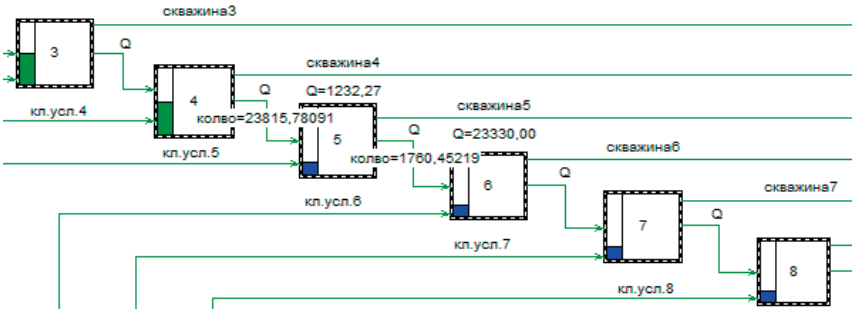


Рис. 3.77. Изменение уровня подземных вод в соседних скважинах при увеличении количества откачиваемой воды

Таким образом, увеличение забора воды в одной из скважин водозабора приводит к осушению скважин водозабора, находящихся по направлению течения воды.

Рассмотрим имитацию процесса водозабора через дренажную систему ГОКов. Для этого запустим модель с масштабом времени: 1 секунда реального времени равна 1 суткам модельного времени (рис. 3.78).

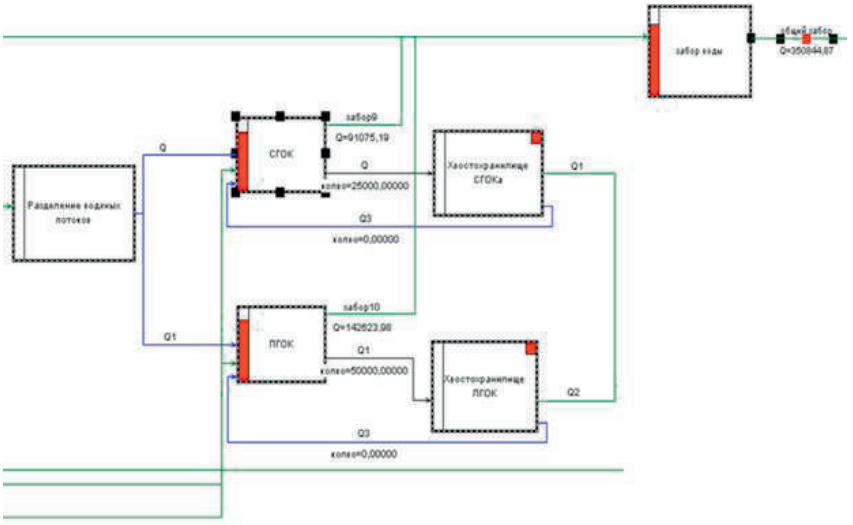


Рис. 3.78. Функционирование дренажных систем ГОКов и транспортировка пульпы в хвостохранилище

На рисунке 3.78 показан процесс забора подземных вод из всех водоносных слоев, затронутых при разработке ГОКов. Прямоугольные

индикаторы на блоках ГОКов показывают количество забираемой воды на нужды города, остаточное количество воды используется ГОКами в процессе функционирования. Количество забираемой воды в сутки отражено через связи «*забор9*» и «*забор10*» и параметр «*Q*», при этом общий забор воды на нужды Старооскольского и Губкинского городского округа вычисляется в узловом объекте «*забор воды*». Прямоугольный индикатор блока «*забор воды*» отражает долю от максимально возможного забора воды в сутки. Также на рисунке 13 отражено взаимодействие ГОКов со смежными хвостохранилищами. Через связь «*Q*» и параметр «*колво*» имитируется подача пульпы в хвостохранилище.

На рисунке 3.79 отражены графики количества забираемой воды на городские нужды в сутки. Верхний график показывает количество забираемой воды из СГОКа, нижний – из ЛГОКа.

Из приведенных выше графиков следует, что количество забираемой воды через дренажную систему СГОКа составляет примерно 90000 м^3/сут, а через систему ЛГОКа – примерно 140000 м^3/сут. На рисунке 3.80 демонстрируется совокупный забор воды в сутки на территории Старооскольского и Губкинского городского округа.

Из приведенного выше графика следует, что количество забираемой воды на нужды Старооскольского и Губкинского городского округа приходится примерно 360000 м^3/сут.

В качестве второго эксперимента увеличим потребление воды из ГОКов на городские нужды. Для этого изменим значение показателя «*dolya_goroda*» с 0,85 (85% всего водозабора) до 0,99 (практически полный забор воды на городские нужды). Рассмотрим обновленные графики показателей количества воды, забираемой из ГОКов, а также их влияние на общий забор воды в сутки. На рисунке 3.81 представлены обновленные графики водозабора объектов «*СГОК*», «*ЛГОК*» в соответствии с измененным показателем «*dolya_goroda*».

Приведенные графики свидетельствуют о том, что количество забираемой воды через дренажную систему СГОКа увеличилось на 15000 м^3/сут и составило примерно 105000 м^3/сут, а через систему ЛГОКа – увеличилось на 25000 м^3/сут и составило примерно 165000 м^3/сут. На рисунке 3.82. представлен обновленный график водозабора объекта «*забор воды*» в соответствии с измененным показателем «*dolya_goroda*».

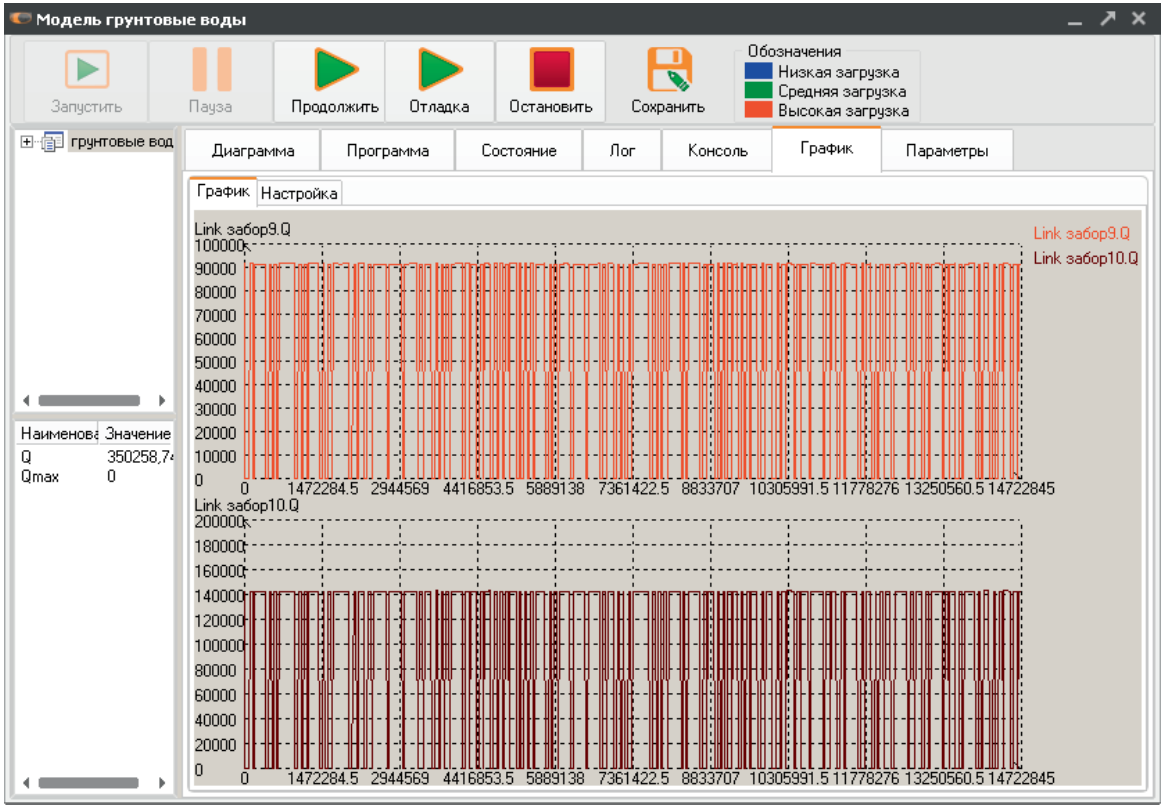


Рис. 3.79. Объем забираемой воды из ГОКов на нужды городских округов

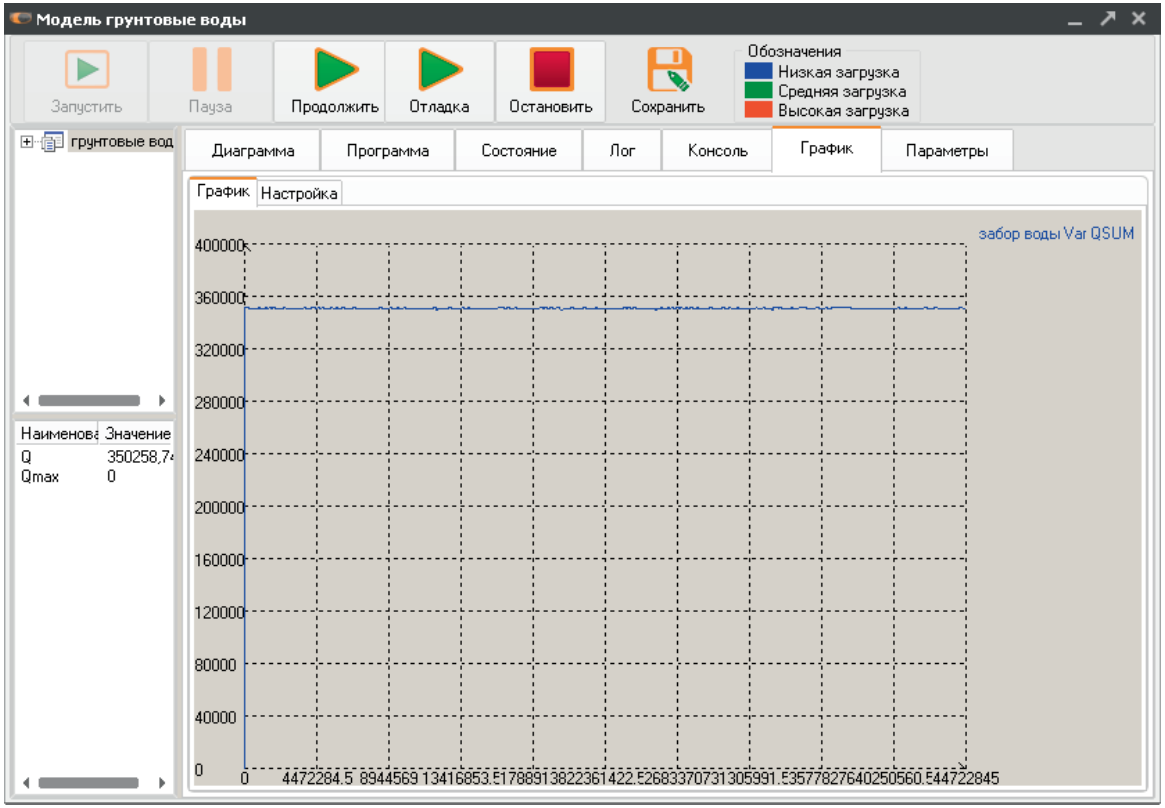


Рис. 3.80. Общее количество забираемой воды на нужды городских округов

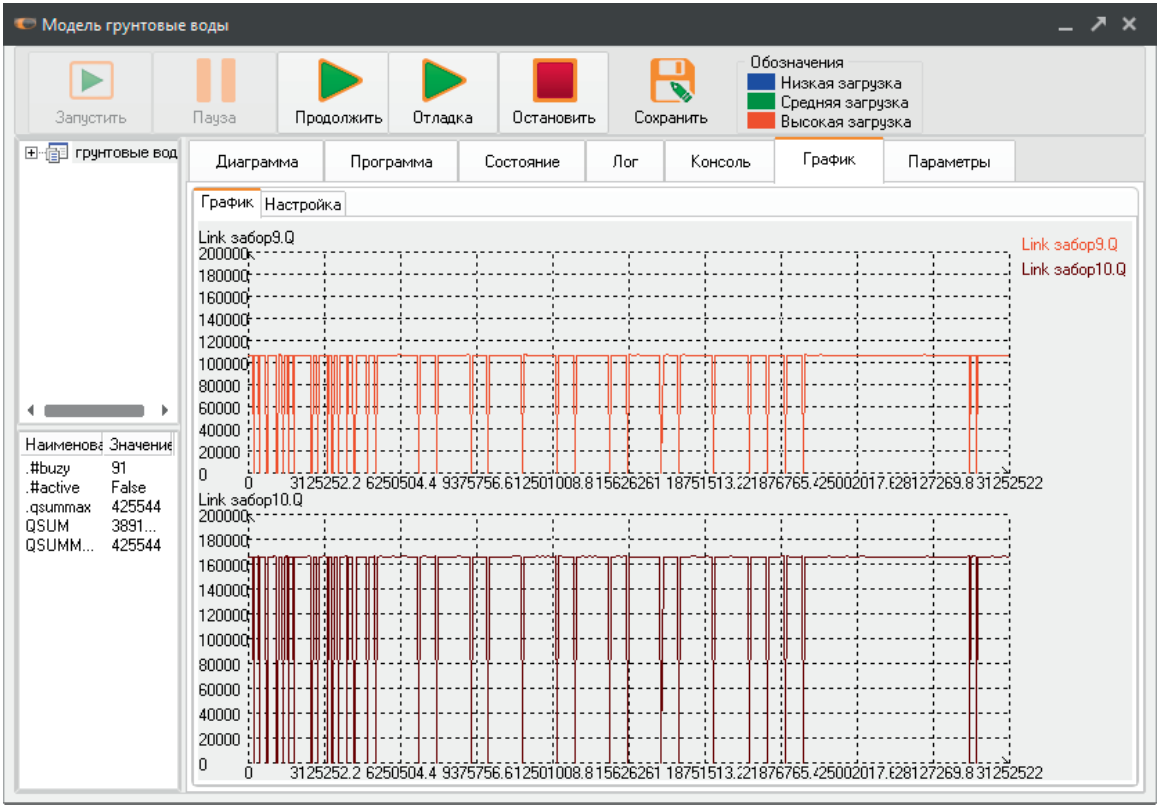


Рис. 3.81. Увеличение объемов забираемой воды из ГОКов на нужды городских округов

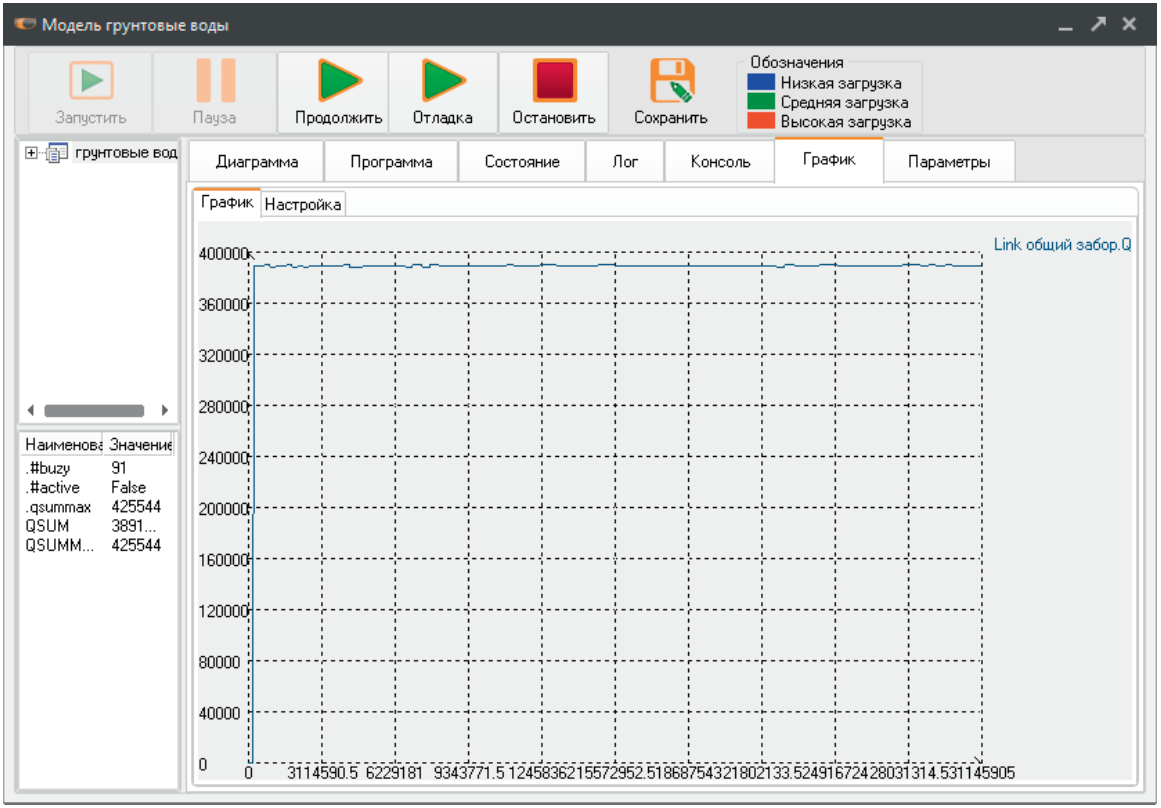


Рис. 3.82. Увеличение объема забираемой воды на нужды городских округов

Согласно графику, приведенному выше, совокупный водозабор увеличился с 350000 м^3/сут до 390000 м^3/сут. Что соответствует увеличенному объему заборов воды из дренажных систем СГОКа и ЛГОКа на нужды Старооскольского и Губкинского городского округа.

Таким образом, увеличение забора воды на городские нужды через дренажные системы ГОКов привели к увеличению совокупного забора воды.

Рассмотрим имитацию процесса функционирования хвостохранилищ ГОКов, а также оценим их техногенное влияние на окружающую среду. Для этого запустим модель с масштабом времени: 1 секунда реального времени равна 1 суткам модельного времени (рис. 3.83).

На рисунке 3.83 показан процесс отделения хвостов от воды, благодаря которому очищенная вода поступает обратно в производственный процесс ГОКов. Красные квадраты-индикаторы на блоках хвостохранилищ указывают на их функционирование. Количество возвращаемой воды в ГОКи в сутки отражено через связь «*Q3*» и параметр «*колво*». Стоит отметить, что количество возвращаемой из хвостохранилища воды в модели определяется параметром «*dolya_vody*» Также на рисунке 3.83 отражено влияние ГОКов на окружающую среду. Через исходящие связи «*Q1*» и «*Q2*» и параметры «*колво*» и «скорость» имитируется проникновение загрязненной воды в почву.

На рисунке 3.84 отражены графики количества отфильтрованной в хвостохранилищах воды, возвращенной в ГОКи в сутки. Верхний график показывает количество возвращенной воды в СГОК, нижний – в ЛГОК.

Из приведенных выше графиков следует, что количество возвращаемой воды через хвостохранилище СГОКа составляет примерно 12000 м^3/сут, а через хвостохранилище ЛГОКа – примерно 23000 м^3/сут.

Из приведенных выше графиков следует, что количество проникающей в почву воды через хвостохранилище СГОКа составляет примерно 670 м^3/сут, а через систему ЛГОКа – примерно 1950 м^3/сут.

Из приведенных выше графиков следует, что скорость проникающей в почву загрязненной воды через хвостохранилище СГОКа снизилась на 0,00002 м/сут примерно до 0,00006 м/сут, а через хвостохранилище ЛГОКа – примерно на 0,00003 м/сут до 0,00012 м/сут.

Таким образом, изменение толщины защитной стенки хвостохранилища, а также использование эффективных материалов для изоляции хвостов, напрямую влияет на объем и скорость протекающей сквозь защитную стенку загрязненной воды.

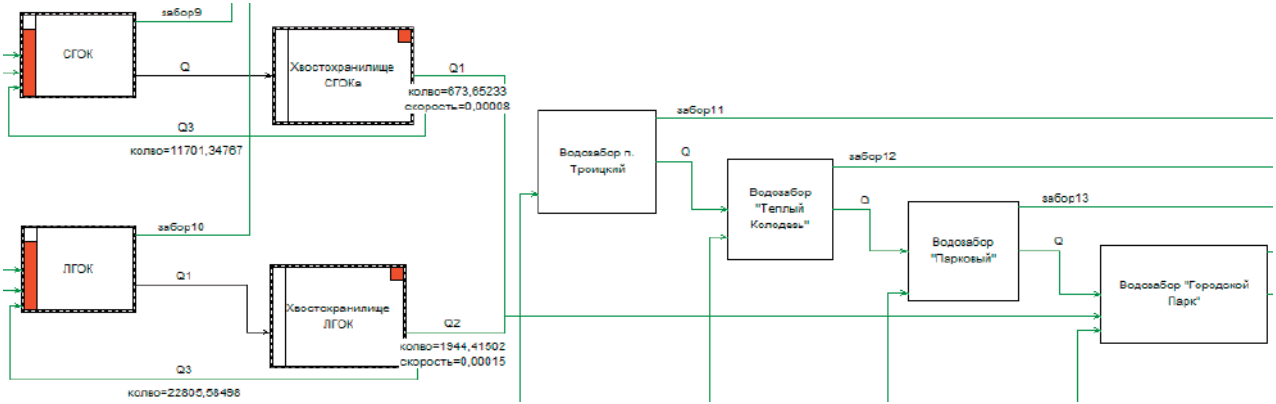


Рис. 3.83. Очищение и транспортировка воды из хвостохранилищ в ГОКи

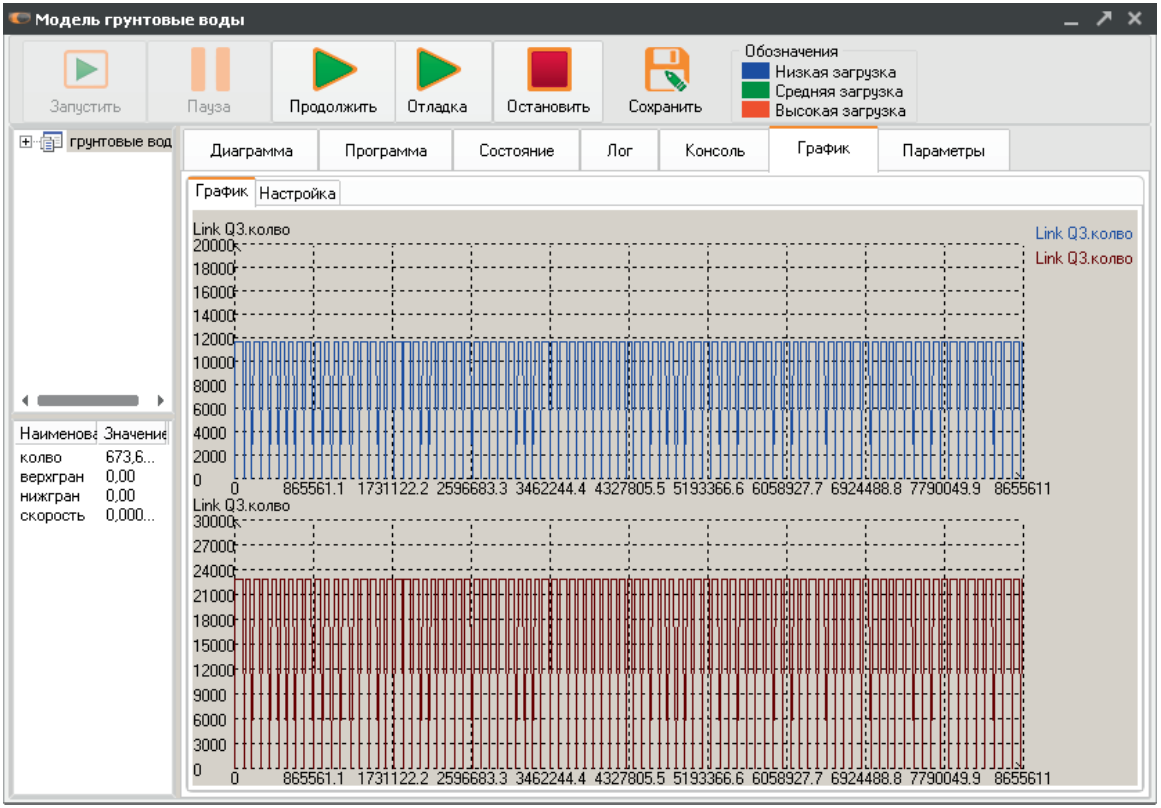


Рис. 3.84. Количество отфильтрованной и транспортированной в ГОКи воды

Разработанная имитационная модель отражает сезонные колебания уровня подземных вод. Наличие встроенного языка описания функциональных узлов делает модель очень гибкой в использовании. При проведении экспериментов имеется возможность имитировать различные техногенные условия путем изменения соответствующих методов, а также значений показателей узловых объектов. Далее рассмотрим подробнее программный инструментарий имитационного моделирования процессов и систем UFOModeler.

4.2.2. Программный инструментарий имитационного системно-объектного моделирования

Рассмотрим программный пакет системно-объектного имитационного моделирования («UFOModeler»; URL: https://ufomodeler.ru) на примере моделирования систем массового обслуживания (СМО).

Построим модель одноканальной СМО с отказами, для этого в UFOModeler необходимо создать узлы «Клиенты», «Обработка». Для этого нужно открыть программу и, выбрав «Создание модели», откроется окно, в котором заполняем поля названия модели, ее описание, размеры, стили. На рисунке 3.85 изображен результат заполнения полей для создания модели.

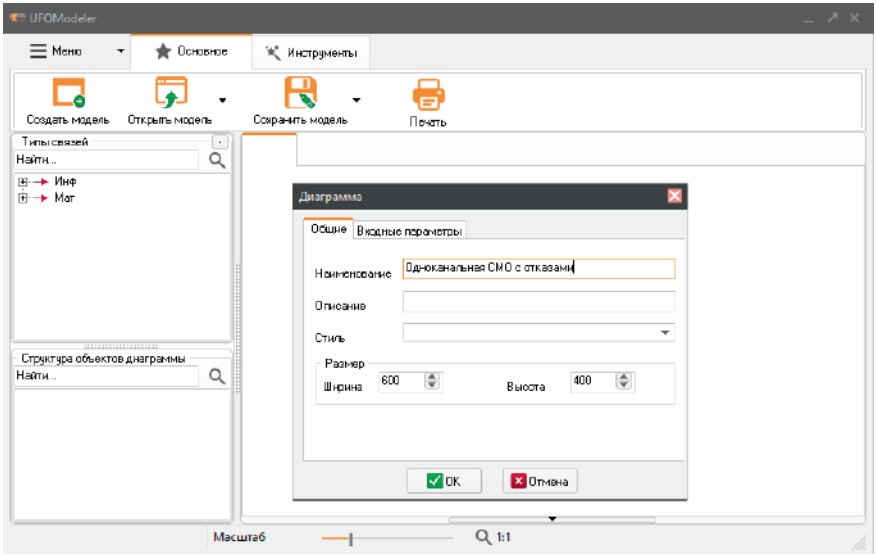


Рис. 3.85. Фрагмент создания одноканальной СМО с отказами

После заполнения полей подтверждаем нажатием на кнопку «ОК». Затем переходим на закладку «Инструменты» и выбреем инструмент «Узел», для создания на рабочей области узла. После создания будущего узла происходит открытие дополнительного окна, в котором указываются характеристики узла (название узла, его функция, объект, свойства объекта). На рисунке 3.86 изображено окно свойств узла.

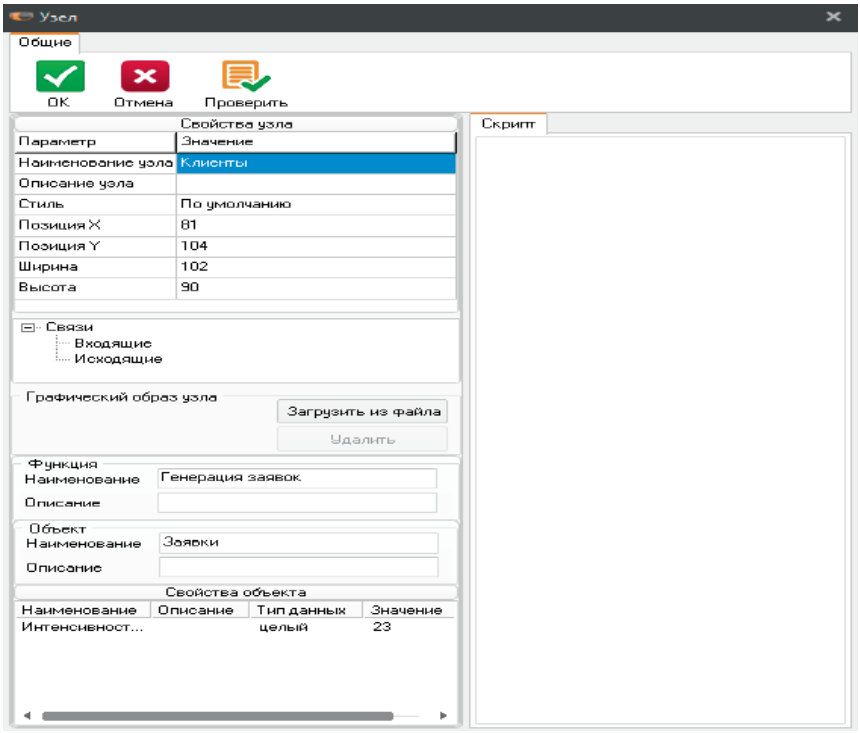


Рис. 3.86. Характеристики узла «Клиенты» в одноканальной СМО с отказами

Заполнив все параметры, необходимо подтвердить это нажатием на кнопку «ОК». Аналогичным способом создается узел «Обработка». Характеристики узла представлены на рисунке 3.87.

После подтверждения введенных параметров, необходимо определиться какие требуются связи для взаимосвязи узлов. Так как предполагается что узел «Клиенты» будет производить генерацию потока заявок за определенный период времени, поэтому необходимо создать связи позволяющие: передавать поток заявок; отслеживание заявок, получивших отказ; взаимопонимание того что по

прошествии некоторого количества времени, наступил новый период времени. Назначение узла «Обработка» заключается в получении значений из другого узла, по окончанию передачи заявок за определенный промежуток времени выполнять и предоставлять вычисления характеристик СМО.

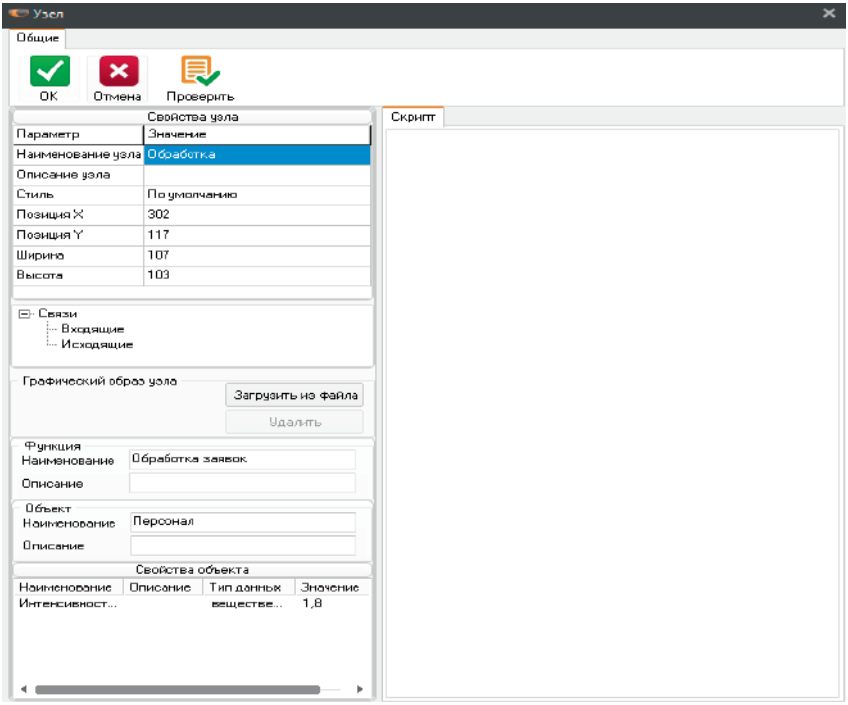


Рис. 3.87. Характеристики узла «Обработка» в одноканальной СМО с отказами

Для создания связей, требуется в левой части программы в дереве связей, раскрыть дерево связей, например, «Инф» и щелкнув правой кнопкой мыши по связи «По данным», в появившемся списке выбрать «Добавить вложенный элемент». В открывшемся окне «Тип связей» во вкладке «Общее» введем название связи и, если нужно описание. Во вкладке «Параметры» щелкнув правой кнопкой мыши в появившемся списке выбрать «Создать». В появившемся окне «Параметр типа связи» заполняем поля. Окна создания связей и параметров отображены на рисунке 3.88.

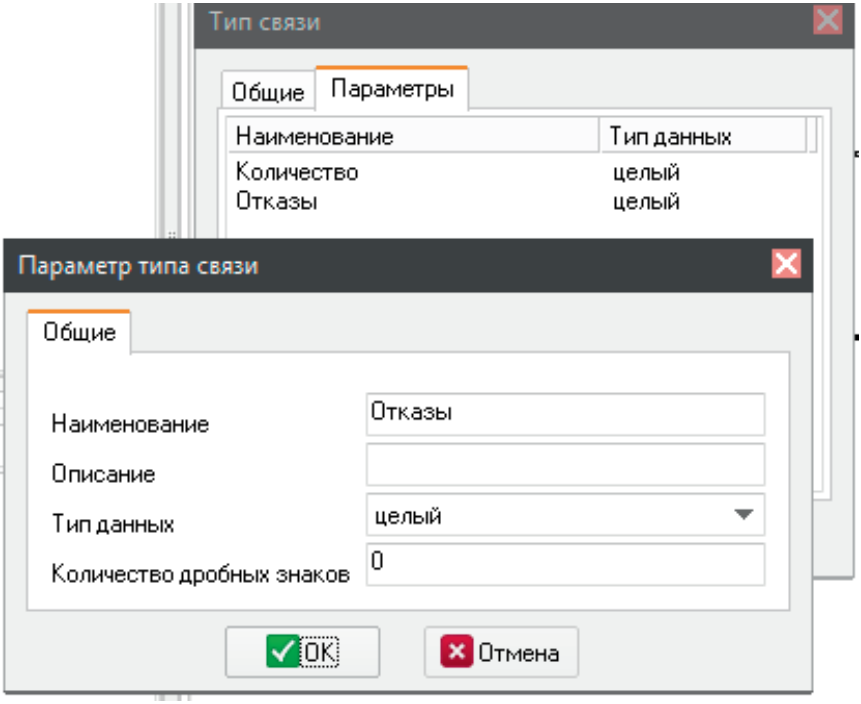


Рис. 3.88. Параметры связи «Заявки» в одноканальной СМО с отказами

По аналогичному способу, были созданы остальные связи, полученное дерево связей представлена на рисунке 3.89.

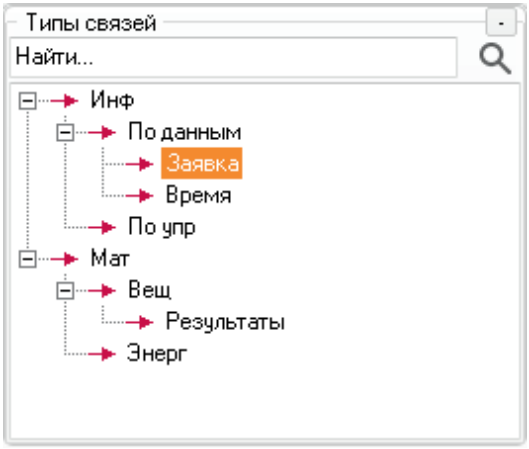


Рис. 3.89. Дерево связей одноканальной СМО с отказами

Перейдем к процессу соединения узлов. В дереве связей выберем связь «Время», во вкладке «Инструменты» выбрав инструмент «Связь». Щелкнем левой кнопкой мыши для начала по узлу «Клиенты», а затем по «Обработка», в результате получаем связь между узлами. Аналогичным образом создается связь «Заявка». Результат создания связей «Время», «Заявка» отображен на рисунке 3.90.

Что касается связи «Результаты», то данная связь не имеет конечной связи, в данном случае она выступает в роли информатора полученных результатов, но при изменении модели, может быть использована как связь с другим узлом. Для создания исходящей связи из узла «Обработка», требуется выбрать в дереве связей «Результаты» перейти во вкладку «Инструменты» и выбрать инструмент «Исходящая связь», далее щелкнуть левой кнопкой мыши по «Обработка». В результате получилась модель, представленная на рисунке 3.91.

Даже после того как были созданы узлы и выстроены связи, модель не будет выполнять ни каких действий из-за того, что узел не имеет набор функций.

Разберем алгоритм работы узла «Клиенты» одноканальной СМО представленный на рисунке 3.92. Перед выполнением работы считываются значения периода времени и интенсивности времени, заданные в свойствах узла. Затем находится время, через которое нужно отправлять следующую заявку (интервал между заявками), выполнив деление интенсивности потока заявок на период времени.

Пока существует интенсивность потока заявок, передаются заявки с вычисленной интенсивностью и после каждой передачи заявки выполняется проверка на наличие оставшихся не отправленных заявок и наличие оставшегося времени.

Если не осталось заявок и времени, производится передача сигнала по связи «Время» о начале нового периода времени и происходит завершение работы алгоритма, иначе выполняется другая проверка на наличие не отправленных заявок и о отсутствии оставшегося времени.

Если есть не отправленные заявки, а времени на их передачу в обработку не осталось, то заявки, не успевшие пройти обслуживание отправляются как заявки, получившие отказ и алгоритм, завершает работу, иначе выполняется другая проверка.

Если есть заявки и время происходит снова отправка заявки с задержкой между заявками, иначе происходит ожидание завершения периода времени и передача сигнала о изменении времени.

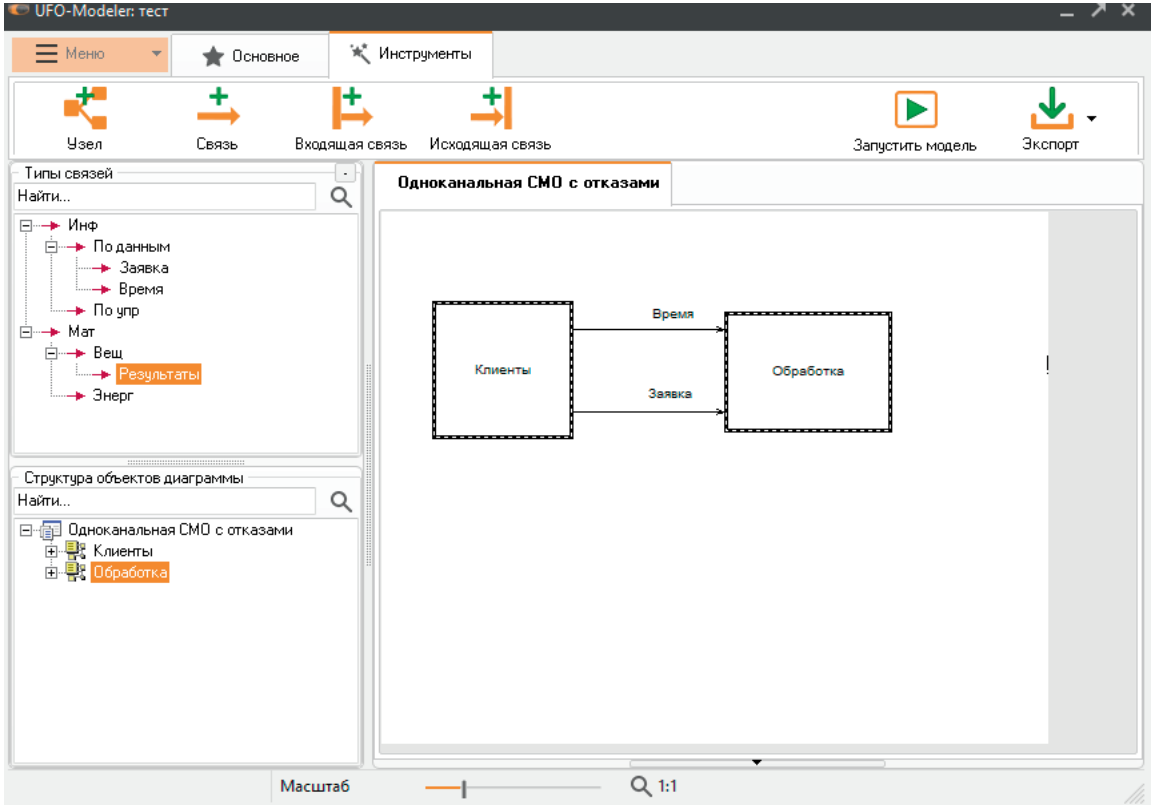


Рис. 3.90. Результат создания связей «Время», «Заявка» в одноканальной СМО с отказами

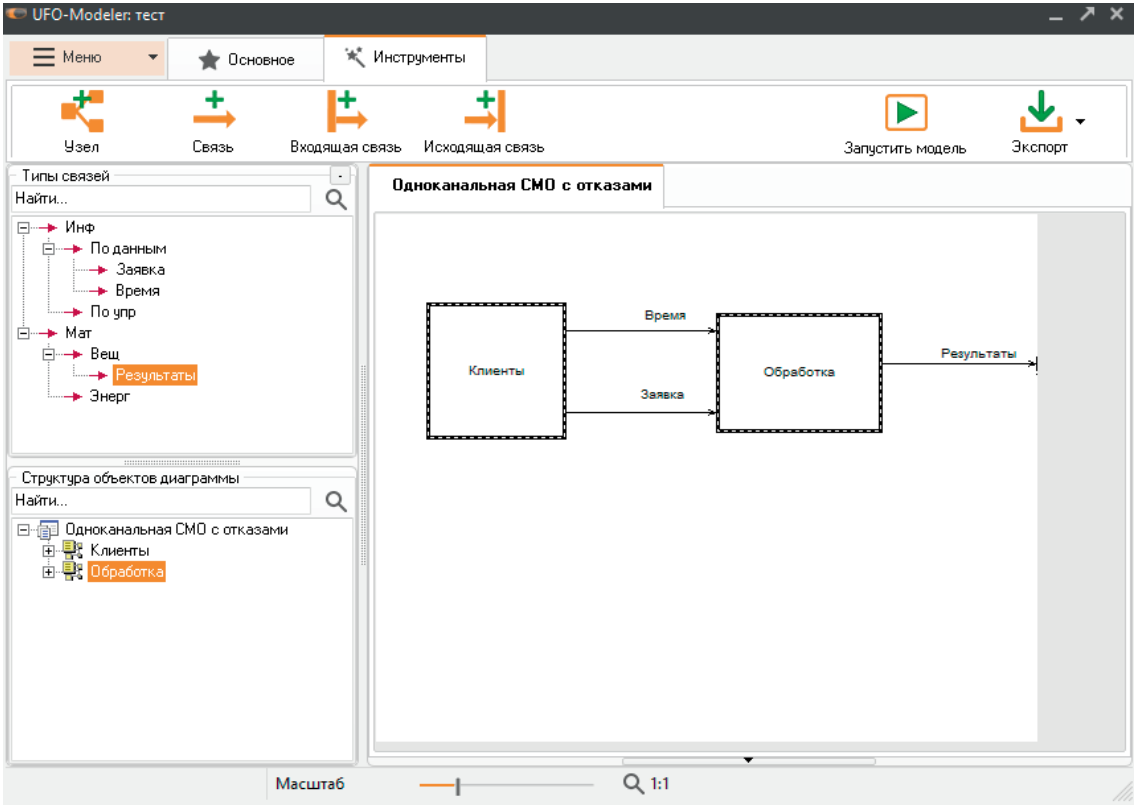


Рис. 3.91. Имитационная модель одноканальной СМО с отказами

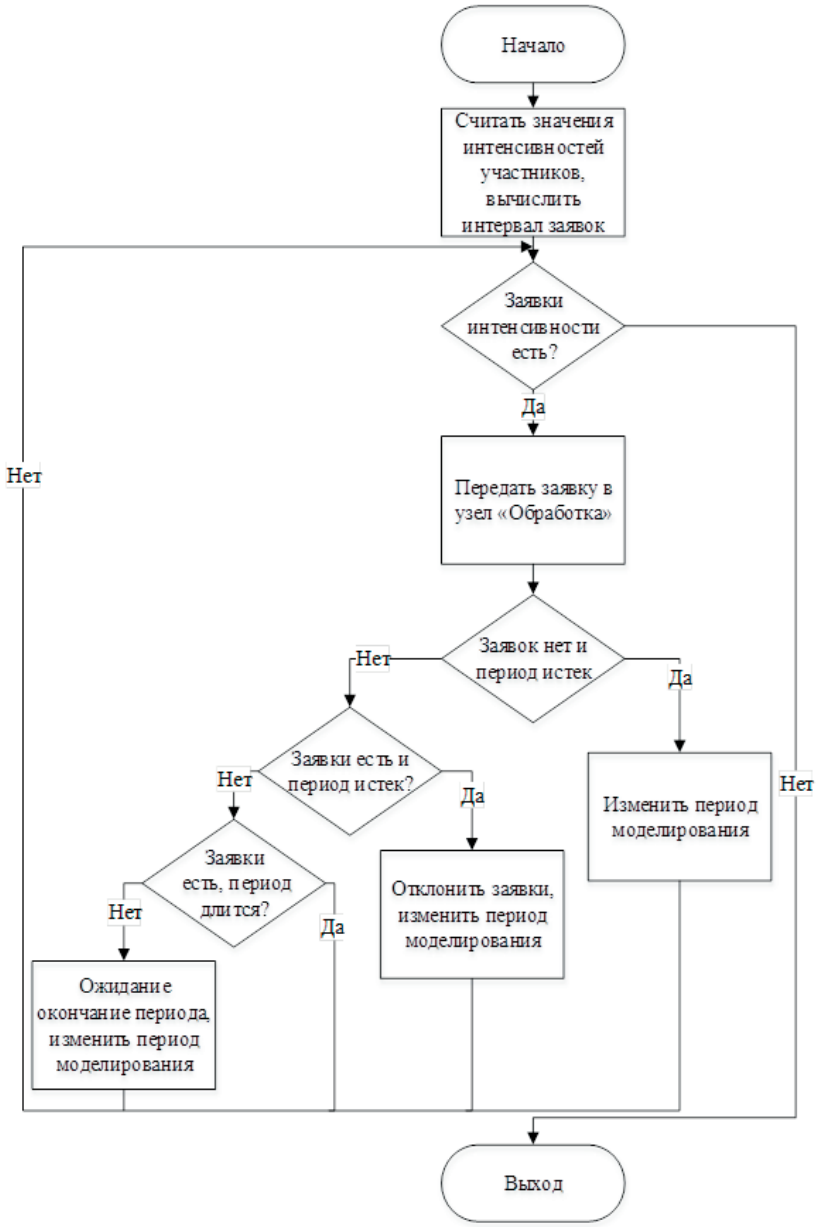


Рис. 3.92. Алгоритм функционирования узла «Клиенты» одноканальной СМО с отказами

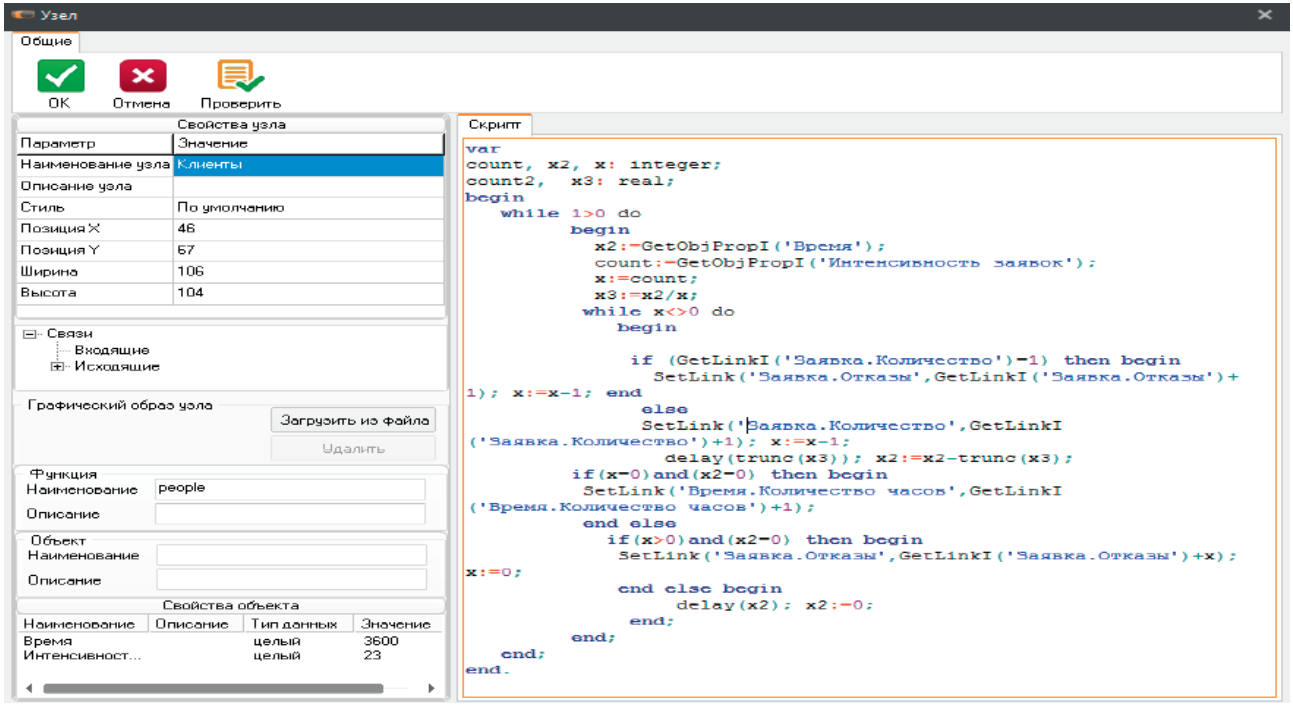


Рис. 3.93. Фрагмент кода узла «Клиент» одноканальной СМО с отказами

.

После выполнения всех перечисленных действий происходит возврат к проверке на наличие необслуженных заявок, если их не оказалось, то алгоритм завершает работу.

На рисунке 3.93 представлен код, который был реализован по разработанному алгоритму. Для его реализации необходимо написать в свойства узла «Клиенты» в области «Скрипт»

На рисунке 3.94 изображен алгоритм работы узла «Обработка» одноканальной СМО заключается в считывании из свойств узла времени обработки одной заявки. Далее выполняется прием поступающих заявок, а именно пока заявки поступают выполняется обслуживание заявки с определённой задержкой. Значение задержки считывается из свойств узла «Обработка». После того как прошло время обработки выполняется проверка на наличие текущего периода из связи «Время». Если период был изменен, то выполняется вычисление среднего значения каждой характеристики данного типа СМО. Если результаты вычислены в первый период, то они запоминаются и записываются в связь «Результаты» без деления, когда вычисления выполняются не в первый раз тогда обработанные заявки и заявки с отказами складываются и используют это значения, каждый результат делится на количество пройденных периодов. Затем полученные средние значения передаются на связь «Результаты».

По разработанному алгоритму был реализован код в свойства узла «Обработка», в области «Скрипт». Реализованный код представлен на рисунке 3.95.

Перед моделированием зададим интенсивность потока – 23 заявки в час перейдя в свойства узла «Клиенты» и в свойстве объекта щелкнем по созданному ранее пункту «Интенсивность потока» установим значение 23 и в пункте «Время» значение 3600 подтвердив изменения кнопкой «ОК».

Для задания интенсивность обслуживания перейдем в свойства узла «Обслуживание» и в свойстве объекта щелкнем по пункту «Интенсивность обслуживания» установим значение 1,8 минут подтвердив изменения кнопкой «ОК».

После всех выполненных операций, можно приступить к выполнению моделировании одноканальной СМО с отказами. Для этого необходимо либо во вкладке «Инструменты» выбрать «Запустить модель», либо щелкнув левой кнопкой мыши по пункту «Меню» и в выпадающем списке выбрать «Запустить модель». После нажатия появляется окно, в котором на панели инструментов выбираем «Запустить модель». После нажатия происходит выполнение имитационного

моделирования. В данном окне можно посмотреть текущие свойства, которые имеет каждый узел, для этого щелкнув по узлу «Клиенты», в левой части программы появляется список всех переменных и их результаты. Окно после запуска модели отображена на рисунке 3.96.

Для просмотра значений, находящихся в связях, можно щелкнув по связи левой кнопкой мыши, и увидеть в левой части программы параметры связи и их значения.

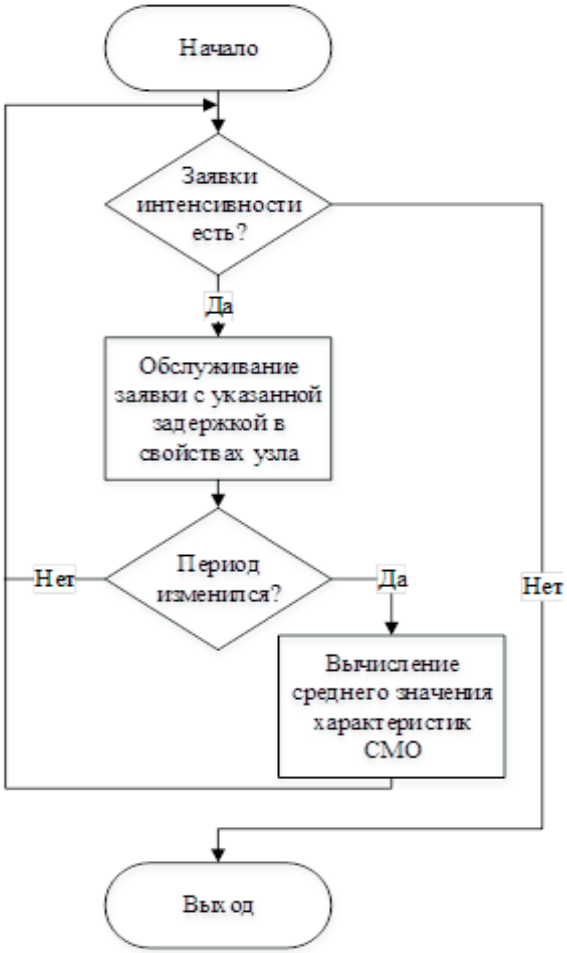


Рис. 3.94. Алгоритм узла «Обработка» одноканальной СМО с отказами

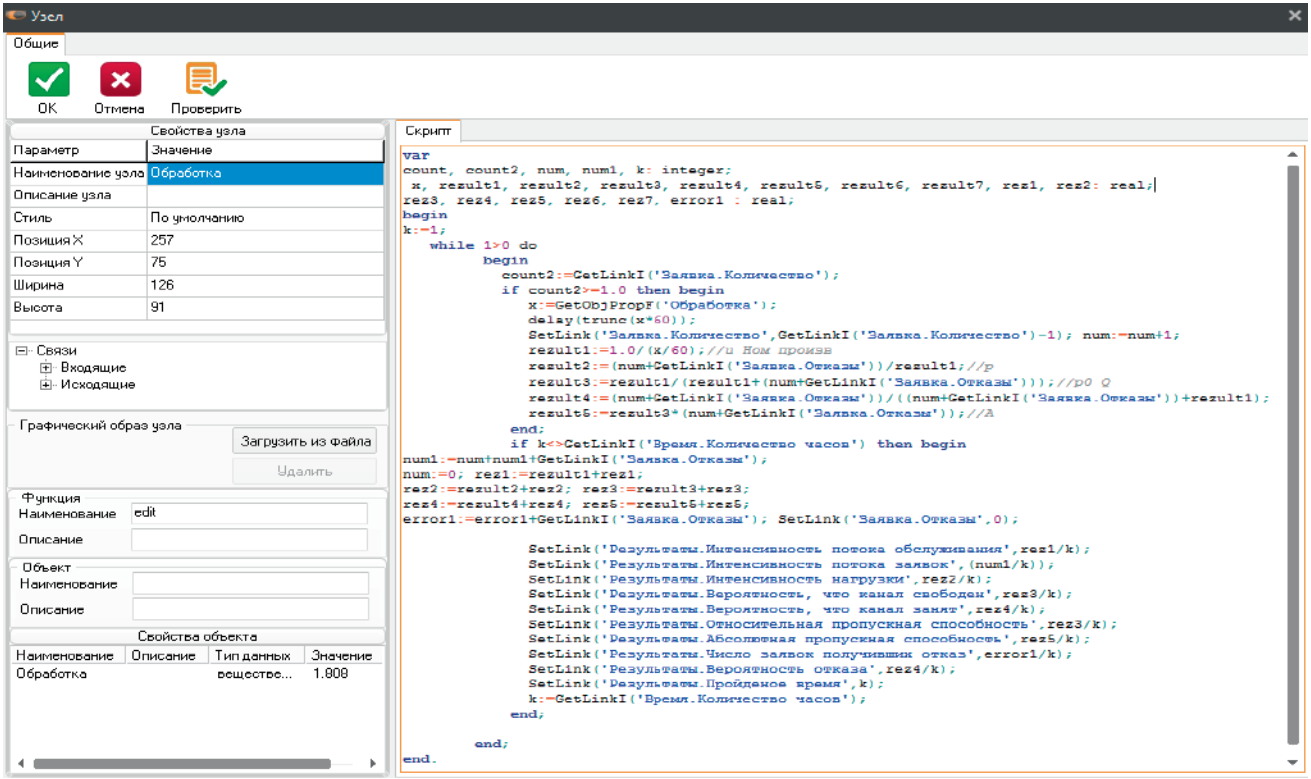


Рис. 3.95. Фрагмент кода узла «Обработка» одноканальной СМО с отказами

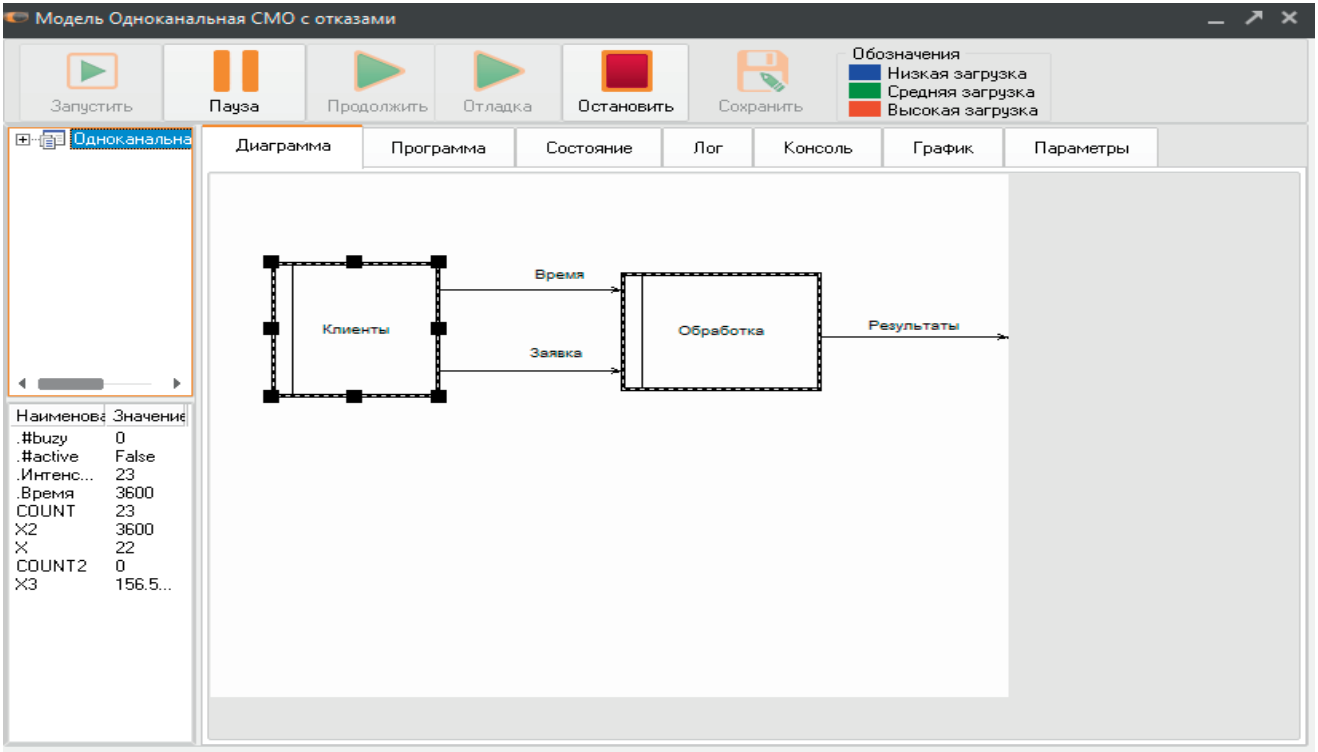


Рис. 3.96. Запуск модели «Одноканальная СМО с отказами»

Также можно посмотреть историю изменений параметров в каждом узле, значения параметров в связях, переменные используемые в каждом коде. Для этого требуется перейти во вкладку «Состояние» и выбрать галочкой параметры «Заявка.Количество», «Заявка.Отказы» затем перейдя во вкладку «Графики» произвести настройку. Выбрать способ отображения графика «Для каждого набора данных свой график» и поставить галочку в поле «Автоматическое обновление графика». На рисунке 3.97 отображено окно настройки графиков.

На рисунке 3.98 запечатлён процесс отображения результатов на графике.

Для того что бы перейти для просмотра графика нужно перейти во вкладку «График», где можно наблюдать различные виды отображения графиков в зависимости от выбранных настроек.

Для того, чтобы управлять моделируемым временем, нужно перейти во вкладку «Параметры» и выбрав в выпадающем списке масштаб времени «5к1» Это означает 5 секунд модельного времени за 1 секунду реального времени. На рисунке 3.99 можно увидеть список масштабов времени моделирования.

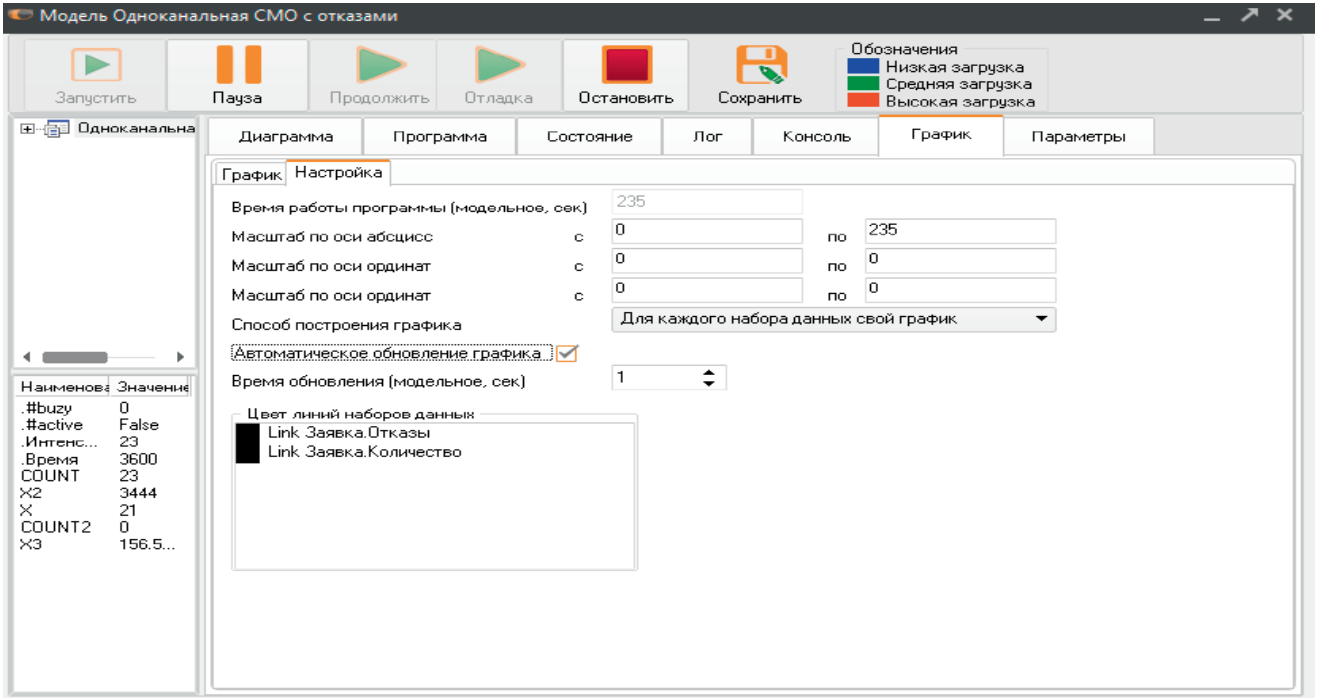


Рис. 3.97. Настройка графика в модели «Одноканальная СМО с отказами»

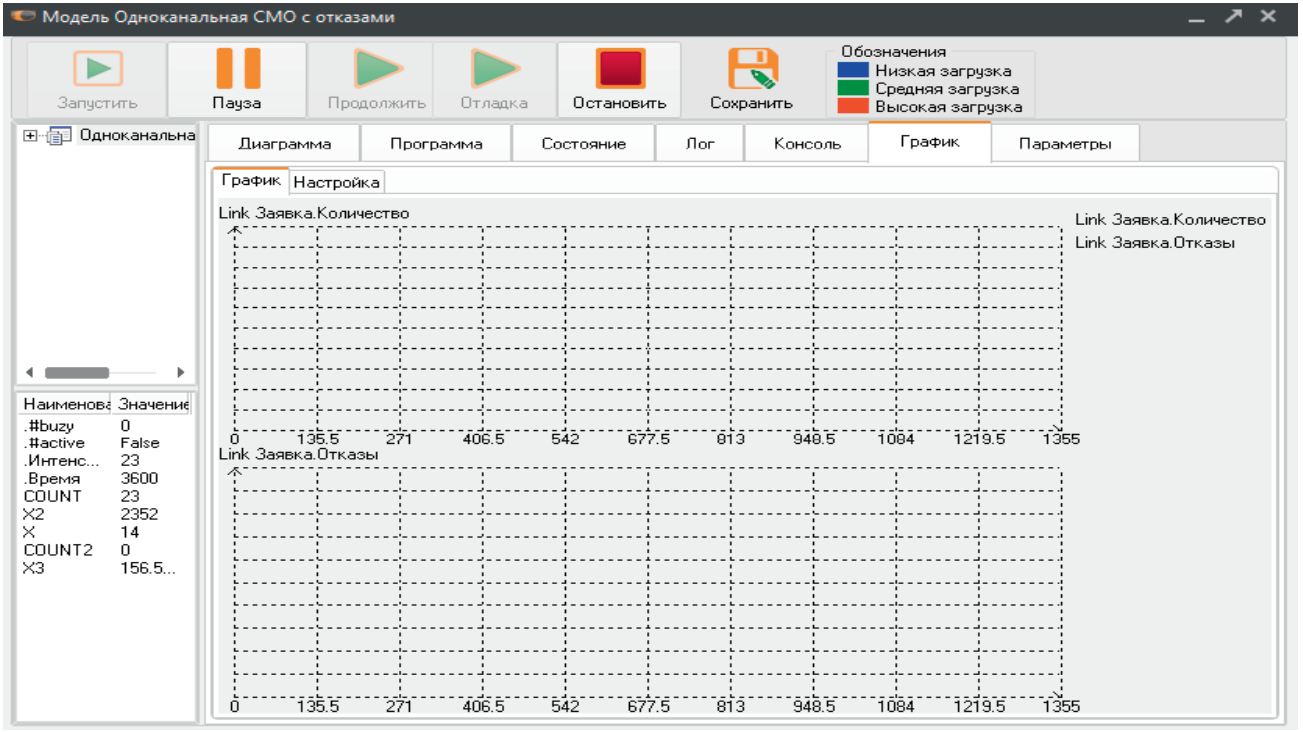


Рис. 3.98. Графики параметров «Заявка.Количество», «Заявка.Отказы» в модели «Одноканальная СМО с отказами»

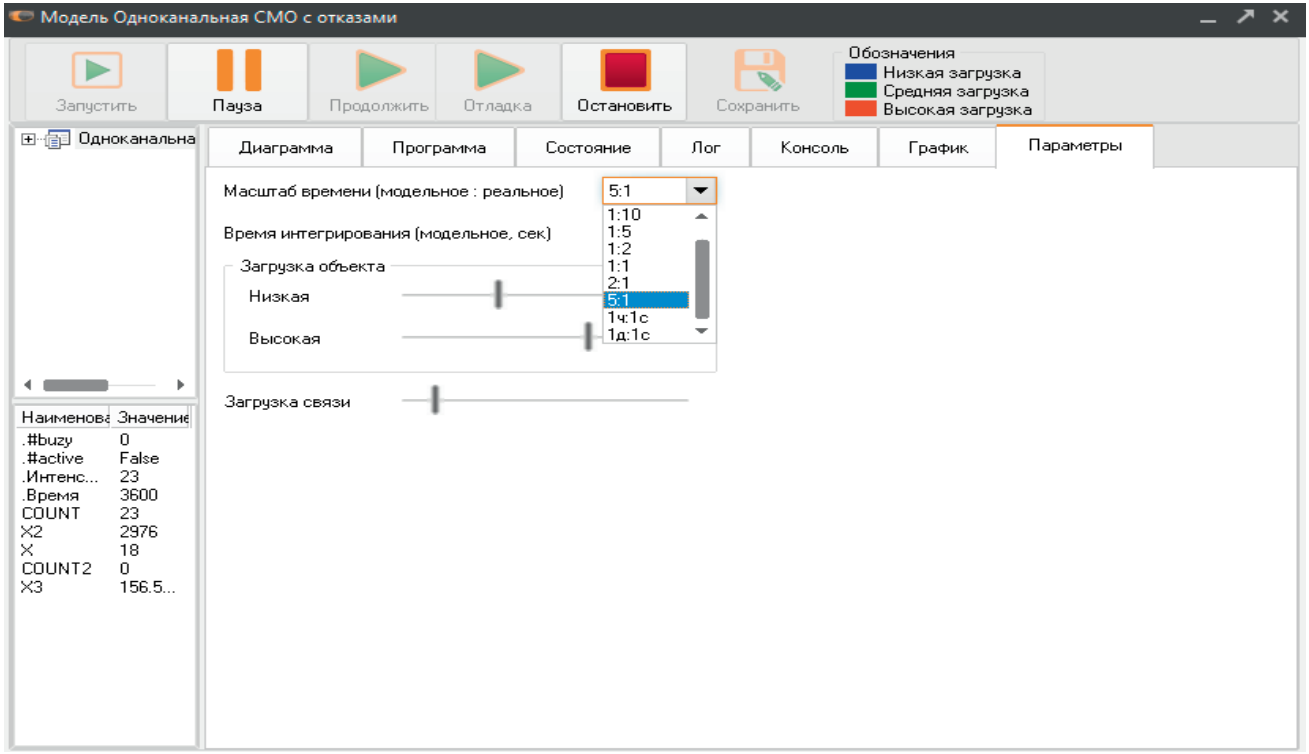
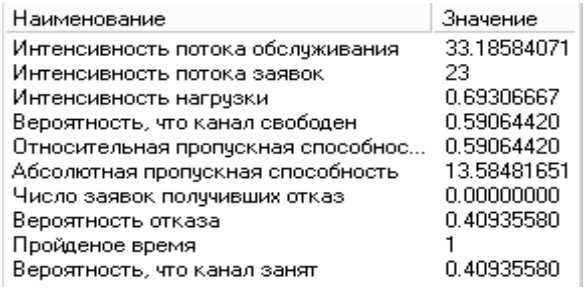


Рис. 3.99. Настройка параметров в модели «Одноканальная СМО с отказами»

После моделирования модели «Одноканальная СМО с отказами», были получены результаты в связи «Результаты», представленные на рисунке 3.100.



Наименование	Значение
Интенсивность потока обслуживания	33.18584071
Интенсивность потока заявок	23
Интенсивность нагрузки	0.69306667
Вероятность, что канал свободен	0.59064420
Относительная пропускная способнос...	0.59064420
Абсолютная пропускная способность	13.58481651
Число заявок получивших отказ	0.00000000
Вероятность отказа	0.40935580
Пройденое время	1
Вероятность, что канал занят	0.40935580

Рис. 3.100. Результаты одноканальной СМО с отказами

После имитационного моделирования полученные результаты, необходимо сравнить с результатом, полученным с помощью формул. Результаты вычислений с помощью формул приведены в таблице 3.20.

Таблица 3.20

Вычисления по формулам для одноканальной СМО с отказами

Название	Результат
Интенсивность потока заявок λ=	23
Интенсивность потока обслуживания μ=	33.18584071
Интенсивность нагрузки p=	0.693066667
Вероятность, что канал свободен p0=	0.590644196
Вероятность, что канал занят p1=	0.409355804
Вероятность отказа Ротк=	0.409355804
Относительная пропускная способность Q=	0.590644196
Абсолютная пропускная способность A=	13.58481651

Как видно из полученных результатов, результаты, получившиеся с помощью формул и программы, являются одинаковыми. Вероятность того что канал занят обслуживанием составляет 0,40=40%, что соответствует графику на рисунке 3.101, полученному в процессе моделирования одноканальной СМО.

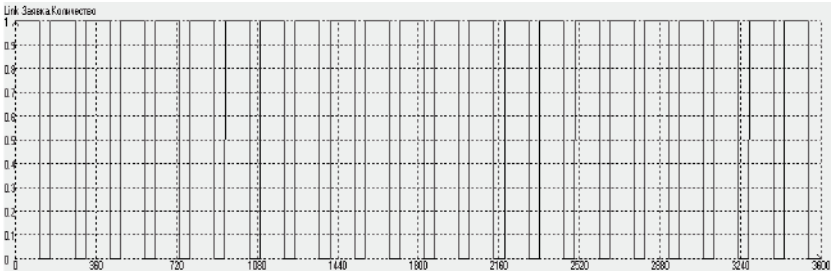


Рис. 3.101. График моделирования одноканальной СМО с отказами

Мы можем наблюдать, что UFOModeler обеспечивает получение точных результатов и корректное моделирование одноканальной СМО. Теперь необходимо проверить как себя ведет система при моделировании многоканальной СМО.

Для этого была создана модель с 3 каналами для обслуживания входящего потока. Данная модель имеет небольшие изменения от предыдущие, отличие заключается только в количестве каналов обслуживания и наличие узла, собирающего с 3 каналов обработанные заявки. Также подвергся небольшим изменениям и узел «Клиенты. Результат разработанной модели представлен на рисунке 3.102.

На рисунке 3.103 представлен алгоритм работы узла «Клиенты» многоканальной СМО заключается в считывании значений периода времени, интенсивности заявок, длине допустимой очереди, установленных в свойствах узла. Затем находится интервал между заявками делением общего количества заявок на период времени. До тех пор, пока существует интенсивность потока заявок, выполняется проверка всех каналов на наличие свободного места на какой-либо связи.

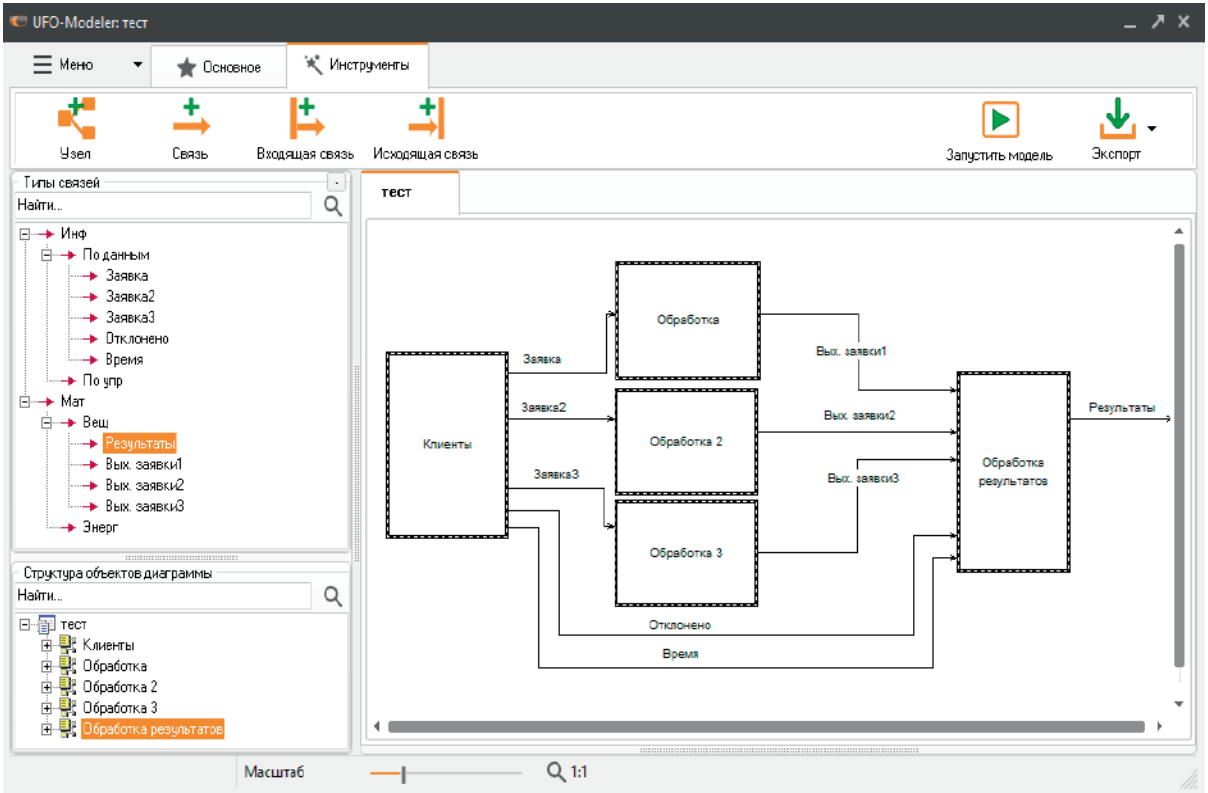


Рис. 3.102. Имитационная модель многоканальной СМО с ограниченной очередью

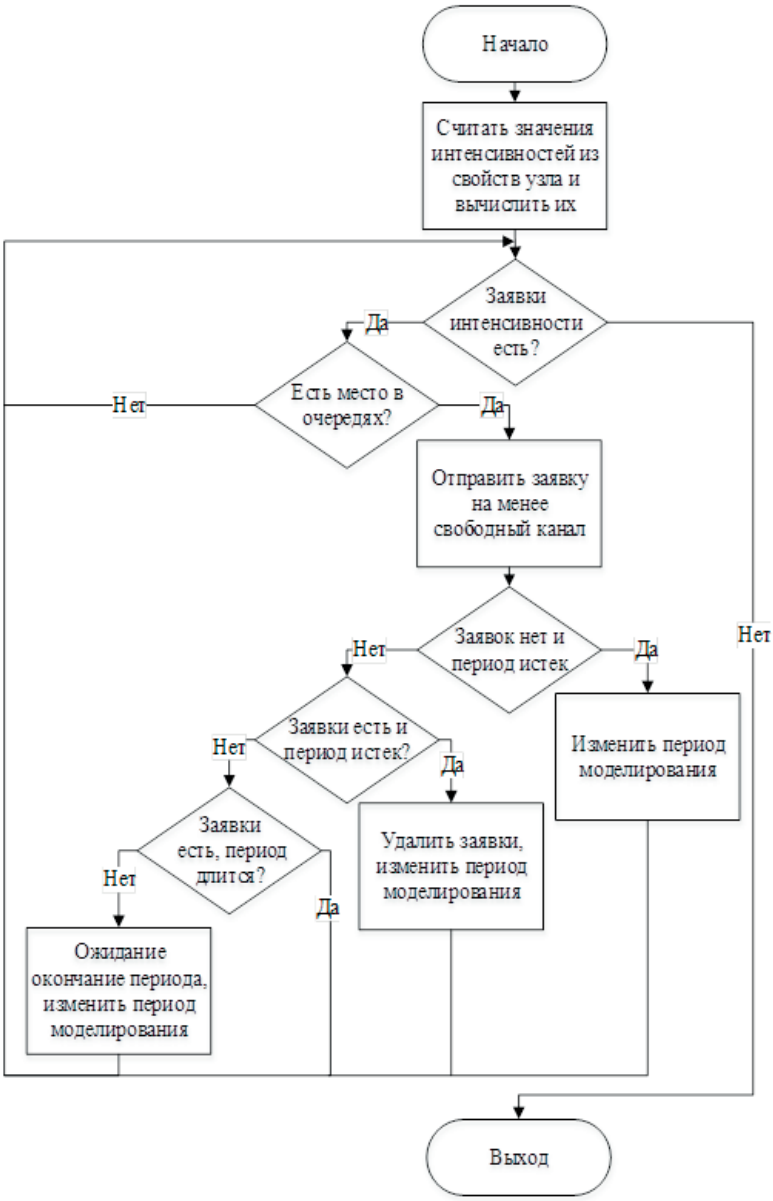


Рис. 3.103. Алгоритм узла «Заявки» многоканальной СМО с ограниченной длинной очереди

Если места есть, выполняется передача заявки в наиболее свободный канал и после передачи выполняется проверка. Если заявок больше нет и время периода закончилось, производится передача сигнала по связи «Время» в узел «Обработка результатов» о начале нового периода времени и алгоритм завершает работу, иначе выполняется другое условие, где проверяется количество заявок и оставшегося времени. Если заявки не все переданы, а времени уже не осталось, тогда заявки в подсчете результатов не рассматриваются и удаляются и алгоритм завершает работу, иначе выполняется другая проверка. Если заявки и время на их передачу есть, происходит возврат к месту где проверяется наличие необслуженных заявок и процесс передачи заявок повторяется, иначе ожидается окончания времени периода и передается сигнал о изменении периода.

На рисунке 3.104 представлен код, который был реализован код, по разработанному алгоритму. Для его реализации необходимо написать в свойства узла «Клиенты» в области «Скрипт».

На рисунке 3.105 изображен алгоритм работы узлов «Обработка», «Обработка2», «Обработка3» многоканальной СМО, он заключается в считывании из свойств объекта времени обслуживания одной заявки. Далее пока есть необслуженные заявки, выполнять обслуживание поступающих заявок с заданным временем. Когда необслуженные заявки заканчиваются алгоритм завершает работу.

Рис. 3.105. Алгоритм узлов «Обработка», «Обработка2», «Обработка3» многоканальной СМО с ограниченной длинной очереди

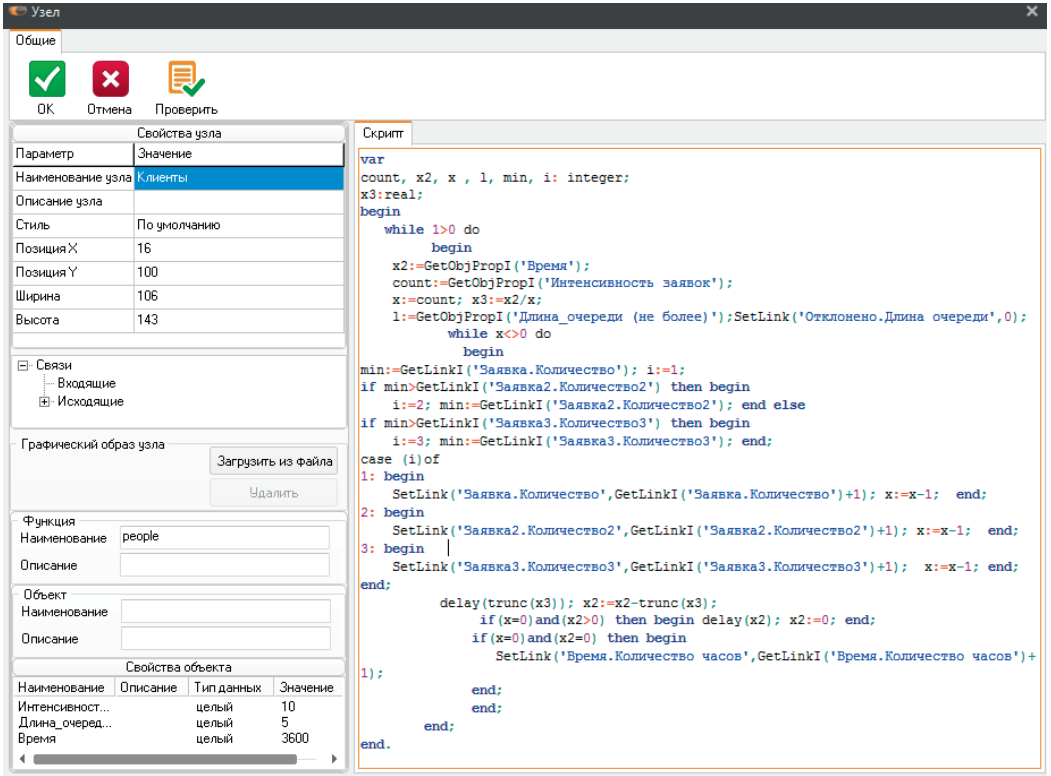


Рис. 3.104. Фрагмент кода узла «Клиенты» многоканальной СМО с ограниченной длинной очереди

Используя разработанный алгоритм, его можно применить для реализации кода узла «Обработка». Для этого код необходимо вставить в область «Скрипт» в свойствах узла «Обработка». На рисунке 3.106 можем наблюдать результат реализованного кода в области «Скрипт».

Используя разработанный алгоритм на рисунке 3.105, его можно применить также для реализации кода узла «Обработка 2». На рисунке 3.107 изображен результат вставленного кода в область «Скрипт» в свойствах узла «Обработка2».

Используя разработанный алгоритм на рисунке 3.105, его можно применить для реализации кода узла «Обработка 3». Для этого код необходимо вставить в область «Скрипт» в свойствах узла «Обработка3», результат представлен на рисунке 3.108.

Алгоритм, выполняющий сбор обработанных результатов и выполнения вычислений представлен на рисунке 3.109. Данный алгоритм разработан для узла «Обработка результатов» его принцип работы заключается в проверке наличия заявок, если они есть, то выполняется сбор заявок из каналов обслуживания, затем проверяется время периода в связи «Время». Если период изменился происходит вычисление среднего значения для каждой характеристики данного типа СМО. Если результаты вычислены в первый период, то они запоминаются и записываются в связь «Результаты» без деления, когда вычисления выполняются не в первый раз тогда обработанные заявки и заявки с отказами складываются, использую это значения. Каждый результат делится на количество пройденных периодов, затем полученные средние значения передаются на связь «Результаты».

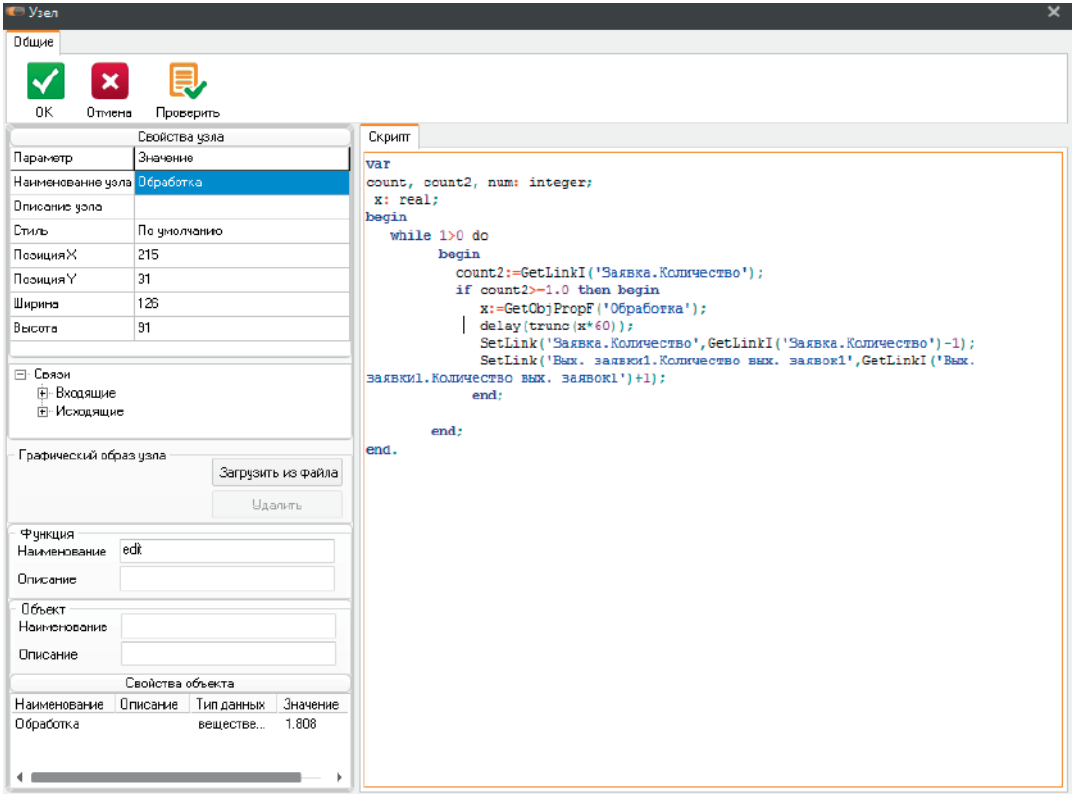


Рис. 3.106. Фрагмент кода узла «Обработка» многоканальной СМО с ограниченной длинной очереди

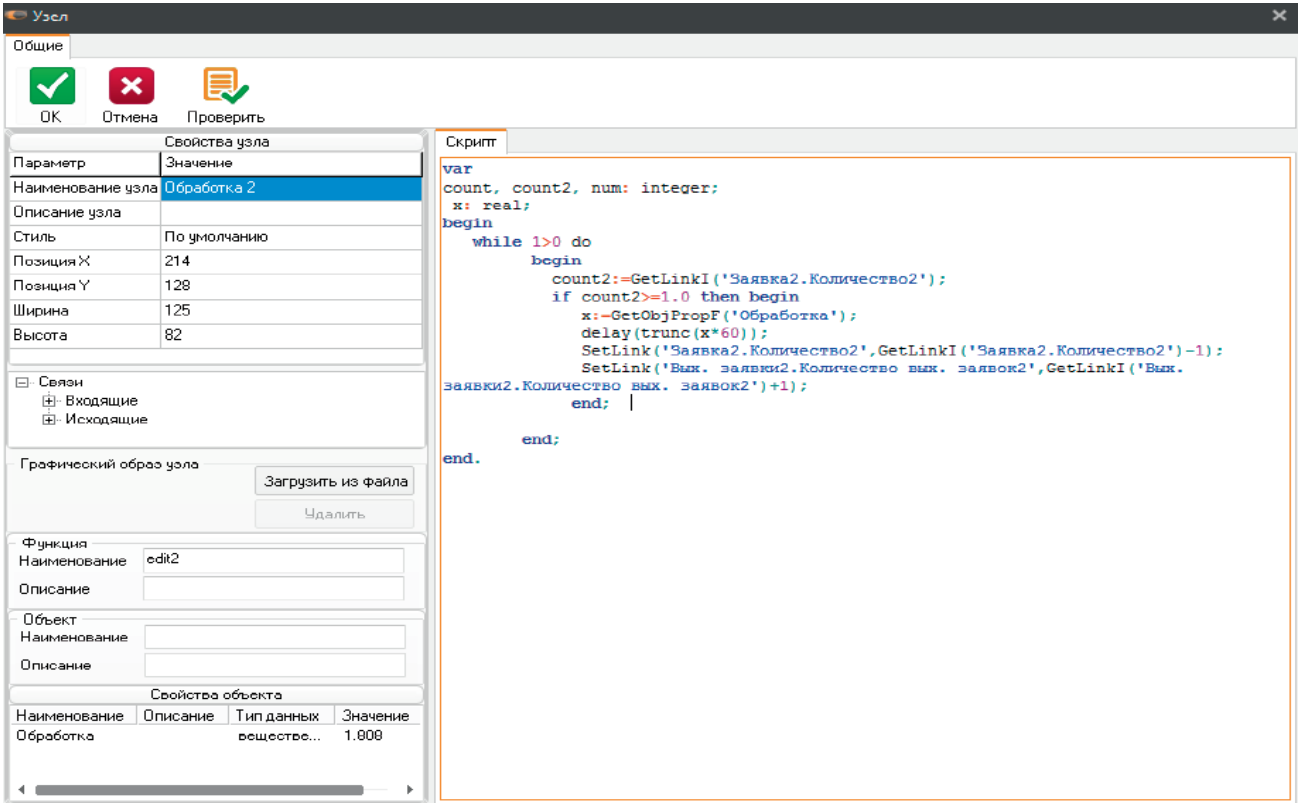


Рис. 3.107. Фрагмент кода узла «Обработка 2» многоканальной СМО с ограниченной длинной очереди

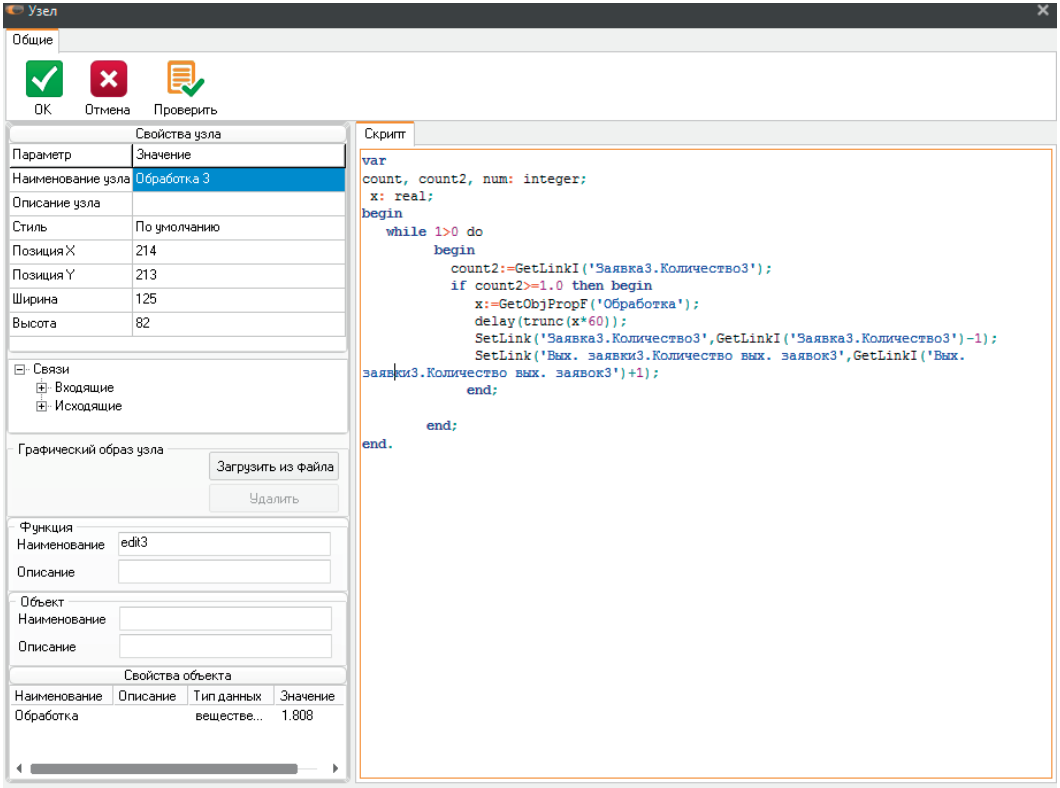


Рис. 3.108. Код узла «Обработка 3» многоканальной СМО с ограниченной длинной очереди

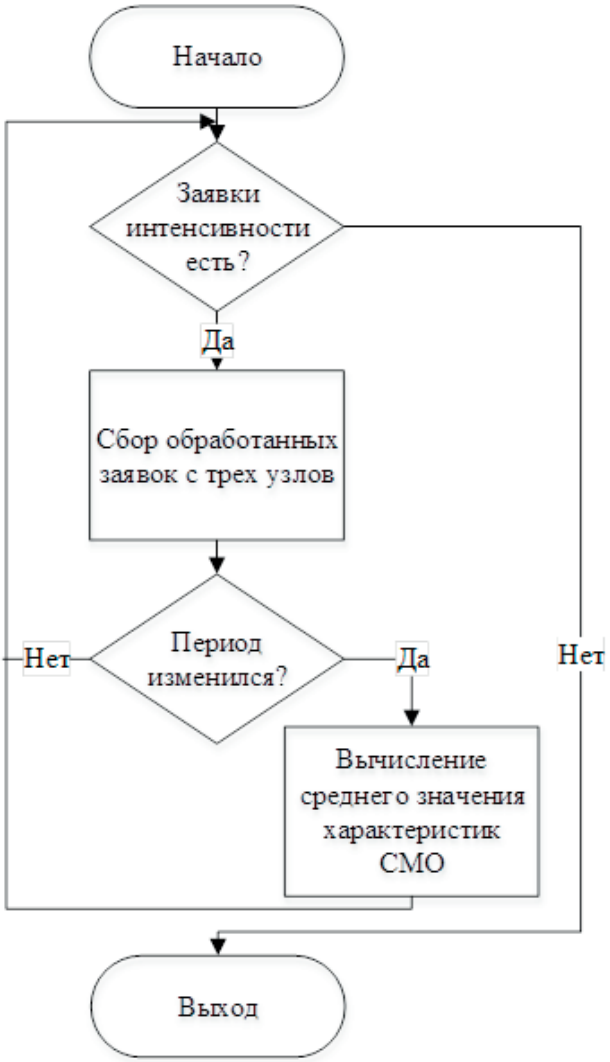


Рис. 3.109. Алгоритм узла «Обработка результатов» многоканальной СМО с ограниченной длинной очереди

По разработанному алгоритму был реализован код, фрагмент которого изображен на рисунке 3.110, а полный код приведен в Приложении Д. Для его работоспособности необходимо написать в свойстве узла «Обработка результатов» в области «Скрипт».

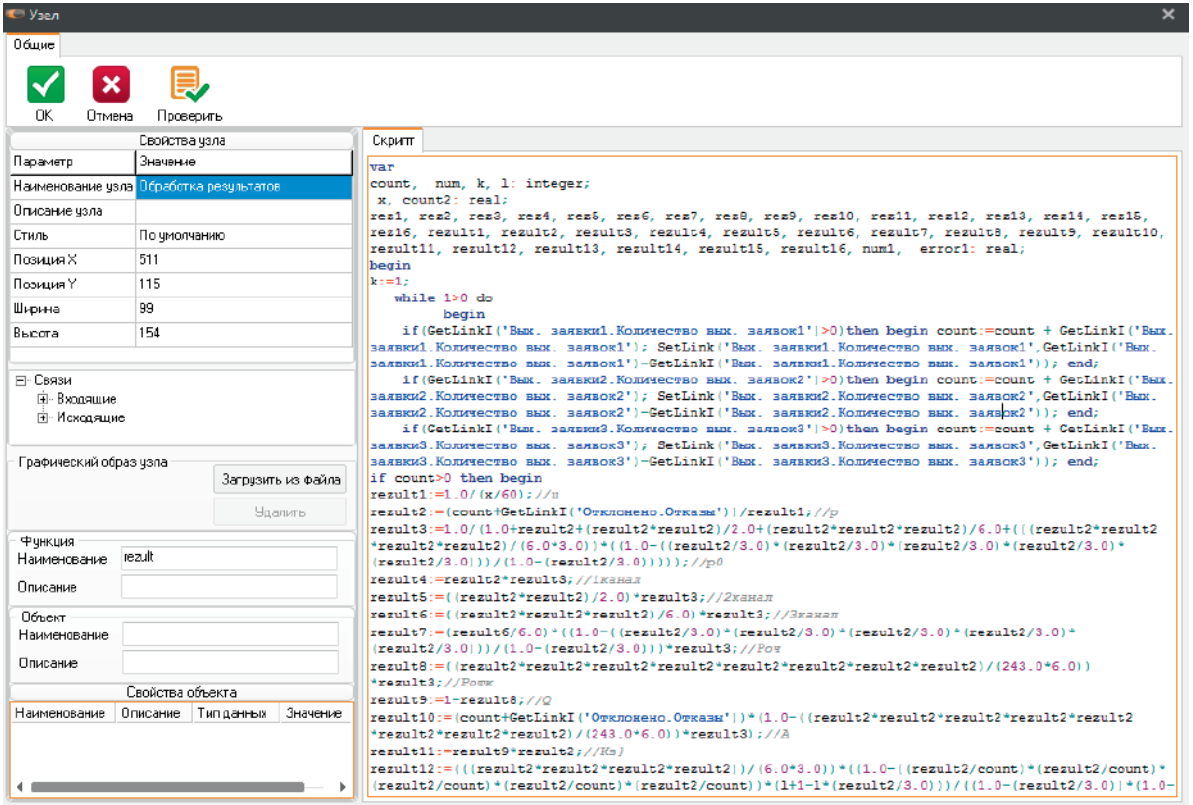


Рис. 3.110. Фрагмент кода узла «Обработка результатов» многоканальной СМО с ограниченной длинной очереди

Для имитационного моделирования использована интенсивность входящего потока заявок равным 110 заявки в час, длина очереди не более 5 заявок, интенсивность потока обслуживания для всех каналов обслуживания равным 1.8 минуты. Результаты, полученные входе моделирования системы представлены на рисунке 3.111.

Наименование	Значение
Интенсивность потока обслуживания	33.18584071
Интенсивность потока заявок	97.00000000
Интенсивность нагрузки	2.92293333
Вероятность, что канал свободен	0.03162854
Вероятность того, что обслуживанием занят 1 канал	0.09244812
Вероятность того, что обслуживанием занят 2 канала	0.13510984
Вероятность того, что обслуживанием занят 3 канала	0.13163902
Вероятность образования очереди	0.00329588
Вероятность отказа	0.11557740
Относительная пропускная способность	0.88442260
Абсолютная пропускная способность	85.78899238
Коффициент занятости каналов обслуживанием	2.58510830
Среднее число заявок, находящихся в очереди	194.35317495
Среднее число заявок, обслуживаемых в СМО	2.58510830
Среднее число заявок, находящихся в СМО	196.93828325
Среднее время пребывания заявки в СМО	2.03029158
Среднее время пребывания заявки в очереди	2.00364098
Число заявок получивших отказ	0.00000000
Пройденое время	1.00000000

Рис. 3.111. Результаты многоканальной СМО с ограниченной длинной очереди

Таблица 3.21

Вычисления по формулам для многоканальной СМО с ограниченной очередью

Название		**Результат**
Длина очереди m=		5
Количество каналов n=		3
Интенсивность потока заявок λ=		110
Интенсивность потока обслуживания μ=		33.18584071
Интенсивность нагрузки p=		3.314666667
Вероятность, что канал свободен p0=		0.017476028
Вероятность того, что обслуживанием занят:	1канал	0.057927206
	2канал	0.09600469
	3канал	0.106074515
Вероятность образования очереди Роч=		0.001904677
Вероятность отказа Ротк=		0.174664276

Название	Результат
Относительная пропускная способность Q=	0.825335724
Абсолютная пропускная способность A=	90.78692966
Коэффициент занятости каналов обслуживанием Кз=	2.735712814
Среднее число заявок, находящихся в очереди Lоч=	2.31106965
Среднее число заявок, обслуживаемых в СМО Lобс=	2.735712814
Среднее число заявок, находящихся в СМО Lcmo=	5.046782464
Среднее время пребывания заявки в СМО Tcmo=	0.045879841
Среднее время пребывания заявки в очереди Tоч=	0.021009724

Результаты, полученные при вычислении с помощью формул равны результатам полученные при моделировании. На рисунке 3.112 изображены графики работы многоканальной СМО с ограниченной очередью, 1 график – первый канал обслуживания, 2 график – второй канал обслуживания.

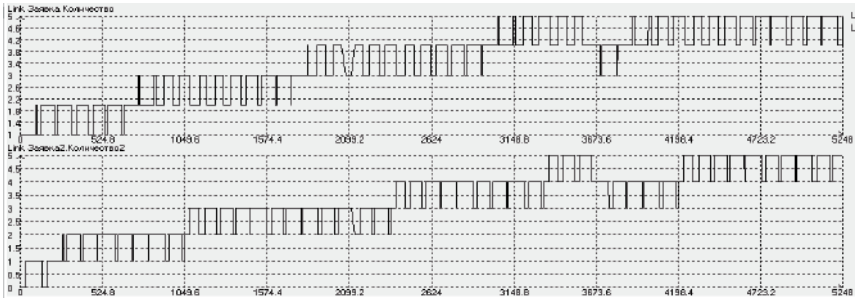


Рис. 3.112. Графики обслуживания первого и второго канала

На рисунке 3.113 представлены следующие графики: 1 график – третий канал обслуживания, 2 график – количество заявок, получивших отказ, т.е. поступившая заявка покинула систему не обслуженной. Проанализировав результаты вычислений и графики, полученные при имитационном моделировании, можно сказать что из-за того, что результат интенсивности нагрузки больше количества каналов, вследствие чего возникла подобная ситуация, при которой система не смогла справиться с обработкой данных.

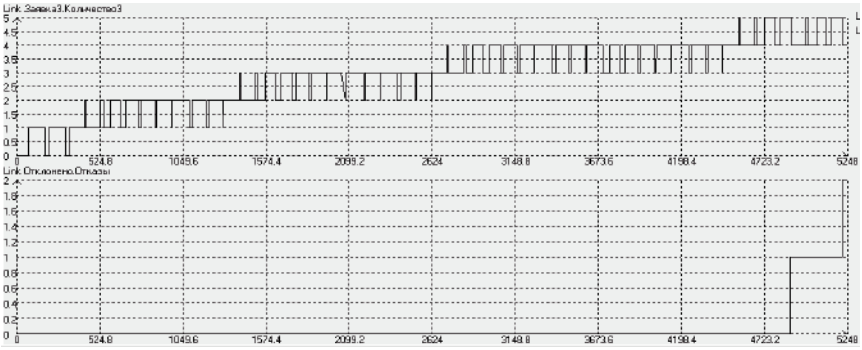


Рис. 3.113. Графики обслуживания третьим каналом и количество отказов

На основании проведенных тестов по различным видам СМО, можно сделать вывод, что UFOModeler обеспечивает заданную точность при построении имитационных моделей систем разлиной природа, а также систем массового обслуживания, в частности.

ЗАКЛЮЧЕНИЕ

Приведенные в представленной книге рассуждения и факты позволяют сделать следующие выводы:

1. Понятие «система» не является средством, обеспечивающим удобство изучения и описания исследуемых объектов, по аналогии с понятием «множество». Системы существуют в реальной действительности, т.е. имеют онтологический статус. Мир устроен системно! Причем существуют и системы-явления, и системы-классы.

2. Традиционные теоретические рассуждения о системах не позволяют создать действительно системную теорию (общую или абстрактную) в связи с неконструктивностью традиционного системного подхода.

3. Описанный выше системно-объектный подход позволяет создать теорию систем, обладающую объяснительными и предсказательными возможностями.

4. Предлагаемая теория систем обладает хорошим потенциалом для ее формализации с помощью различных алгебраических средств.

5. Представленные элементы новой системной теории позволяют, в свою очередь, создать метод системно-объектного графоаналитического моделирования, обеспечивающий учет большего количества общесистемных принципов и закономерностей, чем учитывают средства, используемые до него.

СПИСОК ЛИТЕРАТУРЫ

Ackoff R. L. General system theory and systems research: Contrasting conceptions of system science. // In Proceedings of the Second Systems Symposium at Case Institute of Technology, 1964.

Coad P., Yourdon E. Object-Oriented Analysis. New Jersey: Prentice-Hall, 1990.

Bondarenko M.F., Elchaninov D.B., Solov'eva E.A. and Matorin S.I. Systemological and Mathematical Foundations of a Natural Classification // International Journal on Information Theories & Applications. Sofia: FOI-COMMERCE. 2001. V.8. No.3. P. 151–157.

Zhikharev A., Matorin S., Egorov I. Formal principles of system-object simulation modeling of technological and production processes // Journal of Advanced Research in Dynamical and Control Systems. 2018. V. 10 (10 Special Issue). P.1806–1812.

Matorin S. I. Modelling Intelligent Understanding Of The Language Of Business Communication // Automatic Document and Mathematical Linguistics. New York: Allerton Press, Inc. 1998. V.31. No.2. P. 47–58.

Matorin S.I. A New Method Of Systemological Analysis Coordinated With The Object-Oriented Design Procedure. I // Cybernetics and Systems Analysis. Plenum Publishing Corporation. 2001. Вып. 37. № 4. С. 562–572.

Matorin S.I. A New Method Of Systemological Analysis Coordinated With The Object-Oriented Design Procedure. II // Cybernetics and Systems Analysis. Plenum Publishing Corporation. 2002. Вып. 38. № 1. С. 100–109.

Matorin S.I. System analysis of human evolution // Современные исследования социальных проблем (электронный научный журнал). 2013. № 8.

Matorin S.I., Zhikharev A.G. Calculation of the function objects as the systems formal theory basis // Advances in Intelligent Systems and Computing. 2018. №679. P. 182–191.

Matorin S.I., Zhikharev A.G., Zimovets O.A. Object Calculus in the System–Object Method of Knowledge Representation // Scientific and Technical Information Processing. 2018. Vol. 45. No. 5. P. 1–10.

Miller J.G. The Nature of Living Systems // Behavioral Science. 1975. V. 20. No. 6. P. 345–365.

Milner R., Parrow J., Walker D.A. Calculus of Mobile Processes. Part I. LFCS Report 89–85. University of Edinburgh, 1989. 46 p.

Ross R. Entity Modeling: Techniques and Application. Boston, MA: Database Research Group, 1987.

SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission 21 May 2004 / National Research Council of Canada, Network Inference, and Stanford University. Режим доступа: http://www.w3.org/Submission/SWRL/#8.4.

Агошкова Е.Б., Ахлибинский Б.В., Флейшман Б.С. Системология: сущность и место в научном знании // Синергетика и методы науки. СПб.: Наука, 1998. С. 63–76.

Андрианов Б.В. Исторический прогресс: хозяйственно-культурные аспекты // Природа. 1989. №3. С.75–82.

Безматерных В.Н. Что такое системный подход? Зачем он нужен? Алексеев П.В., Чернавский Д.С. и Виноград Э.Г. о системном подходе. Режим доступа: https://files.scienceforum.ru/pdf/2017/31116.pdf.

Белов С.П., Зимовец О.А., Маторин С.И. Формализация графических моделей административных процедур и их описание на языке исполнения бизнес-процессов // Научные ведомости БелГУ. Сер. Информатика. 2014. № 15 (186). Выпуск № 31/1. С. 128–138.

Бондаренко М.Ф., Маторин С.И., Соловьева Е.А. Анализ системологического инструментария концептуального моделирования проблемных областей // НТИ. Сер. 2. 1996. №4. С. 1–11.

Бондаренко М.Ф., Соловьева Е.А., Маторин С.И. Основы системологии. Харьков: Изд-во ХТУРЭ, 1998. 118с.

Бондаренко М.Ф. Маторин С.И., Соловьева Е.А. Моделирование и проектирование бизнес-систем: методы, стандарты, технологии / Под ред. Э.В. Попова. Харьков: «Компания СМИТ», 2004. 272 с.

Бреховских С.М. Основы функциональной системологии материальных объектов. М.: Наука, 1986. 192 с.

Бурков В.Н., Новиков Д.А. Теория активных систем: состояние и перспективы. М.: СИНТЕГ, 1999. 128 с.

Бусленко Н.П. Моделирование сложных систем. М.: Наука, 1978. 399 с.

Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя. М.: ДМК, 2000. 432 с.

Буч Гради, Роберт А. Максимчук, Майкл У. Энгл, Бобби Дж. Янг, Джим Коналлен, Келли А. Хьюстон Объектно-ориентированный анализ и проектирование с примерами приложений. СПб.: М.: «Вильямс». 2010. 720 с.

Вендров А.М. Объектно-ориентированный анализ и проектирование информационных систем с помощью Rational Rose. М.: «Академия АйТи», 2000. 210 с.

Вендров А.М. Объектно-ориентированный анализ и проектирование информационных систем с помощью Rational Rose. Режим доступа: http://www.raskatova.ru/files/doc01.pdf.

Вернадский В.И. Научная мысль как планетное явление. М.: Наука, 1991. 271 с.

Верников Г. Основные методологии обследования организации. Стандарт IDEF0. Режим доступа: http:// www.consulting.ru/main/mgmt/texts/m7/079_idef.shtml.

Верников Г. Основные методологии обследования организации. Стандарт IDEF0 Режим доступа: http://www.consulting.ru/main/mgmt/texts/m7/080_idef.shtml.

Верников Г. Основы методологии IDEF1, IDEF1X, IDEF3, IDEF5. Режим доступа: http://www.citforum.ru/cfin/vernikov.

Винер Н. Творец и робот. М.: Прогресс, 1960. 104 с.

Волкова В.Н. Теория систем и системный анализ в управлении организациями: Справочник: Учеб. пособие / Под ред. В.Н. Волковой и А.А. Емельянова. М.: Финансы и статистика, 2006. 848 с.

Волкова В.Н., Денисов А.А. Теория систем и системный анализ: учебник для академического бакалавриата. М.: Издательство Юрайт, 2015. 616 с.

Выдержки из перевода спецификации к нотации BPMN компании DIRECTUM. Режим доступа: http://www.DIRECTUM-Journal.ru/docs/1624827.html

Гвишиани Д.М. Методологические проблемы моделирования глобального развития. М.: 1977. 28 с. (Препр. / ВНИИСИ)

Гвишиани Д.М. Материалистическая диалектика – философия основы системных исследований // Системные исследования: Ежегодник, 1979. М.: Наука, 1980. С. 7–28.

Гернек Ф. Альберт Эйнштейн. М.: Прогресс, 1966. 107 с.

Гиг Дж. ванн. Прикладная общая теория систем. М.: Мир, 1981. Т.1. 336 с.

Глушков В.М. Введение в АСУ. К.: Техніка, 1972. 312 с.

Гренандер У. Лекции по теории образов. 1 Синтез образов. / Пер с англ. М.: Мир, 1979. 384 с.

Диалектика процесса познания / Под ред. М.Н. Алексеева и А.М. Коршунова. М.: Изд-во МГУ, 1985.

Донских О.А. К истокам языка. Новосибирск: Наука. Сиб. отдние, 1988. 192 с.

Дубейковский В.И. Практика функционального моделирования с AllFusion Process Modeler 4.1. Где? Зачем? Как? М.: ДИАЛОГ–МИФИ, 2004. 464 с.

Дубровский В.Я. К разработке системных принципов: общая теория систем и альтернативный подход. Режим доступа: http://gtmarket.ru/laboratory/expertize/6566.

Емельянов А.А., Емельянова Н.З. Имитационное моделирование и компьютерный анализ экономических процессов. Смоленск: Издательство «Универсум», 2014. 230 с.

Жихарев А.Г., Маторин С.И. Метод формализации организационных знаний // Искусственный интеллект и принятие решений. 2011. № 2. С. 12–18.

Жихарев А.Г., Маторин С.И., Маматов Е.М., Смородина Н.Н. О системно-объектном методе представления организационных знаний // Научные ведомости БелГУ. Сер. Информатика. 2013. №8(151). Выпуск №26/1. С. 137–146.

Жихарев А.Г. Формализованное графоаналитическое представление организационных знаний // Автореф. дисс. ... канд. техн. наук. Белгород, 2013. 23 с.

Жихарев А.Г., Маторин С.И., Зайцева Н.О. Разработка средств и методов имитационного моделирования транспортных потоков города // Научные ведомости БелГУ. Сер. Информатика. 2014. №1(172). Выпуск №29/1. С. 66–69.

Жихарев А.Г., Маторин С.И., Зайцева Н.О. Системно-объектный инструментарий для имитационного моделирования технологических процессов и транспортных потоков // Искусственный интеллект и принятие решений. 2015. №4. С. 95–103.

Жихарев А.Г., Маторин С.И., Рябцева Я.Н., Махота А.С., Капустин А.В. Об имитационном моделировании функционирующих систем // Научные ведомости БелГУ. Сер. Экономика. Информатика. 2016. № 9(230). Выпуск 38. С. 139–146

Забродин В.Н. О критериях естественности классификации // НТИ. – Сер.2. – 1981. – №8. – С.92–112.

Заец Р.В. Содержание системного анализа проблем городского функционирования и развития // Проблемы информатики города. К.: Наук. Думка, 1990. С. 91–110.

Зимовец О.А., Зиньков С.В., Маторин С.И. Формализация представления знаний в ЭВМ и автоматизация процедуры их моделирования с применением математического аппарата теории паттернов // Вопросы радиоэлектроники. Сер. ЭВТ. №1. - 2008. - С. 18-32

Зимовец О.А., Маторин С.И. Представление диаграмм в нотациях DFD, IDEF0 и BPMN с помощью системно-объектных моделей «Узел – Функция – Объект» // Научные ведомости БелГУ. Сер. Информатика. 2011. № 19 (114). Выпуск № 20/1. С. 126–136.

Зимовец О.А., Маторин С.И. Моделирование административных процедур с использованием системного подхода «Узел – Функция – Объект» // Научные ведомости БелГУ. Сер. Информатика. 2012. № 1 (120). Выпуск № 21/1. С. 166–172.

Зимовец О.А., Маторин С.И. Формализованное визуальное моделирование административных процедур // Прикладная информатика. 2012. № 2 (38). С. 100–110.

Зимовец О.А., Маторин С.И. Интеграция средств формализации графоаналитических моделей «Узел-Функция-Объект» // Искусственный интеллект и принятие решений. 2012. №1. С. 57–64.

Зимовец О.А., Маторин С.И. Формально-семантическое описание графоаналитических моделей административных процедур // Научные ведомости БелГУ. Сер. Информатика. 2012. № 7 (126). Выпуск № 22/1. С. 129–136.

Зимовец О.А., Маторин С.И., Цоцорина Н.В. Гуль С.В. Исчисление функций ─ алгебраический аппарат процессного подхода // Научные ведомости БелГУ. Сер. Информатика. 2014. № 21(192). Выпуск (32/1). С. 154–161.

Зимовец, О.А. Методы компьютерной визуализации и трансформации административных процедур // Автореф. дисс. ... канд. техн. наук. Белгород. 2014. 23 с.

Игнатьев М.Б. Моделирование сложных систем // Синергетика и методы науки. СПб.: Наука. 1998. С. 425–431.

Йордан Э., Аргила К. Структурные модели в объектно-ориентированном анализе и проектировании. М.: «ЛОРИ». 1999, 264 с.

Калашников В.В. Сложные системы и методы их анализа. М.: Знание, 1980. 64 с.

Калянов Г.Н. Консалтинг при автоматизации предприятий. М.: СИНТЕГ, 1997. 316 с.

Кватрани Т. Rational Rose 2000 и UML. Визуальное моделирование. М.: ДМК Пресс, 2001. – 176 с.

Кликс Ф. Пробуждающееся мышление. История развития человеческого интеллекта / Пер. с нем. К.: Выща шк. Изд-во при Киев. ун-те, 1985. 295 с.

Клир Д. Системология: автоматизация решения системных задач. М.: Радио и связь, 1990. 539 с.

Колесников С. Методики моделирования в бизнесе Режим доступа: http://www.consulting.ru/main/ mgmt/texts/m3/032_method1.htm.

Колесников С. Что такое КИС и как с ней бороться. Режим доступа: http://www.consulting.ru/main/soft/texts/m4/043_s_red0-2.htm.

Кондаков Н.И. Логический словарь-справочник. М.: Книга по Требованию, 2012. 721 с.

Кондратенко А.А. Представление используемой в УФО-подходе базовой классификации связей с помощью модели RDF // Информатика: проблемы, методология, технологии: материалы XV международной конференции, Воронеж. В 3 т. Т.1. Воронеж: Издательско-полиграфический центр Воронежского государственного университета, февраль 2015. С. 102–107.

Кондратенко А.А., Маторин С.И. Формальные аспекты взаимосвязи УФО-подхода и языка представления онтологий RDF // Научные ведомости БелГУ. Сер. Экономика. Информатика. 2016. №2(223). Выпуск 37. С. 119–127.

Кондратенко А.А. Построение онтологий на основе системно-объектного подхода // Автореф. дисс. ... канд. техн. наук. Белгород. 2016. 23 с.

Кондратенко А.А., Маторин С.И. Логический вывод на визуальных графоаналитических УФО-моделях путем интеграции со средствами онтологического инжиниринга // Научные ведомости БелГУ. Серия Экономика. Информатика. 2016. №9 (230) Выпуск 38. С. 156–164.

Косарев Ю.Г. Вступительная статья // Системология и языковые аспекты кибернетики / Г.П. Мельников. М.: Сов. радио, 1978.

Косыгин Ю.А. Человек. Земля. Вселенная. М.: Наука, 1995. 335 с.

Ларман Крег Применение UML и шаблонов проектирования. М.: «Вильямс», 2001. 496 с.

Левин В.И. Структурно-логические методы исследования сложных систем с применением ЭВМ. (Теория и методы системного анализа). М.: Наука, 1987. 304 с.

Леоненков А. Самоучитель UML. СПб.: «БХВ-Петербург», 2001. 304 с.

Маклаков С.В. Моделирование бизнес-процессов с BPwin 4.0. М.: ДИАЛОГ–МИФИ, 2002. 224 с.

Маторин С.И., Соловьева Е.А. Детерминантная модель системы и системологический анализ принципов детерминизма и бесконечности мира // НТИ. Сер. 2. 1996. №8. С. 1–8.

Маторин С.И. Системологическое исследование структуры системы категорий // НТИ. Сер. 2. 1997. №3. С. 3–7.

Маторин С.И. О моделировании интеллектуального понимания языка делового общения // НТИ. Сер. 2. 1997. №4. С. 9–17.

Маторин С.И. Детерминантный анализ системы переработки информации человека // Проблемы бионики. 1998. №49. С. 72–80.

Маторин С.И. Анализ и моделирование бизнес-систем: системологическая объектно-ориентированная технология. Харьков: ХНУРЭ, 2002. 322 с.

Маторин С.И., Ельчанинов Д.Б. Применение теории паттернов для формализации системологического УФО-анализа // НТИ. Сер. 2. 2002. №11. С. 1–11.

Маторин С.И., Ельчанинов Д.Б., Зиньков С.В., Маторин В.С. Синтез и анализ систем в свете подхода «Узел-Функция-Объект». // НТИ. Сер. 2. 2006. №8. С. 10–16.

Маторин С.И. Системный подход к личной жизни // Сборник РФФИ. 2014. №17. С. 300–309.

Маторин С.И., Жихарев А.Г., Зайцева Н.О. Имитационное моделирование с использованием системно-объектного подхода. // Прикладная информатика. 2015. №6(60). Выпуск 10. С. 91–104.

Маторин С.И., Зимовец О.А. Теория систем и системный анализ. Белгород: Изд-во БУКЭП, 2016. 259 с.

Маторин С.И., Зимовец О.А., Жихарев А.Г. Общесистемные принципы в терминах системно-объектного подхода «Узел-Функция-Объект» // Труды ИСА РАН. 2016. №1. Том 66. С. 10–17.

Маторин С.И., Жихарев А.Г., Зимовец О.А. Обоснование взаимосвязей общесистемных принципов и закономерностей с позиции системно-объектного подхода // Труды Института системного анализа. 2017. №3. Том 67. С. 54–63.

Маторин С.И., Жихарев А.Г. Общесистемные закономерности как содержательные элементы системной теории, основанной на системно-объектном подходе // Научные ведомости БелГУ. Сер. Экономика. Информатика. 2018. № 2. Том 45. С. 372–284.

Маторин С.И., Жихарев А.Г. Учет общесистемных закономерностей при системно-объектном моделировании организационных знаний // Искусственный интеллект и принятие решений. 2018. №3. С. 115–126.

Маторин С.И., Тубольцева О.М. Метод формализованного описания систем ресурсного обеспечения проектов // Вестник БУКЭП. 2019. №4(77). С. 73–83.

Мацяшек Л.А. Анализ требований и проектирование систем: разработка информационных систем с использованием UML. М.: «Вильямс», 2002. 432 с.

Мельник М.С. Формирование общей теории систем: результаты и проблемы исследования Режим доступа: https://cyberleninka.ru/article/n/formirovanie-obschey-teorii-sistem-rezultaty-i-problemy-issledovaniya.

Мельников Г.П. Системология и языковые аспекты кибернетики. М.: Сов. радио, 1978. 368 с.

Мельников Г.П., Преображенский С.Ю. Методология лингвистики. М.: Изд-во Ун-та дружбы народов, 1989. – 83 с.

Месарович М., Михайло Д., Такахара Я. Общая теория систем: математические основы. М.: Мир, 1978. 311 с.

Михелев М.В., Маторин С.И. Формализация УФО-элементов с помощью алгебраического аппарата пи-исчисления // Научные ведомости БелГУ. Сер. Информатика. 2010. №19(90). Выпуск №16/1. С.145–149.

Моисеев Н. Н. Универсальный эволюционизм и коэволюция // Природа. 1989. №4. С. 3–8.

Никаноров С.П. Системный анализ: этап развития методологии решения проблем в США // Системный анализ для решения деловых и промышленных проблем / С.Л. Оптнер. М.: Советское радио, 1969. С. 7–45.

Ойхман Е.Г., Попов Э.В. Реинжиниринг бизнеса. М.: Финансы и статистика, 1997. 336 с.

Оптнер Станфорд Л. Системный анализ для решения деловых и промышленных проблем. М.: Советское радио, 1969. 216 с.

Ошо Р. Психология эзотерического. Корни и крылья. М.: «АСТ», 1992. 435 с.

Панова Н.С., Шрейдер Ю.А. Принцип двойственности в теории классификации // НТИ. Сер. 2. М.: ВИНИТИ. 1975. №10. С. 1–8.

Перегудов Ф.И., Тарасенко Ф.П. Введение в системный анализ. М.: Высш. шк., 1989. 367 с.

Петров Ю.А. Методологические вопросы анализа научного знания. М.: «Высш. школа», 1977. 224с.

Полищук, Д.М., Хон В.Б. Теория автоматизированных банков информации. М., 1989. 184 с.

Попов Э.В., Фоминых И.Б., Кисель Е.Б., Шапот М.Д. Статические и динамические экспертные системы. М.: Финансы и статистика, 1996. 320с.

Прангишвили И.В. Системный подход и общесистемные закономерности. М.: СИНТЕГ, 2000, 528 с.

Процессный подход Режим доступа: http://www.kpms.ru/General_info/Process_approach.htm

Пугачев Н.Н. Теория, онтология и реальность. Воронеж: Изд-во Воронежск. Ун-та, 1991. 144 с.

Пьер Тейяр де Шарден Феномен человека. М.: Наука, 1987. 240 с.

Реинжиниринг бизнес-процессов: учебное пособие / Под ред. А.О. Блинова. М.: Юнита Дана, 2012. 684 с.

Родионов М. Г. Структурно-функциональный и системный анализ как инструменты организационного проектирования // Вестник Сибирского института бизнеса и информационных технологий. 2013. №2(6). С. 40–47.

Садовский В.Н. Основания общей теории систем. Логико-методологический анализ. М., 1974. 106 с.

Сахаров П. Rational Rose, BPwin и другие – аспект анализа бизнес-процессов // Директору информационной службы: электронный журн. Режим доступа: http://www.osp.ru/cw/cio/2000/011_0.htm.

Слободюк А.А. Расширенная классификация фактов, извлекаемых из УФО-модели в целях построения онтологии предметной области // Шестая международная научно-техническая конференция «Инфокоммуникационные технологии в науке, производстве и образовании» (Инфоком-6): сборник научных трудов. Часть II. Ставрополь: Северо-Кавказский федеральный университет, апрель 2014. С. 404–406.

Слободюк А.А., Маторин С.И. О возможности извлечения фактов из УФО-моделей и представлении их с помощью RDF. // Информатика: проблемы, методология, технологии: материалы XIV международной конференции. В 3 т. Т.1. Воронеж: Издательско-полиграфический центр Воронежского государственного университета, февраль 2014. С. 269–273.

Смирнов Э.А. Основы теории организации. М.: «Аудит», 1998. 375 с.

Современный философский словарь / Под общ. ред. В.Е. Кемерова и Т.Х. Керимова. 4-е изд., испр. и доп. М.: Академический проект; Екатеринбург: Деловая книга, 2015. 823 с.

Соловьев А.В. Экспериментальное исследование психологических механизмов формирования понятий: Автореф. дис. … канд. пед. наук / Моск. Гос. пед. ин-т. М.: 1973. 20 с.

Технология моделирования бизнес-процессов / НИЦ CALS-технологий «Прикладная логистика». Режим доступа: http:// http://www.cals.ru.

Тихомиров О.К. Психология мышления. М.: Изд-во МГУ, 1984. 271 с.

Трофимов С.А. CASE-технологии: практическая работа в Rational Rose. М.: ЗАО «Издательство БИНОМ», 2001. 272 с.

Тубольцев М.Ф., Маторин С.И., Тубольцева О.М., Михайлюк Е.А. Архитектура цифровых моделей систем финансирования инвестиционных проектов // Научные ведомости БелГУ. Сер. Экономика. Информатика. 2018. № 4. Том 45. С. 718–727.

Тубольцева О.М., Маторин С.И. Моделирование деловых процессов на основе специализированного УФО-метода // Научные ведомости БелГУ. Сер. Информатика. 2014. №15(186). Выпуск №31/1. С. 83–89.

Уемов А.И. Системный подход и общая теория систем. М.: Мысль, 1978. 272 с.

Урсул А.Д. Путь в ноосферу: Концепция выживания и устойчивого развития человечества. М.: Луч, 1993. 275 с.

Урсул А.Д. Информационная стратегия и безопасность в концепции устойчивого развития // НТИ. Сер. 1. М.: ВИНИТИ. 1996. №1. С. 1–9.

Успенский П. Д. Новая модель вселенной: Пер. с англ. СПб: Изд-во Чернышева, 1993. 560 с.

Фаулер М., Скот К. UML в кратком изложении. Применение стандартного языка объектного моделирования. М.: Мир, 1999. 191 с.

Франс Р., Гош С., Дин-Тронг Т., Соулберг Э. Разработка на базе моделей с использованием UML 2.0: обещания и просчеты // Открытые системы. 2006. №3. С. 34–43.

Фромм Э. Искусство любить. Исследование природы любви / Перевод Л. А. Чернышёвой. М.: Педагогика, 1990. 160 с.

Хомяков П.М. Системный анализ: Краткий курс лекций. М.: КомКнига, 2006.

Чупахин И.Я. Методологические проблемы теории понятий. Л.: Изд-во ЛГУ, 1973. 104 с.

Шлеер С., Меллор С. Объектно-ориентированный анализ: моделирование мира в состояниях. К.: Диалектика, 1993. 240 с.

Шрейдер Ю.А., Шаров А. А. Системы и модели. М.: Радио и связь, 1982. 152 с.

Шрейдер Ю.А. Теория познания и феномен науки // Гносеология в системе философского мировоззрения. М.: Наука, 1983. С. 173–193.

ПРИЛОЖЕНИЯ

Приложение 1. Сравнение возможностей учета общесистемных закономерностей различными средствами графоаналитического моделирования

Общесистемные принципы и закономерности	**Системно-структурный подход (системно-функциональный и процессный)**	**Объектно-ориентированный подход**	**Нотация BPMN**	**Системно-объектный подход**
УЗЛОВЫЕ/структурные				
Коммуникативности (система связана множеством коммуникаций с окружающей средой)	На контекстной диаграмме в виде функциональных (внешних) связей	На диаграмме вариантов использования в виде связей с акторами	Нет	На контекстной диаграмме в виде функциональных (внешних) связей
Иерархичности (система на любом ярусе иерархии является частью системы более высокого яруса, т.е. надсистемы)	В виде иерархии диаграмм декомпозиции произвольного числа уровней	На диаграмме классов	В виде BPD-диаграммы, показывающей не более трех уровней иерархии	В виде иерархии диаграмм декомпозиции произвольного числа уровней
Моноцентризма (устойчивая система обладает одним центром)	В виде контекстной диаграммы с одним процессом	Нет	Нет	В виде контекстной диаграммы с одной системой
Организационной непрерывности (констатирует факт наличия между всякими двумя системами звеньев, вводящих их в одну «цепь ингресси»)	Путем прослеживания связей между процессами в рамках всей модели	Путем прослеживания связей в рамках одного прецедента	Путем прослеживания связей в рамках одной BPD-диаграммы	Путем прослеживания связей между процессами в рамках всей модели

Общесистемные принципы и закономерности	Системно-структурный подход (системно-функциональный и процессный)	Объектно-ориентированный подход	Нотация BPMN	Системно-объектный подход
Обратной связи (устойчивость в сложных динамических системах достигается за счёт замыкания петель обратных связей)	В виде необязательных двухсторонних связей и на контекстной диаграмме, и на диаграммах декомпозиции	Нет	В виде необязательных двухсторонних связей на конкретной BPD-диаграмме	В виде двухсторонних связей и на контекстной диаграмме, и на диаграммах декомпозиции в соответствии с правилом замкнутости
Внешнего дополнения (восходящие к системному центру воздействия координируемых элементов подвергаются своеобразному «обобщению», а нисходящие от системного центра координационные импульсы подвергаются «специфицированию» в зависимости от характера локальных процессов за счет обратных связей от этих процессов)	Возможность содержательной декомпозиции потоков объектов и информации при декомпозиции процессов (классификация потоков связей не предусмотрена)	Нет. Связи не рассматриваются как потоки объектов и информации	Нет. Связи не рассматриваются как потоки объектов и информации, а только как средство отображения порядка действий или передачи сообщений	Возможность содержательной декомпозиции материальных и информационных потоков при декомпозиции процессов (задана содержательная категориальная классификация потоков связей)
Взаимно-дополнительных соотношений/комплиментарности (устойчивость системы достигается взаимно-дополнительными связями между её элементами в виде замкнутых контуров обратных связей)	В виде необязательных двухсторонних связей и на контекстной диаграмме, и на диаграммах декомпозиции	Нет	В виде необязательных двухсторонних связей на конкретной BPD-диаграмме	В виде двухсторонних связей и на контекстной диаграмме, и на диаграммах декомпозиции в соответствии с правилом замкнутости

Общесистемные принципы и закономерности	**Системно-структурный подход (системно-функциональный и процессный)**	**Объектно-ориентированный подход**	**Нотация BPMN**	**Системно-объектный подход**
Прогрессирующей сегрегации (фиксирует прогрессирующую потерю взаимодействия между элементами системы в ходе ее дифференциации при усилении связей с некоторым элементом, выступающим в роли системного центра)	На диаграмме декомпозиции любого уровня в виде сокращения внутренних связей и появления новых граничных связей	Нет	Нет	На диаграмме декомпозиции любого уровня в виде сокращения внутренних связей и появления новых граничных связей
ФУКЦИОНАЛЬНЫЕ				
Прогрессирующей механизации (части системы в ходе ее развития специализируются или становятся фиксированными по отношению к определенным функциям или механизмам)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня	Нет. Иерархия процессов не может быть представлена	Задачи являются частями подпроцессов, которые являются частями одного процесса (только три уровня)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня
Актуализации функций (объект выступает как организованный лишь в том случае, если свойства его частей (элементов) проявляются как функции сохранения и развития этого объекта)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня	Нет. Иерархия процессов не может быть представлена	Задачи являются частями подпроцессов, которые являются частями одного процесса (только три уровня)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня

Общесистемные принципы и закономерности	Системно-структурный подход (системно-функциональный и процессный)	Объектно-ориентированный подход	Нотация BPMN	Системно-объектный подход
Самоорганизации (процесс поступательной функционализации элементов системы)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня	Нет. Иерархия процессов не может быть представлена	Задачи являются частями подпроцессов, которые являются частями одного процесса (только три уровня)	Подпроцессы на иерархии диаграмм любого уровня декомпозиции являются поддерживающими частями процесса верхнего уровня
Иерархических компенсаций (в системе рост разнообразия на верхнем уровне иерархии обеспечивается его ограничением на более низких уровнях)	В работающей системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям	Нет. Иерархия процессов не может быть представлена	В работающей системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям	В работающей системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям
Необходимого разнообразия (для создания системы, способной справится с решением проблемы, обладающей определенным разнообразием, необходимо обеспечить, чтобы система имела большее разнообразие возможностей, чем разнообразие решаемой проблемы)	В создаваемой системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям	Нет. Иерархия процессов не может быть представлена	В создаваемой системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям	В создаваемой системе процессы верхнего уровня соответствуют более общим разнообразным функциям. Процессы нижнего уровня соответствуют более специальным ограниченным функциям

Общесистемные принципы и закономерности	Системно-структурный подход (системно-функциональный и процессный)	Объектно-ориентированный подход	Нотация BPMN	Системно-объектный подход
Семиотической непрерывности (система есть образ её среды, т.е. система как элемент окружающей среды отражает некоторые существенные ее свойства)	Функциональный запрос надсистемы к системе отображается в виде функциональных связей процесса на любом уровне иерархии	Функциональный запрос надсистемы к системе отображается в виде связей с акторами на диаграмме вариантов использования	Нет	Функциональный запрос надсистемы к системе отображается в виде функциональных связей процесса на любом уровне иерархии
ОБЪЕКТНЫЕ/субстанциальные				
Совместимости (условием взаимодействия между системами является наличие у них относительной совместимости, то есть относительной качественной и организационной однородности)	Нет	Нет	Нет	Обеспечивается установленными правилами системной композиции, базовой иерархией связей и классификацией узлов
Эквифинальности (способность системы достигать состояния, которое не зависит от времени и начальных условий, а зависит только от параметров системы)	Нет	Нет	Нет	- « -
Минимума (устойчивость системы определяется устойчивостью ее самого слабого звена)	Нет	Нет	Нет	- « -

Общесистемные принципы и закономерности	Системно-структурный подход (системно-функциональный и процессный)	Объектно-ориентированный подход	Нотация BPMN	Системно-объектный подход
Опыта (единообразное воздействие на некоторое множество элементов уменьшает разнообразие состояний этого множества)	Нет	Нет	Нет	- « -
Расхождения (различные части однородной системы подвержены действию сил, различающихся по качеству и величине, вследствие чего они изменяются различно)	Нет	Нет	Нет	Обеспечивается возможностью описания процесса адаптации системы к запросу надсистемы за счет изменения объектных характеристик
Эмерджентности (возникновения у системы новых интегративных свойств, отсутствующих у ее элементов)	Необходимое условие: принципы иерархичности и моноцентризма. Достаточное условие: принцип актуализации функций и закон иерархических компенсаций			

Приложение 2. Описание языка УФО-скрипт

Таблица 1

Операторы языка УФО-скрипт

Наименование	Параметры	Описание
begin Оператор 1; Оператор 2; … Оператор N **end**		Составной оператор
if Условие **then** Оператор	Условие:boolean – логическое выражение	Условный оператор (сокращенная форма)
if Условие **then** Оператор1 **else** Оператор2	Условие:boolean – логическое выражение	Условный оператор (полная форма)
while Условие **do** Оператор	Условие:boolean – логическое выражение	Цикл с предусловием: выполняется пока Условие дает результат True.
repeat Оператор 1; Оператор 2; … Оператор N **until** Условие	Условие:boolean – логическое выражение	Цикл с пост условием: выполняется до тех пор, пока Условие не даст результат True
for Переменная := Значение1 **to** / **downto** Значение2 **do** Оператор	Переменная – счетчик цикла; Значение1 – начальное значение; Значение2 – конечное значение	Цикл с параметрами
case Выражение **of** Метка1: Оператор1; Метка2: Оператор2; … МеткаN: ОператорN; **else** ОператорElse 1; ОператорElse 2; … ОператорElse N **end**	Выражение – выражение, по значению которого выполняется переход к оператору с соответствующей меткой	Оператор выбора. Выполняется оператор в зависимости от значения выражения. Если не совпало ни одно значение, то выполняется список операторов в разделе между служебными словами else и end

Таблица 2

Процедуры и функции – работа с объектными характеристиками системы

Наименование	Параметры	Описание
GetObjPropI (propertyName:string):integer;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
GetObjPropF (propertyName:string):real;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
GetObjPropS (propertyName:string):string;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
GetObjPropB (propertyName:string):boolean;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
SetObjProp (propertyName:string; value: integer);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
SetObjProp (propertyName:string; value: real);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
SetObjProp (propertyName:string; value: integer; time:integer);	propertyName – наименование свойства; value – устанавливаемое значение; time – время, в течение которого должно установиться значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы

Наименование	Параметры	Описание
SetObjProp (propertyName:string; value: real; time:integer);	propertyName – наименование свойства; value – устанавливаемое значение; time – время, в течение которого должно установиться значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
SetObjProp (propertyName:string; value: string);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы
SetObjProp (propertyName:string; value: boolean);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы

Таблица 3

Процедуры и функции – работа с исходящими и входящими потоковыми объектами

Наименование	Параметры	Описание
GetLinkI (linkName:string):integer;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkF (linkName:string):real;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkS (linkName:string):string;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkB (linkName:string):boolean;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkInI (linkName:string):integer;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkInF (linkName:string):real;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkInS (linkName:string):string;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы

Наименование	Параметры	Описание
GetLinkInB (linkName:string):boolean;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkOutI (linkName:string):integer;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkOutF (linkName:string):real;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkOutS (linkName:string):string;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
GetLinkOutB (linkName:string):boolean;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLink (linkName:string; value: integer);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLink (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLink (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы

Наименование	Параметры	Описание
SetLink (linkName:string; value: boolean);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkIn (linkName:string; value: integer);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkIn (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkIn (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkIn (linkName:string; value: boolean);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkOut (linkName:string; value: integer);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkOut (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
SetLinkOut (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы

Наименование	Параметры	Описание
SetLinkOut (linkName:string; value: boolean);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению связи указанное значение (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению связи указанное значение (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkInAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению входящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkInAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению входящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkOutAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению исходящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы
LinkOutAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению исходящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы

Таблица 4

Арифметические функции

Наименование	Параметры	Описание
Abs(x: integer): integer;	x – аргумент функции	модуль числа
Abs(x:real):real;	x – аргумент функции	модуль числа
ArcTan(x:real):real;	x – аргумент функции	арктангенс (тригонометрическая функция)
Cos(x:real):real;	x – аргумент функции	косинус (тригонометрическая функция)
Exp(x:real):real;	x – аргумент функции	экспонента
Frac(x:real):real;	x – аргумент функции	дробная часть числа
Odd(x:integer):boolean;	x – аргумент функции	является ли число x нечетным
Ord(x:string):integer;	x – аргумент функции	ASCII-код первого символа строки
Random():real;		Генератор случайных вещественных чисел в диапазоне от 0 до 1.
Random(x:integer):integer;	x – аргумент функции	Генератор случайных целых чисел в диапазоне от 0 до x-1
Round(x:real):integer;	x – аргумент функции	округление числа (до 0,5 в меньшую сторону, иначе – в большую)
Sin(x:real):real;	x – аргумент функции	синус (тригонометрическая функция
Sqrt(x:real):real;	x – аргумент функции	квадратный корень (x-неотрицательное число)
Sqr(x:real):real;	x – аргумент функции	квадрат вещественного числа
Sqr(x:integer):integer;	x – аргумент функции	квадрат целого числа
Trunc(x:real):integer;	x – аргумент функции	урезание дробной части числа

Таблица 5

Прочие процедуры и функции

Наименование	Параметры	Описание
Delay (value:integer);	value – время задержки (секунды)	Задержка в выполнении программы текущего узла в секундах
DelayDay (value:integer);	value – время задержки (дни)	Задержка в выполнении программы текущего узла в днях
Write(value1, value2, …, valueN)	value1, value2, …, valueN – список выводимых значений (любого типа: integer, real, string, boolean)	Вывод информации на консоль без перевода строки
WriteLn(value1, value2, …, valueN)	value1, value2, …, valueN – список выводимых значений (любого типа: integer, real, string, boolean)	Вывод информации на консоль с последующим переводом строки
WriteLn()	-	Перевод строки консоли