Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»

Кафедра информационных технологий и вычислительных систем

На правах рукописи



ГАВРИЛОВ АНДРЕЙ ГЕННАДЬЕВИЧ

РАЗРАБОТКА МЕТОДА МОДЕЛИРОВАНИЯ И СРЕДСТВ ПОДДЕРЖКИ УПРАВЛЕНИЯ РАЗВИТИЕМ ВИЗУАЛЬНОЙ ИНТЕГРИРОВАННОЙ СРЕДЫ ПРОЕКТИРОВАНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

2.3.1 – Системный анализ, управление и обработка информации

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
доктор технических наук, профессор
Волкова Галина Дмитриевна

Москва – 2022

ОГЛАВЛЕНИЕ

Введение 4

Глава 1 Аналитический существующих подходов, методов и средств создания автоматизированных систем 9

1.1 Анализ существующих подходов и методов, применяемых при создании АС 9

1.1.1 Общие положения и обоснования 9

1.1.2 Характеристика структурного подхода 10

1.1.3 Характеристика объектно-ориентированного подхода 16

1.1.4 Характеристика когнитивного подхода 21

1.1.5 Сравнительный анализ рассмотренных подходов и методов, применяемых при создании АС 25

1.2 Анализ средств поддержки методов объектно-ориентированного и когнитивного подходов 28

1.2.1 Общие положения и обоснования 28

1.2.2 Анализ средств поддержки методов объектно-ориентированного подхода 29

1.2.3 Анализ средств поддержки методов когнитивного подхода 35

1.2.4 Сравнительный анализ рассмотренных инструментальны средств 40

1.3 Выводы по главе 43

Глава 2 Разработка методов моделирования визуальной интегрированной среды 45

2.1 Особенности и требования к визуальной интегрированной среде 45

2.1.1 Особенности методологии автоматизации интеллектуального труда 45

2.1.2 Разработка требований к ВИС 53

2.2 Разработка формального описания визуальной интегрированной среды 56

2.2.1 Разработка формального описания визуальной интегрированной среды на основе системного подхода 56

2.2.2 Разработка концептуальной модели процесса функционирования визуальной интегрированной среды 72

2.2.3 Уточнение формального описания визуальной интегрированной среды 82

2.3 Выводы по главе 86

Глава 3 Разработка методики управления развитием визуальной интегрированной среды . 87

3.1 Общие положения и обоснования 87

3.2 Место методики управления развитием визуальной интегрированной среды 88

3.3 Разработка классификатора модулей визуальной интегрированной среды 90

3.4 Разработка структуры элементов визуальной интегрированной среды 95

3.4.1 Описание основных конфигурационных элементов ВИС 95
3.4.2 Структурная диаграмма процесса развития ВИС 98
3.5 Разработка системы действий методики управления конфигурацией ВИС 100
3.6 Разработка спецификаций описания конфигурации ВИС 106
3.7 Выводы по главе 110
Глава 4 Описание реализации визуальной интегрированной среды 111
4.1 Описание средств реализации ВИС 111
4.1.1 Описание Microsoft Visual Studio 2015 111
4.1.2 Описание Qt 5.7.1 111
4.2 Описание программного комплекса «ИС-2» 112
4.3 Проектирование структур данных ПК «ИС-2» 114
4.4 Описание реализации функциональных модулей ВИС 122
4.4.1 Создание проекта 122
4.4.2 Описание редактора системы предметных действий 124
4.4.3 Формирование списка параметров 132
4.4.4 Формирование взаимосвязей параметров и предметных действий 134
4.4.5 Формирование основной концептуальной структуры 135
4.4.6 Документирование диаграмм и спецификаций начальной модели 138
4.4.7 Документирование спецификаций концептуальной модели 142
4.5 Применение ПК «ИС-2» и средств его развития 145
4.5.1 Применение ПК «ИС-2» в рамках инициативных проектов 145
4.5.2 Применение ПК «ИС-2» в учебном процессе 146
4.5.3 Оценка эффективности применения ИС-2 149
4.6 Выводы по главе 151
Заключение 152
Список сокращений 154
Список литературы 155
Приложение А Структура моделей АС на ранних этапах создания и их унифицированные элементы 167
Приложение Б Методика управления развитием ВИС 171
Приложение В Инструкция по созданию модуля для ПК «ИС-2» 196
Приложение Г Структуры данных ПК «ИС-2» 207
Приложение Д Свидетельства о государственной регистрации программы для ЭВМ 218
Приложение Е Справки об использовании результатов диссертационного исследования 223

ВВЕДЕНИЕ

Актуальность. Современные тенденции в сфере разработки программного обеспечения (ПО) предполагают погружение всего процесса создания в единую виртуальную среду, от анализа традиционного способа решения предметной задачи и формирования технического задания до реализации и сопровождения.

Анализ существующих подходов к автоматизации информационных и интеллектуальных задач и процессов, применяемых в них методов и инструментальных средств их поддержки, позволил выявить комплекс проблем, сдерживающих темпы внедрения и развития автоматизированных систем (АС):

- достаточно большое разнообразие специалистов в области создания автоматизированных систем;
- необходимость постоянного взаимодействия разработчиков и заказчиков АС;
- высокая степень итерационности процесса разработки АС;
- необходимость постоянного внесения изменений в процессе разработки и совершенствования АС;
- отсутствие стандартных методик формирования модельных представлений АС на ранних этапах её создания;
- необходимость обеспечения требуемого качества АС.

Выявленные проблемы определили необходимость формирования единой экосистемы разработки ПО на базе существующих методов и инструментальных средств. Основной проблемой при формировании подобных единых сред является сращивание графических моделей этапа проектирования и будущего программного кода.

Одной из рассмотренных методологий является методология автоматизации интеллектуального труда (МАИТ) разработанная в ФГБОУ ВО «МГТУ «СТАНКИН» на кафедре информационных технологий и вычислительны систем.

Основными преимуществами МАИТ являются: 1) наличие системы взаимосвязанных модельных представлений автоматизируемых задач – от начального до даталогического представления, которые содержательно увязаны на основе семантического представления этих задач; 2) наличие модельных представлений задач как инвариантных, так и ориентированных на программно-техническую среду и средства реализации АС; 3) выделение универсального модельного представления на каждом этапе создания, обеспечивающего порождение и интеграцию прикладных моделей предметных задач; 4) ориентация на промышленный способ создания АС.

Выполненная ранее разработка интегрированной среды поддержки создания учебно-

проектной САПР позволяла обеспечить взаимодействие уже созданных и создаваемых инструментальных средств поддержки отдельных функциональных процедур МАИТ на уровне полуавтоматизированной технологии, т.е. поддержки моделирования прикладных задач в форме спецификаций.

Для обеспечения взаимодействия создаваемых и совершенствуемых инструментальных средств на уровне автоматизированной технологии, т.е. моделирования прикладных задач в форме диаграмм и спецификаций, приобретает актуальное значение задача разработки новой визуальной интегрированной среды (ВИС).

Требования к процессу формирования модельных представлений прикладных задач в визуальной форме и требования к взаимодействию инструментальных средств поддержки отдельных функциональных процедур МАИТ определили необходимость разработки метода моделирования визуальной интегрированной среды.

Степень разработанности темы. Когнитивные технологии и прикладные концептуальные методы активно разрабатывались и применялись такими учеными как Никоноров С.П., Теслинов А.Г., Ашихмин В.С., Варламов О.О., Осипов В.Г., Гурьев А.Т., Шабаров В.Н. Методология автоматизации интеллектуального труда, обеспечивающая промышленный способ создания АС на базе когнитивного подхода, разработана доктором технических наук, профессором Волковой Галиной Дмитриевной. Дальнейшее развитие методы и методики методологии получили в работах Новоселовой О.В., Семячковой Е.Г., Курышева С. М., Щукина М.В., Сироты И.М., Бычковой Н.А., Протасовой С.В., Тюрбеевой Т.Б. и др. Исследования в области интеграции средств поддержки отдельных процедур МАИТ проводились Володиным Д.А.

Целью работы является повышение эффективности процесса создания автоматизированных систем за счет разработки метода моделирования визуальной интегрированной среды поддержки проектирования автоматизированных систем и средств поддержки управления её развитием.

Для достижения поставленной цели в работе решена ***научная задача***, включающая:

- исследование существующих методологий и средств их поддержки, применяемых при создании АС;
- разработку формального описания визуальной интегрированной среды и её элементов;
- разработку метода моделирования визуальной интегрированной среды;
- разработку методики управления развитием визуальной интегрированной среды;
- разработку прототипа визуальной интегрированной среды поддержки проектирования автоматизированных систем и средств её развития.

Объектом исследования является процесс автоматизации задач при создании автоматизированных систем.

Предметом исследования является процесс моделирования, разработки и организации инструментальных средств поддержки этого процесса.

Научная новизна:

- выявлены взаимосвязи между характеристиками содержательных функций создания автоматизированных систем, функциями визуального моделирования и функциями управления развитием ВИС, которые позволяют описать её состав и структуру;

- разработан метод моделирования визуальной интегрированной среды, особенностью которого является унификация конструктивных элементов составляющих её модулей.

- предложена модель визуальной интегрированной среды, реализующая перечисленные функции и позволяющая представить её в виде множества разнородных элементов файловой системы двух типов: универсальных (многократно используемые в различных составляющих ВИС) и специализированных;

Теоретическая значимость. Разработанный метод может быть использован для дальнейшего развития теоретических основ даталогического моделирования в рамках методологии автоматизации интеллектуального труда.

Практическая значимость. Разработана методика управления развитием визуальной интегрированной среды, обеспечивающая возможность планирования и реализации интегрированной среды из существующих и создаваемых программных модулей. Разработан прототип визуальной интегрированной среды поддержки создания АС и инструментальные средства поддержки процесса её развития.

Методы исследования. В процессе разработки теоретических положений диссертационной работы использованы аппараты теории множеств, математической логики, теории графов, метод концептуального моделирования (в соответствии с МАИТ).

Соответствие паспорту специальности. Указанная область исследования соответствует паспорту специальности 2.3.1 – Системный анализ, управление и обработка информации, а именно: п. 2 – «Формализация и постановка задач системного анализа, оптимизации, управления, принятия решений и обработки информации»; п. 8 – «Теоретико-множественный и теоретико-информационный анализ сложных систем»;

Положения, выносимые на защиту. На защиту выносятся:

1. Результаты анализа методов создания АС и средств их поддержки, на основе которых сформулированы требования к визуальной интегрированной среде, выявлена необходимость разработки метода её моделирования и средств поддержки управления её развитием.
2. Метод моделирования визуальной интегрированной среды, обеспечивающий описание среды на различных уровнях детализации с использованием унифицированных

элементов.

3. Методика управления развитием визуальной интегрированной среды, обеспечивающая поэтапное наращивание функциональных возможностей среды на основе промышленного подхода к разработке АС.
4. Разработанный прототип визуальной интегрированной среды, обеспечивающий процесс автоматизированного моделирования как сторонних автоматизированных систем, так и собственных будущих версий.

Апробация работы и степень достоверности результатов. Материалы работы докладывались и обсуждались на научно-практических конференциях: студенческой научно-практической конференции «Автоматизация и информационные технологии» (АИТ-2013), XV научной конференции «Математическое моделирование и информатика», VI всероссийской научно-практической конференции «Машиностроение – традиции и инновации (МТИ-2013)», Шестнадцатой международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности и экономике», Восемнадцатой международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности и экономике», X International scientific and practical conference, «Fundamental and applied science», студенческой научно-практической конференции «Автоматизация и информационные технологии (АИТ-2015)», XVII научной конференции «Математическое моделирование и информатика», IV международной научной конференции «Моделирование нелинейных процессов и систем (MNPS-2019), V международной научной конференции «Моделирование нелинейных процессов и систем (MNPS-2020)».

Методика управления развитием ВИС, средства её поддержки, а также прототип ВИС применены в рамках инициативных проектов при поддержке РФФИ № 13-07-00752 «Создание интегрированной среды поддержки проектирования и реализации прикладных автоматизированных систем на основе методологии автоматизации интеллектуального труда», № 17-29-07056 «Разработка моделей и методов представления и обработки проблемно-ориентированных знаний, извлекаемых из научно-технических текстов и конструкторско-технологической документации» и № 17-29-07057 «Разработка методов реструктуризации и интеграции для семантических и синтаксических представлений при создании систем автоматизации процессов проектирования и управления». Также они были применены в учебном процессе кафедры информационных технологий и вычислительных систем МГТУ «СТАНКИН» при подготовке бакалавров, магистров и аспирантов по направлениям 09.03.01, 09.04.01, 09.06.01 «Информатика и вычислительная техника» в качестве учебно-проектного средства поддержки дисциплин «Базы данных», «Концептуальное моделирование проектных задач», «Проектирование прикладных

автоматизированных систем», «Анализ деятельности проектно-конструкторских организаций», «Технологии разработки программного обеспечения» и при подготовке выпускных квалификационных и научно-квалификационных работ, а также в деятельности компании ООО «Специальная интеграция».

Структура и объём работы. Диссертация состоит из введения, 4 глав, заключения, списка литературы (104 наименования), перечня сокращений и приложений. Работа содержит 224 страниц сквозной нумерации, включая 113 рисунков, 20 таблиц и 58 страниц приложений.

ГЛАВА 1
АНАЛИТИЧЕСКИЙ СУЩЕСТВУЮЩИХ ПОДХОДОВ, МЕТОДОВ И СРЕДСТВ СОЗДАНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

1.1 Анализ существующих подходов и методов, применяемых при создании АС

1.1.1 Общие положения и обоснования

В отличие от материального производства, где происходит формирование изделий определенного функционального назначения, производство программного обеспечения опирается на управление информационными потоками. Физические эффекты, лежащие в основе принципов работы любой техники, уступают место законам преобразования знаний, наполняющих вычислительную среду. Автоматизация средствами ЭВМ предполагает извлечение исходной информации из головы специалистов или из технической литературы и её перенос в память компьютера. При этом, насколько не был бы сложен, многомерен, многослоен первоисточник, ему предстоит стать формализованной последовательностью двоичных знаков электронной памяти. Выявление закономерностей такого превращения информации – до сих пор не изученная научная задача.

В пятидесятых и шестидесятых годах XX века, на ранних этапах автоматизации интеллектуальных задач, объектом были в основном трудно вычислимые, но хорошо формализованные задачи физики и математики. Однако, со временем возрастающий объем информации об автоматизируемых задачах, в котором отражена организация процессов реального мира, стал требовать предварительной обработки. Приобрела актуальность задача разработки новых методов и методологий, наглядно представляющих структуру предметной области и её процессы для дальнейшего перевода их в вычислительную среду.

Методология, с одной стороны, определяется как множество методов, описывающих структуру модели, позволяющей отразить рассматриваемую область, с другой стороны, определяется как набор методик и приемов моделирования. Методика описывает процесс моделирования и правила фиксации результата в виде нотации. В качестве нотации могут использоваться графические диаграммы, описывающие различные структуры модели, или иные способы фиксации информации. Инструментальные средства, поддерживающие большинство

методов, используются разработчиками автоматизированных систем на всех этапах жизненного цикла. [1].

Проведенные исследования позволили дифференцировать рассмотренные методологии по подходам. Каждый подход характеризуется определенной технологической парадигмой, т.е. совокупностью способов представления знаний о реальном мире и способов организации знаний в вычислительной среде (хранение, доступ, манипулирование). [2]

В рамках выделенных подходов были рассмотрены следующие методологии:

– в рамках структурного подхода:
  - метод функционального моделирования SADT;
  - методы комплексного моделирования бизнес-процессов семейства IDEF0, IDEF1X, IDEF3;
  - методология моделирования информационных систем ARIS;
– в рамках объектно-ориентированного подхода:
  - язык UML, как комплексное средство моделирования различных составляющих автоматизированных систем на разных этапах их создания;
  - метод IDEF-4;
  - методы проектирования онтологий, в частности метод IDEF5;
– в рамках когнитивного подхода была рассмотрена методология автоматизации интеллектуального труда, содержащая в себе методы представления автоматизированных систем на различных этапах их создания в виде формализованных взаимоувязанных модельных представлений.

1.1.2 Характеристика структурного подхода

Основной принцип структурного подхода заключается в декомпозиции, или разделении сложной системы на отдельные части, автоматизирующие отдельные функции. При этом система сохраняет целостное представление, а её отдельные элементы увязаны в комплексе, который служит для достижения поставленной цели системы.

Разделение системы на части выполняется рекурсивно по принципу функциональной декомпозиции. Сложные действия разбиваются на более простые до тех пор, пока не будет достигнут требуемый уровень детализации. Как правило, на самом детальном уровне выполняются лишь элементарные действия – математические операции, операции обращения к источникам информации и статистические действия.

Увеличение размера системы влечет за собой сложности в обеспечении целостности её составляющих, вследствие чего возникают проблемы информационной связности отдельных компонентов. [3]

Исторически перовой методологий структурного подхода является SADT (structured analysis and design technique), разработанная в конце 1960х годов Дугласом Тейлором Россом в рамках программы MIT по созданию средств автоматизации деятельности программистов. Методология структурного анализа и проектирования позволяет формировать комплекс функциональных моделей, последовательно детализирующий рассматриваемую область. Модель SADT отображает функциональную структуру объекта, т.е. производимые им действия, и совокупность материальных, информационных, финансовых и энергетических потоков между этими действиями. Функциональное моделирование является важнейшим элементом анализа, который выполняется на начальном этапе проектирования любой автоматизированной системы. [4]

Основу методологии SATD составляет графическое представление блочного моделирования. На SATD диаграммах функции представлены блоками, а интерфейсы входа/выхода из нее отображаются в виде стрелок, входящих или выходящих из блоков соответственно. Взаимодействие блоков друг с другом фиксируется при помощи стрелок, накладывающих на них ограничения, которые определяют способы управления функциями, их входные и выходные данные. Реализацией методологии SADT является метод функционального моделирования IDEF0. [3]

Методология IDEF, включающая в себя комплекс различных методов моделирования, создавалась в рамках предложенной ВВС США программы ICAM (Integrated Computer-Aided Manufacturing) – программы компьютеризации производства. Основная цель разработки стека IDEF – обеспечение информационного взаимодействия между всеми участниками программы. Эти методы моделирования нашли множество применений в различных областях человеческой деятельности, показав себя эффективными при анализе, проектировании и документировании бизнес-процессов. Изначально стек включал: IDEF0 – метод функционального моделирования и IDEF1 – метод информационной структуры. В ходе развития информационных технологий комплекс дополнился различными новыми методологиями, выйдя за рамки структурного подхода.

Результатом формирования **IDEF0** модели является набор графических диаграмм, ссылочно-связанных между собой. Первой диаграммой является контекстная, показывающая отношения объекта моделирования – рассматриваемого процесса с внешнем миром. Дальнейшие диаграммы формируются по принципу функциональной декомпозиции объекта – единственный процесс с контекстной диаграммы разбивается на подпроцессы на следующей диаграмме в модели, которые также могут быть декомпозированы. Процесс разложения продолжается до тех пор, пока не будет достигнута степень детализации, определяемая задачами разработки IDEF0 модели. [5]

Диаграммы содержат [6, 7]:

1) Блоки, представляющие основные функциональные элементы диаграмм – функции, процессы, действия, операции, и т.д. Блоки содержат наименование, номер в рамках диаграммы и номер в рамках модели. Каждый блок может иметь дочернюю диаграмму, т.е. диаграмму, на которой он раскрыт более подробно. (рис 1.1).
2) Стрелки, представляющие основные элементы взаимодействия между блоками. В зависимости от того, с какой стороны подходит стрелка, определяется её роль для блока. В зависимости от предметной области стрелки могут обозначать потоки данных, денежные потоки, энергетические, материальные и т.д.

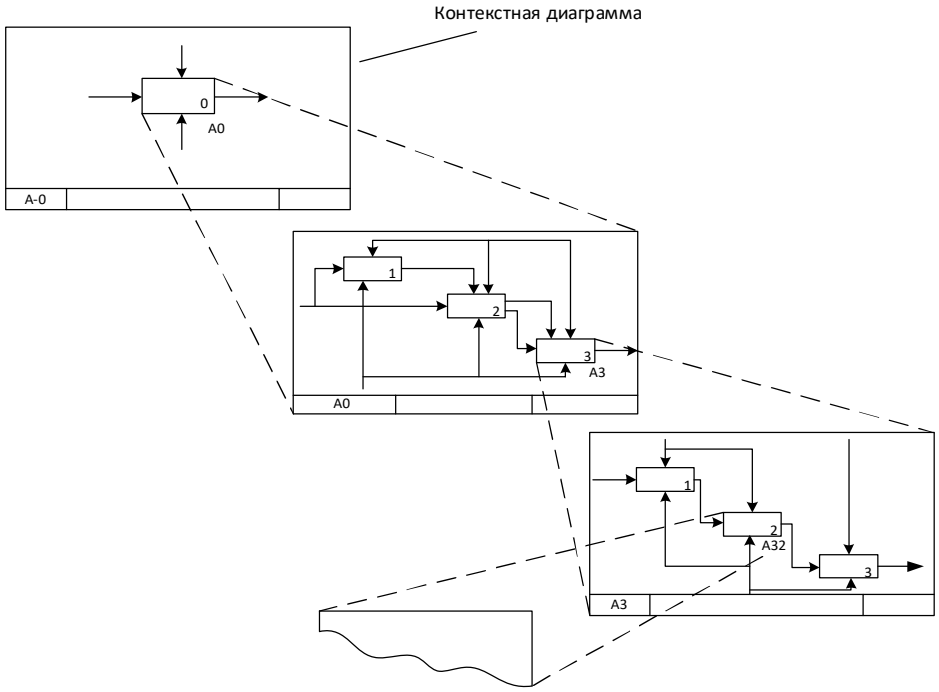


Рис. 1.1 Организация IDEF0 диаграмм в модели

IDEF3 является методом моделирования технологических процессов. Был разработан в середине 1970х годов в качестве дополнения к IDEF0. Метод позволяет строить модели двух типов: PFDD (Process Flow Description Diagrams) – потоковое описание процессов, и OSTN (Object State Transition Network) – описание переходов состояний. В обоих случаях диаграмма позволяет описывать течение процесса во времени с учетом распараллеливания. [8]

Модели фиксируются в виде графических диаграмм. В отличии от IDEF0 стрелки на этой диаграмме отображают временную зависимость функциональных блоков друг от друга, более точная характеристика зависимости влияет на графический вид стрелки. Помимо функциональных блоков на диаграмме размещаются блоки-операнды, которые, используя основные логические операции, позволяют управлять процессом, т.е. организовывать циклы, условия, ветвление и слияние последовательности действий.

Функциональные блоки могут быть декомпозированы на более простые дочерние IDEF3 диаграммы. Проверку диаграмм осуществляет эксперт в предметной области. Анализ модели

выполняется вручную, так как диаграммы имеют образную природу.

Рассмотренные методы позволяют фиксировать только функциональное описание моделируемой предметной области на начальных этапах создания АС, не уделяя внимания содержательной части. В рамках структурного подхода данные рассматриваются в виде структур и их связей. Наиболее полная формальная система, реализующая эту парадигму — реляционная модель Кодда, а её информационное воплощение – модель «сущность-связь», или ER-модель, которая сегодня занимает доминирующую позицию при разработке баз данных. ER-модель была разработана Ченом в середине 70-х годов XX века. Основной принцип – моделирование предметной области в виде графических диаграмм, включающих необходимые для описания сущности и связи между ними. [9, 10]

IDEF1X – методология разработки информационной модели, основанная на методологии Чена, является современной версией более старой методологии IDEF1. ER – диаграммы, из которых формируется модель IDEF1X, состоят из блоков, отображающих сущности физического или информационного мира, и связей между ними. Модели разрабатываются на начальных этапах создания АС и служат средством концептуализации, т.е. представления знаний о предметной области. Сущности агрегируют в себе множество атрибутов. Атрибуты, по которым происходит связь, образуют пару первичного и внешнего ключей. Таким образом, формируемые модели описывают структуру информации из сущностей, соответствующих третьей нормальной форме в реляционной модели данных. [11]

Наиболее распространенные методы структурного подхода методологии IDEF (IDEF0, IDEF1X, IDEF3) описывают модели различных компонентов автоматизированных систем. Позволяют формировать модели как функциональные, так и информационные. Методы не обеспечивают возможность интеграции различных однотипных моделей, а также увязку разнородных в рамках проекта, что не позволяет сформировать целостного представления на различных уровнях детализации при моделировании различных компонентов АС. Обеспечение целостности контролируется экспертом-аналитиком на основании его квалификации без какой-либо методической поддержки со стороны методов и большинства инструментальных средств поддержки.

Другим представителем структурного подхода является методология ARIS (Architecture of Integrated Information Systems), концептуально разработанная в середине 2000 годов Августом-Вильгельмом Шеером, немецким специалистом по менеджменту в IT. ARIS предоставляет целостный инструментарий средств анализа и моделирования бизнес-процессов организации, а также автоматизированных систем. По ходу развития методология вобрала в себя особенности других методов моделирования, некоторые из которых выходят за рамки структурного подхода, совместив разные взгляды на исследуемую систему. ARIS позволяет разрабатывать

взаимоувязанные модели в различных нотациях, что позволяет использовать методологию специалистами различных профилей и настраивать её на работу под конкретную специфическую задачу. Отличительной особенностью этой методологии является способность сопряжения моделей разного типа, описывающих различные аспекты предметной области. Также методология имеет средства поддержки, объединённые в виде одноимённой комплексной среды. [12]

Методология ARIS поддерживает четыре типа моделей, взаимосвязь которых приведена на рис. 1.2, отражающих различные аспекты исследуемой предметной области: [13, 14]

- Организационная модель – иерархическая модель субъектов предметной области. Позволяет построить многоуровневую модель организации, в которой функционируют моделируемые бизнес-процессы.
- Модель процессов – функциональная модель, описывающая процессы, необходимые для достижения поставленных целей, с учетом их иерархии;
- Модель данных – информационная модель, отражающая структуру информации, необходимой для выполнения процессов системы, а также структуру выходной информации;
- Модель входов-выходов – модель управления, приставляющая комплексный взгляд на реализацию бизнес-процессов в рамках всей системы.

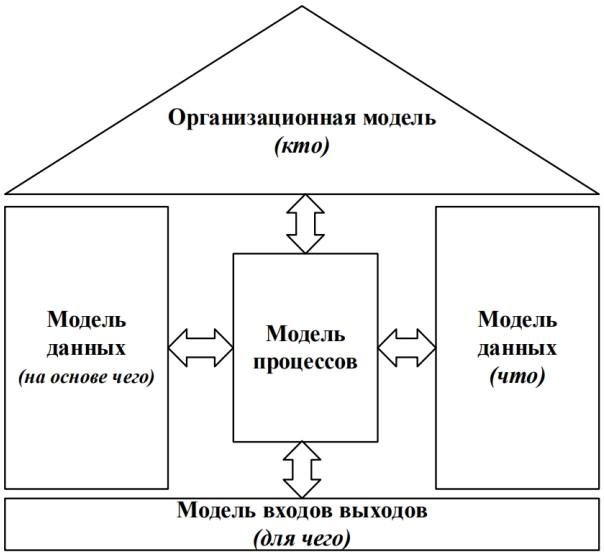


Рис. 1.2 Взаимосвязи моделей в методологии ARIS

Методология ARIS предлагает целостный взгляд на моделируемую область, позволяет отразить её с применением различных нотаций, объединенных в рамках единого системного подхода [15]:

- диаграммы eEPC (Extended Event driven Process Chain - событийная цепочка процесса) (рис 1.3);
- модели ERM – (Entity Relationship Model – модель «сущность-связь»);
- язык UML (Unified Modeling Language – универсальный язык моделирования);

- методики OMT (Object Modeling Technique – методика объектно-ориентированного моделирования);
- методика ABC (Activity Based Costing – пооперационный расчет стоимости процессов);
- методики BSC (Balanced Scorecard – система сбалансированных показателей).

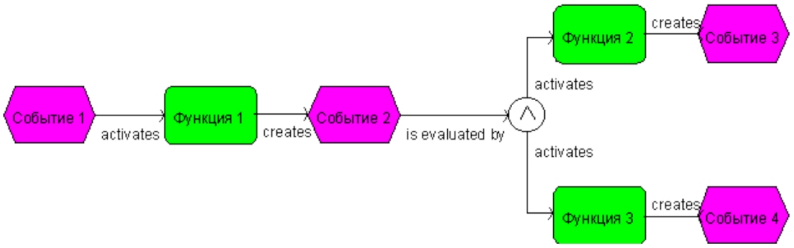


Рис. 1.3 Пример нотации eEPC

Достоинством методологии является детальная проработка каждого аспекта модели, независимо друг от друга. Впоследствии можно разработать интегральную модель, описывающую систему в целом и учитывающую все связи между различными ей компонентами. ARIS не регламентирует жесткий процесс моделирования, что значит – все составляющие модели могут разрабатываться в любом порядке, независимо друг от друга, однако существуют устоявшиеся практики моделирования. Процесс анализа и проектирования можно начинать с любой из них в зависимости от конкретных условий и целей, преследуемых при разработке.

ARIS имеет два основных преимущества:

- позволяет создавать интегрированные модели различных компонентов автоматизированных систем, определять их набор в зависимости от цели моделирования;
- позволяет производить взаимоувязанное моделирование как самой деятельности, так и управления этой деятельностью в рамках единой среды.

К недостаткам рассматриваемой методологии можно отнести:

- необходимость обучения работе с методологией и средствами её поддержки;
- необходимость четкой проработки регламента работы с ARIS;
- избыточность методологии для анализа;
- невозможность генерации каких-либо программных кодов или схем баз данных при проектировании;

С учетом рассмотренных методологий, являющихся самыми распространёнными представителями структурного подхода, можно сделать вывод, что его существенными преимуществами являются:

- Организация информации в моделях методологий структурного подхода отображает реальную структуру предметной области.
- Последовательный анализ системы «от сложного к простому» позволяет описывать её с

различной степенью детализации, что облегчает процесс её обследования при дальнейшей автоматизации.

К недостаткам структурного подхода можно отнести:

- Отсутствие увязки между моделями различных составляющих моделируемой области.
- Необходимость в некоторых случаях производить вынужденную декомпозицию на отдельные диаграммы, чтобы не нагружать диаграмму большим количеством элементов.
- Сложные модели, содержащие в себе большое количество листов, трудно воспринимаемы человеком.

1.1.3 Характеристика объектно-ориентированного подхода

Суть объектно-ориентированного подхода (ООП) – представление предметной области в виде множества взаимодействующих объектов. Совокупность однотипных объектов, т.е. имеющих одинаковую структуру и поведение, называется классом. ООП позволяет моделировать систему в разных аспектах: как совокупность экземпляров классов и множества статических связей между ними, для описания структуры создаваемой АС; как множество функциональных связей между объектами, для описания поведенческой составляющей АС. Объектно-ориентированный подход является самым распространённым на сегодняшний день, так как большинство языков программирования были разработаны под объектно-ориентированную парадигму.

UML, унифицированный язык моделирования, в настоящее время развиваемый консорциумом OMG, представляет собой средство визуального моделирования объектно-ориентированных систем. Наиболее актуально визуальное моделирование при разработке средних и крупных информационных систем, когда для совместной работы над проектом объединяются большие группы разноплановых специалистов, и при этом необходимо организовать между ними эффективную передачу результатов деятельности. Диаграммы, формируемые в рамках этого визуального языка, могут описывать систему с разных точек зрения и на разных увязанных уровнях абстракции.

MOF (Meta-Object Facility, метаобъектное средство) – стандарт разработки метамоделей, разработанный OMG, для описания UML, с использованием его графической нотации. MOF организован как четырехслойная структура, верхний уровень которой определяет метамодель, выступающая по отношению к UML, как модель более высокого уровня абстракции, определяющая

его состав. [16]

Создание средств метамоделирования позволило: однозначно определить элементы модели UML; разработать механизмы развития и расширения UML; определить объектно-ориентированные модели UML как формально-языковое представление АС; создать базис развития средств метамоделирования для создания языков описания объектно-ориентированных систем на базе UML.

На рис. 1.4 приведен фрагмент метамодели, описывающей такие базовые для UML понятия как «Класс» (“Class”), «Наследование» (“Generalization”), «Тип данных» (“Type”), «Свойство» (“Property”), «Операция» (“Operation”), и т.д. [17]

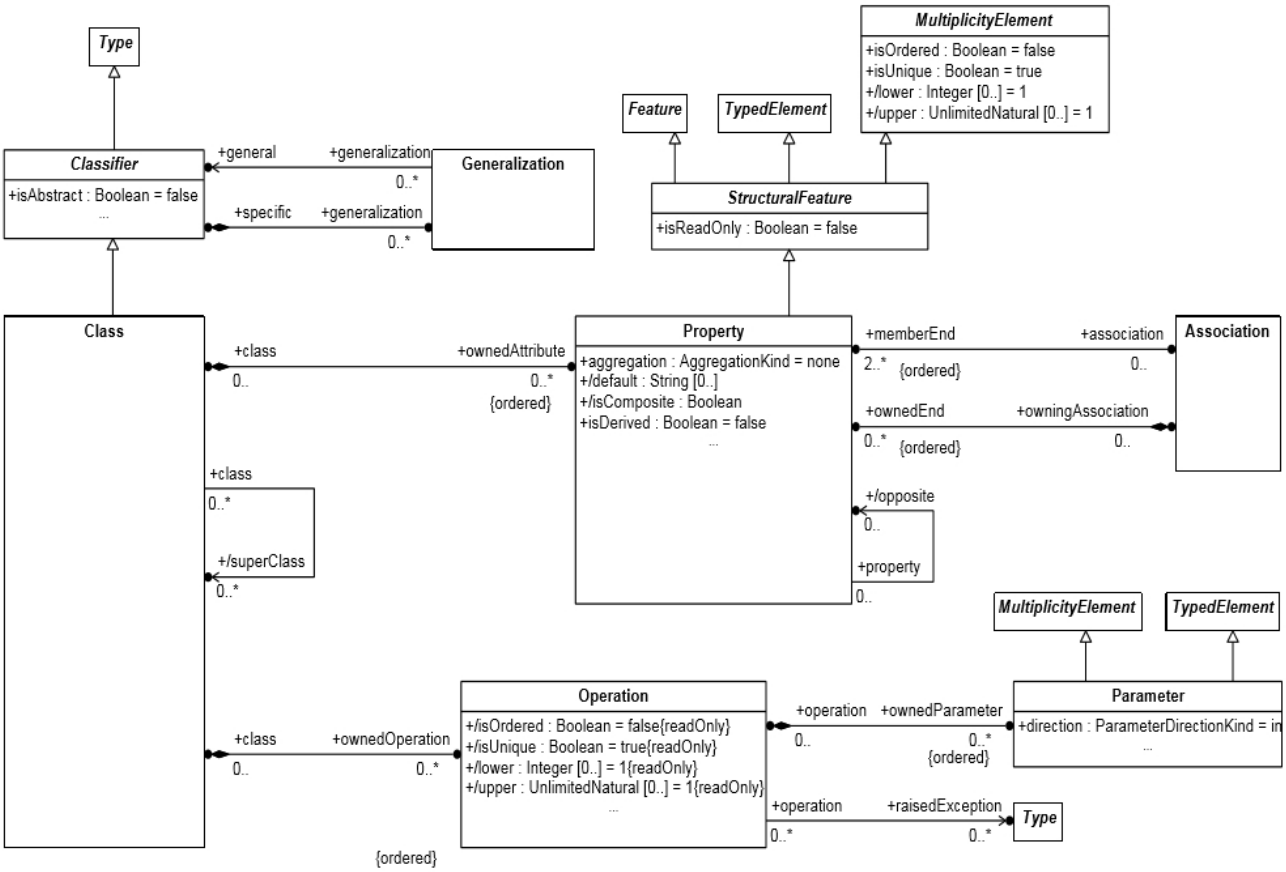


Рис. 1.4 Фрагмент MOF метамодели UML

В UML 2.5 определены 14 типов диаграмм, разделенных на три подгруппы. Шесть диаграмм позволяют описать статические аспекты архитектуры (диаграммы классов, объектов и компонент, композитные структурные диаграммы, диаграммы пакетов и развертывания), три - основные типы поведения (диаграммы сценариев использования, деятельностей и состояний) и четыре представляют различные аспекты взаимодействия (обычно рассматриваются, как подгруппа диаграмм поведения, к которой относятся диаграммы последовательностей, коммуникации объектов, временные диаграммы и обзорные диаграммы взаимодействия) (рис 1.5).

Основные диаграммы, позволяющие наиболее полно описать разрабатываемую систему:

- Диаграмма прецедентов – позволяет организовать и систематизировать функциональные требования к создаваемой системе. Фактически, диаграмма отображает будущий программный продукт с точки зрения функциональности глазами её потенциальных пользователей.
- Диаграмма классов – отображает структуру системы в виде совокупности классов и связей между ними. Диаграмма является одним из самых распространённых средств моделирования объектно-ориентированного подхода за счет своей универсальности. Для классов фиксируются их атрибуты, представляющие свойства объекта, представленного классом, и операции – действия, переводящие класс из одного состояния в другое. Под состоянием класса понимается совокупность значений его атрибутов в один момент времени.

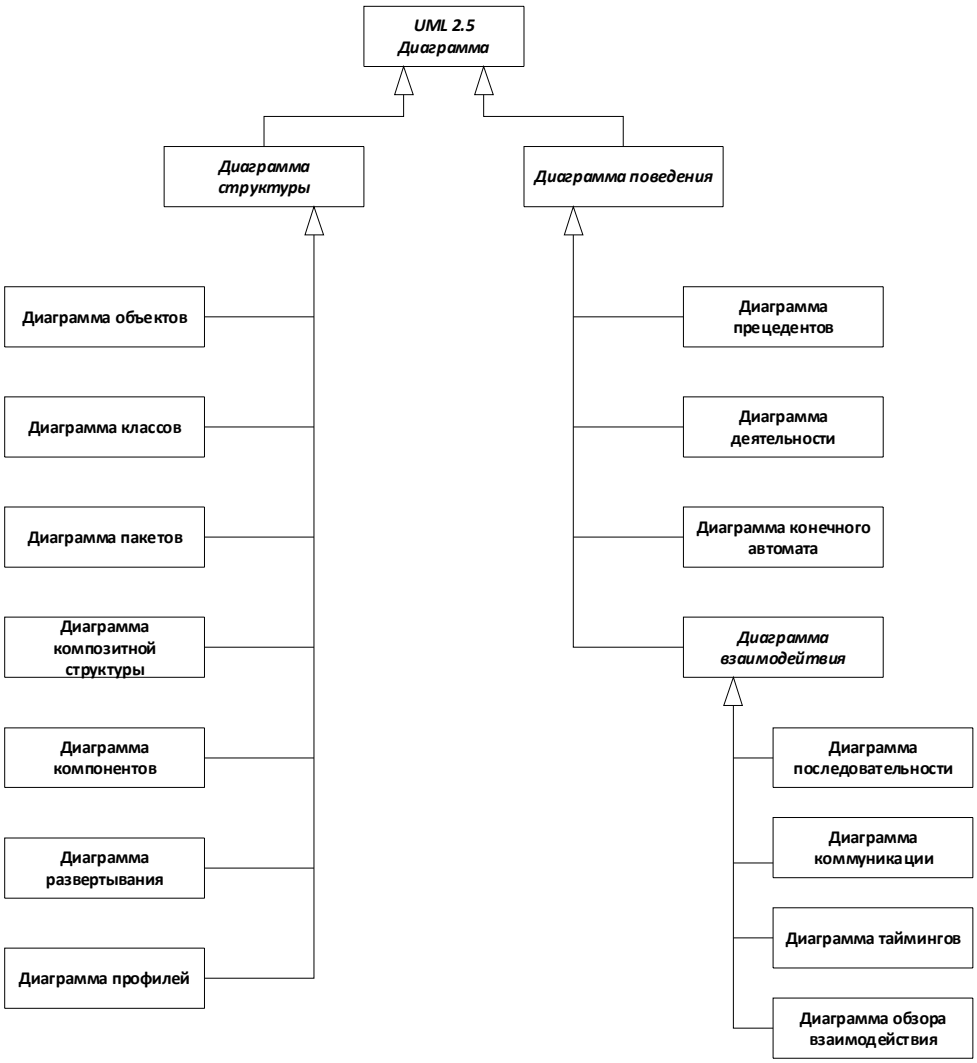


Рис. 1.5 Схема диаграмм UML, представленная диаграммой классов

- Диаграмма последовательностей – описывает систему с точки зрения фактической функциональности. Задача разработки диаграммы – отобразить взаимодействия объектов классов в виде сигналов, передаваемых между объектами во временном потоке.

К преимуществам UML можно отнести:

- структура моделей автоматизированных систем по содержанию и организации информации аналогична будущему программному коду, так как UML создавался как средство моделирования разрабатываемых программных продуктов;
- возможность моделирования систем на различных уровнях абстракции;
- наглядность представления результатов моделирования;
- большое количество моделей, позволяющих отобразить различные аспекты моделируемой системы;
- наличие метамодели и формальное описание языка.

Однако унифицированный язык моделирования обладает слишком широкой областью применения, из-за чего может потребовать дополнительной настройки языка под свои нужды (ввод нестандартных конструкций и элементов). Также при моделировании сложной системы обеспечение целостности модели, состоящей из различных диаграмм, является затруднительным процессом.

Развитие объектно-ориентированного подхода повлияло на стек IDEF-методологий. Как следствие, он дополнился методами описания объектно-ориентированных систем и онтологий IDEF4 и IDEF5 соответственно.

IDEF4 – метод компонентного объектно-ориентированного проектирования клиент-серверных систем. Метод разработан для сокращения времени перехода от моделей предметной области разрабатываемой системы и моделей анализа требований к проектированию и последующей реализации программного продукта. При разработке требуется достаточно подробное наполнение объектов модели для последующей автоматизированной генерации программного кода. [18]

Метод объектно-ориентированного проектирования IDEF4 включает в себя:

- модели различных уровней проектирования: системный уровень, уровень приложения, низкоуровневое проектирование;
- артефакты (элементы моделей) различной принадлежности: приложение, программное обеспечение, переход (между моделями);
- модели различного типа: статические, динамические, поведенческие.

К недостаткам метода IDEF4 можно отнести необходимость принятия решения о будущей реализации создаваемой автоматизированной системы уже на ранних этапах создания. Это требует преждевременное определение как среды реализации, так и используемых в процессе реализации технологий. Также к недостаткам метода можно отнести слабую инструментальную поддержку.

Развитие принципов формирования описаний систем знаний, заложенных в объектно-ориентированном подходе, получило развитие в методах формировании **онтологий**, зародившихся в 1970х годах в ходе исследований в области искусственного интеллекта. Онтология – формальное

описание системы знаний в рассматриваемой области предметной задачи. Разработка онтологий помогает координировать деятельность различных специалистов при разработке АС путем формирования стандартных словарей, систематизирующих и стандартизирующих понятия системной области и их взаимосвязи. Элементом онтологии является понятие. В информатике онтологии используются для фиксации понятий и связанных с ними объектов в определенной предметной области, а также связанных с ними отношений и значений. Понятия представлены классами. Класс может иметь уточнения, определяемые подклассами. Атрибуты класса в онтологии представлены слотами, а ограничения на множество значений слотов – фацетами. Онтологическая база знаний наполняется множеством экземпляров классов с конкретными значениями слотов. [19]

Однако разработка онтологий отличается от проектирования классов и отношений в ООП: объектно-ориентированный подход сосредотачивается на поведенческих аспектах класса – на его методах. Разработчик формирует объектно-ориентированную структуру создаваемой системы, основываясь на операторных свойствах класса. При разработке онтологии основываются на структурных свойствах класса, а не на его поведенческих особенностях. В результате структуры классов и их отношения в онтологии отличаются от схемы классов в ООП. [19]

IDEF5 – метод формирования онтологий, был разработан для построения онтологий с точки зрения понимания специалистом конкретной области. [20] Онтология строится путем фиксации содержания определенных утверждений о реальных объектах, их свойствах и взаимосвязях и представления этого содержания как в графической, так и в текстовой форме. Модель IDEF5 состоит из следующих компонентов: [21]

- графический язык – графическая нотация поддержки визуального представления онтологии;
- текстовый язык – язык структурированного текста для описания элементов графического представления онтологии;
- методика, которая обеспечивает руководящие принципы для эффективного формирования онтологии.

Графический язык SL (Schematic Language) предлагает графическую нотацию, предназначенную для фиксации основных данных в форме онтологической визуальной информации (рис. 1.6). [22] Модели строятся экспертами рассматриваемой области. Язык позволяет формировать первоначальную версию онтологической модели, постепенно наполняя её новыми объектами. Элементами языка являются графические блоки, обозначающие классы, их экземпляры, взаимодействия между экземплярами класса, процессы, в которых участвуют экземпляры классов. На каждый графический элемент можно составить документацию с применением текстового языка EL (Elaboration Language). SL позволяет формировать диаграммы различного функционального назначения (классификации, диаграммы состояний, композиции, реляционные схемы данных) с

целью – получить визуально наглядный и понятный аналитику результат.

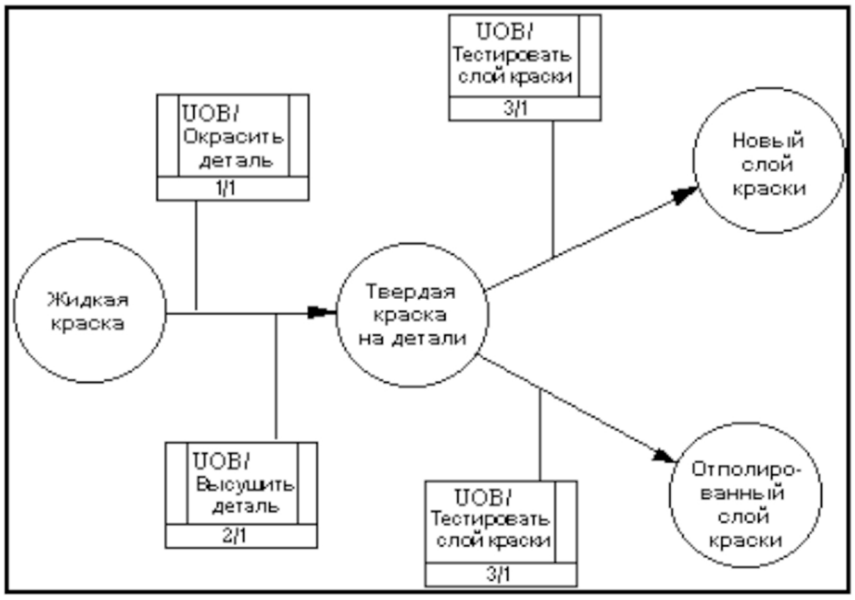


Рис. 1.6 Пример диаграммы состояний в нотации SL (IDEF5)

Во многих дисциплинах сейчас разрабатываются стандартные онтологии. В области медицины создана большая семантическая база – UMLS (Unified Medical Language System), содержащая три онтологические модели: SNOMED – модель знаний в области здравоохранения, LONIC – модель лабораторных данных, RxName – база лекарств. [23] [24] В области электронной коммерции разработана обширная таксономия товаров и услуг ООН – UNSPSC (United Nations Standard Products and Services Code).

Исходя из анализа методологий объектно-ориентированного подхода можно сделать вывод, что все они нацелены на предоставление образного результата моделирования в виде диаграмм. Основным представлением в методологиях этого подхода является информационное представление, основанное на ER-модели, при этом, представления других аспектов моделируемой задачи являются подчинёнными.

1.1.4 Характеристика когнитивного подхода

Самый молодой из рассматриваемых подходов – когнитивный, позволяет формировать многоуровневое преставление предметной области на основе законов, сформулированных в ходе исследований в сфере искусственного интеллекта. Особенностью когнитивного подхода является организация знаний в той форме, в которой они находятся в голове у специалиста. Многослойная структура формируется на элементах трех типов – контекстуальных, структурных и монадических. Контекстуальные элементы раскладываются на структурные, которые, в свою очередь, уточняются

до монадических. Применение закона цикличности позволяет рассматривать монадические элементы как контекстуальные, но более низкого уровня сложности. Таким образом появляется иерархическая, ориентированная на конкретную проблему структура функциональных центров. [25, 26]

Методология автоматизации интеллектуального труда (МАИТ) является одной из методологий когнитивного подхода к разработке АС. Разработана на кафедре информационных технологий и вычислительных систем ФГБОУ ВО МГТУ «СТАНКИН» под руководством член-корреспондента РАН Соломенцева Юрия Михайловича, доктором технических наук, профессором Волковой Галиной Дмитриевной. Свое развитие МАИТ получила в диссертационных работах Новоселовой О.В., Семячковой Е.Г., Курышева С. М., Щукина М.В., Сироты И.М., Володина Д.А., Бычковой Н.А., Протасовой С.В., Тюрбеевой Т.Б., и др. Цель разработки МАИТ – повышение эффективности автоматизации любой интеллектуальной деятельности. «Суть МАИТ заключается в получении последовательности отображений прикладных задач в виде формализованных моделей — инфологической (ИЛМ) и даталогической (ДЛМ) — на основе первоначально формируемой концептуальной модели (КМ), отражающей систему знаний предметной задачи, на трех уровнях абстрагирования». [26] Основой для концептуальной модели является начальная модель (НМ), формируемая на этапе предпроектного обследования. Структура методологии приведена на рисунке 1.7. Вертикальные стрелки описывают взаимосвязь моделей по уровням абстрагирования, горизонтальные стрелки определяют взаимосвязь моделей в рамках одного уровня абстрагирования. Диагональные стрелки показывают, что для планирования эффективной реализации конкретной технологической среды, необходимо знать как устроена любая система знаний.

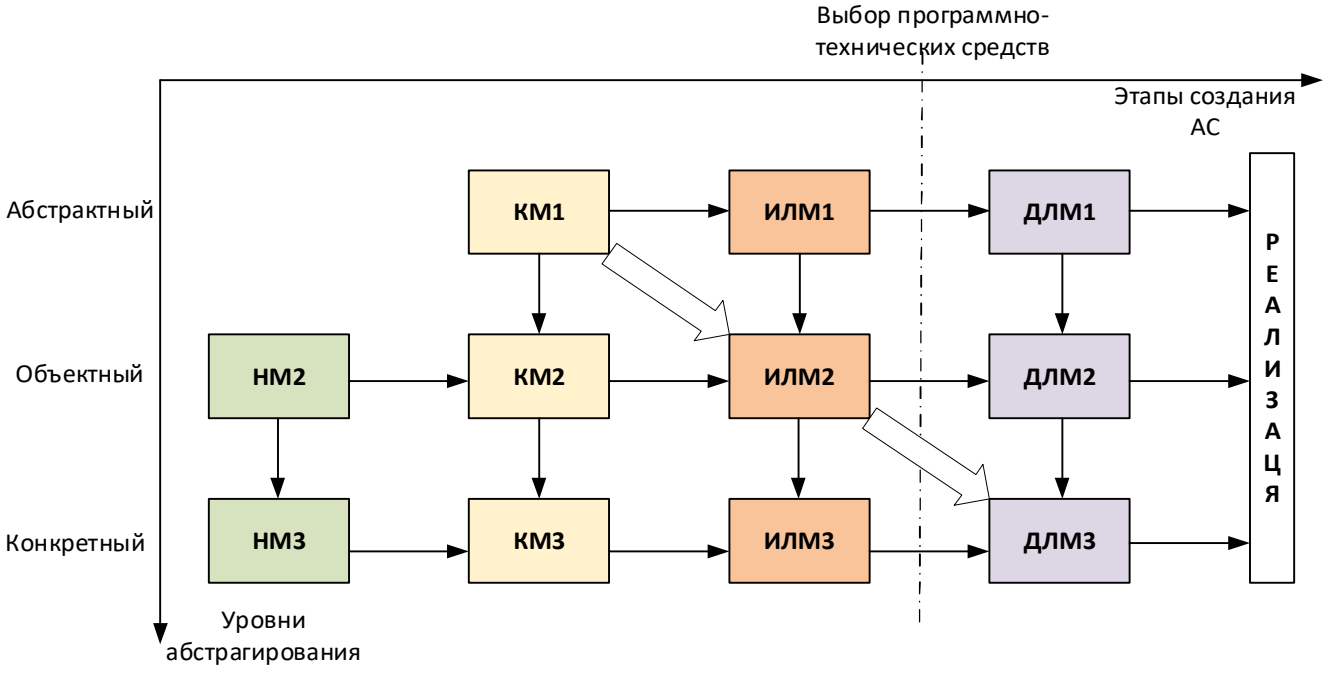


Рис. 1.7 Структура МАИТ

Для каждого этапа методологии определены два модельных представления: универсальное и предметное. Универсальное представление определяет общее строение моделей на трех уровнях абстрагирования (абстрактный, объектный, конкретный). Предметное представление определяет структуры моделей на двух уровнях абстрагирования (объектный, конкретный).

МАИТ позволяет формировать модели на каждом этапе создания не только отдельных предметных задач (ПЗ), но и их комплексов (КПЗ) в единообразной форме. Для комплексов формируются обобщенные модели: первичная и полная, как объединение первичной обобщенной модели и множества локальных моделей отдельных предметных задач. Процесс моделирования на каждом этапе строится из набора унифицированных процедур, обеспечивающих формирование и дальнейшую обработку моделей.

В процессе разработки начальной модели выполняется описание традиционного процесса решения задачи, заключающееся в выявлении множества параметров объектов предметной области и множества функциональных зависимостей на параметрах (предметных действий). Выполняется первичное обследование предметной области и формирование модели, отражающей автоматизирующую задачу.

При разработке концептуальной модели выполняется формирование основных и производных понятийных структур, а также систем динамических ограничений на них. При этом концептуальная модель включает:

- основную понятийную структуру на множество предметных категорий и их связях на различных уровнях семантической сложности;
- производную структуру на схемах предметных категорий;
- системы ограничений на взаимосвязи и состояния элементов основной и произвольной понятийных структур, выраженных в предметных зависимостях;
- модель в целом, отражающую увязку понятийной структуры и систем динамических ограничений.

Под инфологической моделью автоматизированной системы понимается такое формализованное представление, которое отражает организацию информации, механизмы доступа к информации и логику её обработки для решения прикладной задачи, не зависящее от среды и средств реализации. Организация информации фиксируется в виде основной и производной инфологических структур – иерархических многоуровневых конструкций из именованных элементов типа «информационный модуль», «информационная сущность» и «информационный атрибут», а также их схем. В виде отдельных структур оформляются система манипуляций и система доступа к информации. Все компоненты инфологической модели увязаны между собой и представляют в совокупности целостное описание автоматизируемой задачи.

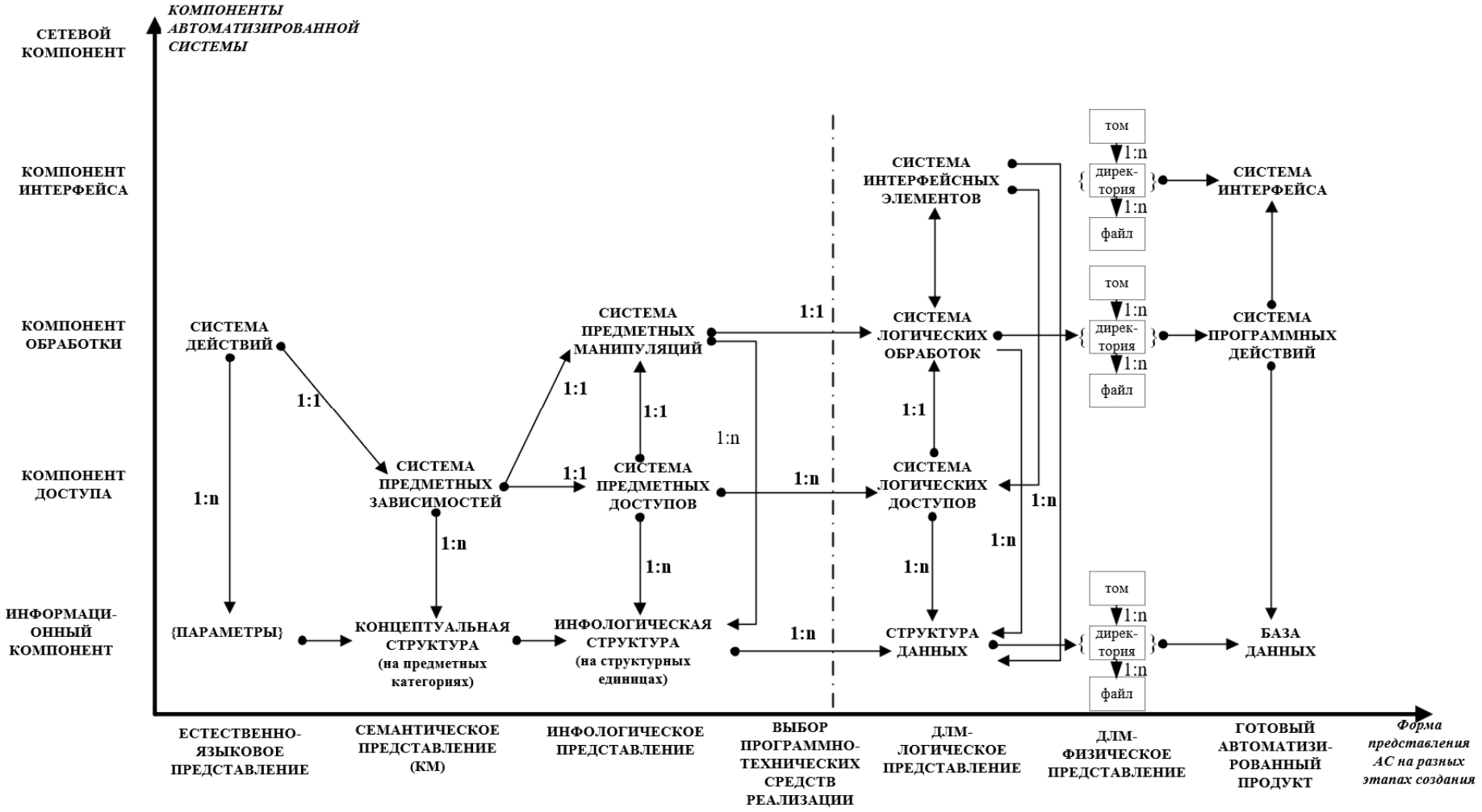


Рис. 1.8 Состав моделей АС создаваемой по МАИТ

Под даталогической моделью АС понимается такое формализованное представление, которое отображает организацию, процедуры обработки и доступа к данным в конкретной программно-технической среде и с помощью конкретных программно-технических средств. Состав даталогической модели логического уровня: база данных на структурных единицах, множество динамических и функциональных отношений на единицах, множество интерфейсных элементов (представлений СЕ) и их связей. Даталогическая модель физического уровня фиксирует состав разрабатываемой системы с точки зрения элементов файловой системы: томов, директорий и файлов.

Модели АС, формируемые в рамках МАИТ, являются сопряженными – присутствуют методики, обеспечивающие последовательное отображение модельных представлений на всем процессе создания АС. Структуры моделей фиксируются в виде диаграмм и спецификаций.

Исходя из рассмотренных возможностей методологии автоматизации интеллектуального труда можно выделить следующие преимущества:

- наличие формального описания всех моделей, обеспечивающего их единообразное представление.
- наличие универсального модельного представления на каждом этапе создания, обеспечивающего порождение и интеграцию прикладных моделей предметных задач.
- наличие целостной системы взаимосвязанных модельных представлений автоматизируемых задач – от начального до даталогического представления, которые содержательно увязаны на основе семантического представления этих задач;
- наличие инфологического модельного представления – инвариантного к среде реализации, позволяющего получить целостный проект АС без учета технологической среды.

1.1.5 Сравнительный анализ рассмотренных подходов и методов, применяемых при создании АС

Для выполнения сравнительного анализа методологий и входящих в их состав методов был сформулирован следующий перечень критериев:

– назначение методологии / метода;

– область использования методологии / метода;

– поддержка этапов жизненного цикла создаваемой АС при применении методологии*;*

– разнообразие модельных представлений на этапах создания АС при применении

методологии;

– степень наглядности полученных результатов моделирования;

– возможность увязки (сопряжения) моделей АС в процессе её создания;

Результаты сравнительного анализа приведены в таблице 1.1.

В качестве модели жизненного цикла выбрана модель промышленного способа создания АС (рис. 1.9).

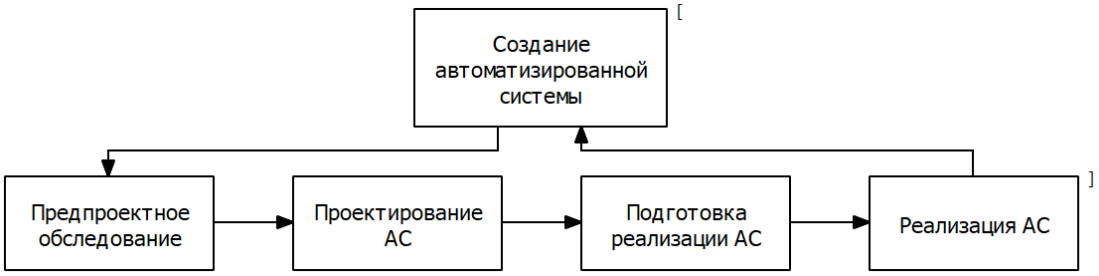


Рис. 1.9 Промышленный способ создания АС

Промышленный способ создания заключается в последовательном выполнении следующих этапов:

Предпроектное обследование – начальный этап производства АС, при котором разрабатывается начальная модель предметной области, создаваемой АС, формируются и уточняются требования к системе.

Проектирование АС – этап, задачей которого является получение такого модельного представления АС, которое будет отражать организацию информации и логику её обработки независимо от будущей среды реализации.

Подготовка реализации АС – этап, на котором формируется модель АС, зависимая от среды реализации.

Реализация АС – этап, на котором происходит разработка программного кода системы, его тестирование и отладка.

Анализ методологий структурного подхода показал, что их использование наиболее эффективно на начальных этапах создания АС, так как формируемые модели отражают фактическое состояние предметной области в виде структурных диаграмм. Однако интеграция различных моделей при работе с большим комплексом предметных задач – довольно затруднительный процесс, который требует специалиста высокой квалификации.

Представление концептуальных конструкций в виде ER-диаграмм эффективно только для задач низкой и средней сложности и позволяет обеспечить наглядность и простоту работы с этими моделями. Сложные модели перегружены графическими элементами, которые затруднительно размещать на одном листе, избегая пересечений и наложений.

Таблица 1.1 Сравнительные анализ методов создания АС

Методология/методы	**IDEF (IDEF0, IDEF1x, IDEF3),**	**ARIS**	**UML, IDEF4**	**Онтологии (IDEF5)**	**МАИТ**
Подход	Структурный		Объектно-ориентированный		Когнитивный
Назначение	Моделирование бизнес-процессов		Моделирование программных комплексов.	Разработка систем знаний	Автоматизация проектно-конструкторской и управленческой деятельности.
Область применения	Моделирование бизнес-процессов.		UML является универсальным языком, который применяют для описания практически любой деятельности. IDEF4 используется только в области программной инженерии.	Разработка систем управления знаниями.	Теоретический аппарат МАИТ позволяет применять её методы при научных исследованиях, а методики моделирования целесообразно применять при разработке автоматизированных систем и программных комплексов, в частности.
Поддерживаемые этапы ЖЦ	Предпроектное обследование.		Предпроектное обследование, подготовка реализации, реализация.	Предпроектное обследование, реализация.	Предпроектное обследование, проектирование, подготовка реализации, реализация.
Разнообразие модельных представлений	Различные модели методологий отражают информационный и структурный аспекты предметной области.		Совокупность моделей, отражающих различные компоненты АС (информационный, функциональный, интерфейсный, сетевой), могут применяться на разных этапах создания АС с различной детализацией.	Одна сетевая модели системы знаний.	Для каждого этапа создания АС формируется собственное модельное представление, содержание модели на трех уровнях абстрагирования. Каждая модель – совокупность составляющих, отражающих компоненты автоматизируемой задачи.
Наглядность и формы представления результатов	Легко читаемые структурные диаграммы.	Удобная графическая нотация с регламентированной цветовой дифференциацией.	Диаграммы, как основное средство моделирования, обеспечивают наглядность моделей только для решения небольших задач (до 20-30 структурных элементов на диаграмме). Обеспечение наглядности требует значительных временных затрат на определение оптимального взаиморасположения элементов на диаграмме. Текстовое представление для хранения и обработки моделей в вычислительной среде.		Диаграммы наглядно описывают модель при её документировании. Спецификации позволяют сохранять и обрабатывать данные моделей в вычислительной среде.
Сопряжение моделей	Нет	Присутствует возможность сопряжения моделей различного типа.	Да (за счет использования одной нотации на всех этапах и наличие её формального описания).	Нет.	Увязке подлежат модели АС на смежных этапах её создания в рамках всего ЖЦ. Локальные модели различных задач на одном этапе создания можно интегрировать в обобщенную.

Отсутствие методов сопряжения различных составляющих модели вызывает необходимость постоянного анализа полной модели при согласовании и доработки отдельных её частей.

Объектно-ориентированный подход имеет ряд преимуществ по отношению к структурному:

– Единая модель, увязывающая информационный и функциональный аспекты моделируемой системы. Каждый класс обладает своим поведением, что выражается в его методах. В структурном подходе эти аспекты отражаются в различных моделях, подлежащих согласованию.

– Объектное представление системы более точно описывает явления реального мира, носящего объектный характер.

– Объектные модели легко переносимы на любой объектно-ориентированный язык программирования.

– Применение инкапсуляции – сокрытии собственной независимой от контекста структуры для каждого объекта добавляет элементам модели гибкость и увеличивает их степень интероперабельности.

Исходя из выявленных преимуществ, МАИТ был выбран в качестве методологического базиса для дальнейшего исследования.

1.2 Анализ средств поддержки методов объектно-ориентированного и когнитивного подходов

1.2.1 Общие положения и обоснования

Так как когнитивный подход является самым молодым из рассмотренных и активно развивается, был проведен сравнительный анализ средств его поддержки с самым распространённым на сегодняшний момент подходом – объектно-ориентированным.

Сравнительный анализ средств создания автоматизированных систем в рамках объектно-ориентированного и когнитивного подходов был проведён в соответствии со следующей системой критериев:

-поддержка моделирования различных компонентов предметных задач на всех этапах промышленного способа создания АС;

-возможность увязки компонентов моделей автоматизируемых задач в рамках смежных этапов создания;

-наличие разнообразия форм ввода и представления результатов моделирования;

-возможность расширения функциональных возможностей;

-уровень сложности инструментальных средств;

-возможность сопряжения графического и табличного описаний модельных представлений.

Средства поддержки рассматриваемых подходов реализованы в большом классе программных продуктов: от простых средств анализа и документирования, до полномасштабных сред поддержки процесса создания, увязывающих все процессы жизненного цикла. [27]

По причине распространённости объектно-ориентированного подхода программные средства его поддержки занимают центральное место в инструментарии современных разработчиков. В процессе развития эти средства проделали путь от небольших программных продуктов, поддерживающих отдельные процедуры создания автоматизированных систем, до комплексных интегрированных сред, поддерживающих весь жизненный цикл. Исходя из этого можно выделить следующие классы систем:

1) CASE-средства поддержки определенных этапов;
2) интегрированные среды разработки;
3) ALM-среды, совмещающие в себе как среду разработки программных продуктов, так и функции управления жизненным циклом разрабатываемых продуктов.

1.2.2 Анализ средств поддержки методов объектно-ориентированного подхода

Однако, в силу методологических недостатков объектно-ориентированного подхода, средства его поддержки опускают этап проектирования автоматизированных систем, совершая скачек от этапа предпроектного обследования к этапу подготовки реализации.

В современном мире существуют несколько технологий разработки автоматизированных систем, основанных на применении объектно-ориентированного подхода. Как правило, они служат для одной цели – сопряжение модели АС и её программного кода, и ставят задачу сокращения времени при переходе на этап реализации. Примером таких технологий служат ORM и MDA.

ORM (Object-relational mapping) - технология разработки программного обеспечения, которая стыкует объектно-ориентированное и реляционное представление АС. Технология позволяет производить одновременную разработку, как объектной структуры АС, так и базы данных для хранения информации об объектах и избавляет программиста от написания однообразного кода.

Данная технология предполагает предварительную разработку даталогической структурной модели (ER-модель любой нотации), которая в последствии будет трансформирована в класс

объектно-ориентированного подхода и в сущность реляционной базы данных. С полученной базой данных можно будет работать как традиционными методами – через запросы, так и через методы, реализованные на языке программирования. Большинство современных реализаций ORM позволяют программисту применять два различных подхода к ORM – code fist, когда реляционная база данных формируется на основе программного кода, и data first – программный код на основании реляционной базы данных. Оба подхода дают разработчику шаблон начального программного решения, который можно настраивать и модифицировать при дальнейшей разработке.

MDA (Model Driven Architecture, архитектура, управляемая моделью) – технология разрабатываемая OMG на основе языка UML. OMG не занимается разработкой программных средств, поддерживающих эту технологию, предполагая, что это будут делать сторонние разработчики. Фактически на данный момент MDA – идеализированный концепт, в полном объеме не реализованный ни в одном средстве поддержки, описывающий разработку программных продуктов на основе моделирования, который базируется на следующих средствах (рис. 1.10):

- UML – как языковое средство моделирования.
- MOF – спецификации формирования метамоделей UML.
- CMW (Common Warehouse Model, база общих моделей) – представляет собой совокупность метамоделей различных концепций, наиболее часто используемых в базах данных и бизнес-анализе. Фактически CMW – набор средств расширения UML для моделирования хранилищ данных, основанных на часто используемых технологиях.
- XMI – язык представления UML модели в формально описанный текстовый формат, который используется как средство хранения и представления моделей в вычислительной среде.

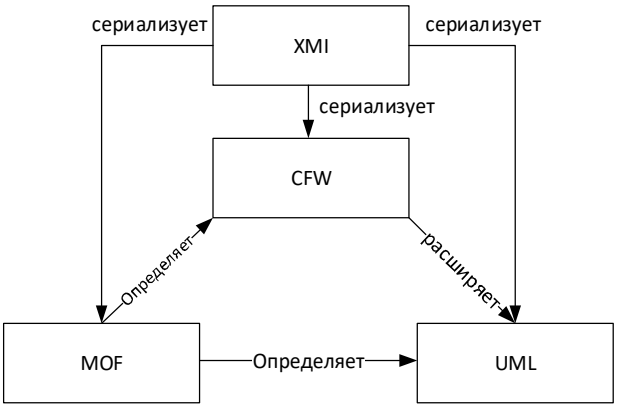


Рис. 1.10 Взаимосвязь понятий, определяющих технологию MDA

В процессе разработки программного обеспечения по технологии MDA формируются следующие модели (см рис. 1.11):

Вычислительно-независимая модель (Computation Independent Model, CIM) описывает общие требования к системе, её окружение и словарь используемых терминов. В модель не

включаются технические сведения, конкретные структуры данных, алгоритмы функционирования, сведения об интерфейсах. CIM – максимально общая и независимая от реализации системы модель. Модель может, но не обязана, быть преобразована в платформенно-независимую модель. Однако для этого она должна быть разработана на языке UML.

Платформенно-независимая модель (Platform Independent Model, PIM) – описывает все требования к системе, её элементы, алгоритмы работы и интерфейсы без конкретных примеров физической реализации и сведений о программно-технической среде функционирования. [28]. Выполненная с учетом требований спецификации MDA, PIM модель может быть автоматически преобразована в платформенно-зависимую модель.

Платформенно-зависимая модель (Platform-specific model, PSM) – формируется на основе модели PIM. Для этого преобразования должны использоваться готовые инструменты и библиотеки правил под конкретную выбранную платформу. Из одной PIM может быть получено несколько PSM, под каждую конкретную программно-техническую среду реализации. Такой обход облегчает разработку и поддержу системы, ориентированной на различные платформы. Платформенно-зависимая модель описывает состав, структуру, функциональность системы в терминах конкретной платформы. Модель PSM создается на основе модели платформы, определяющей правила перехода от PIM к PSM.

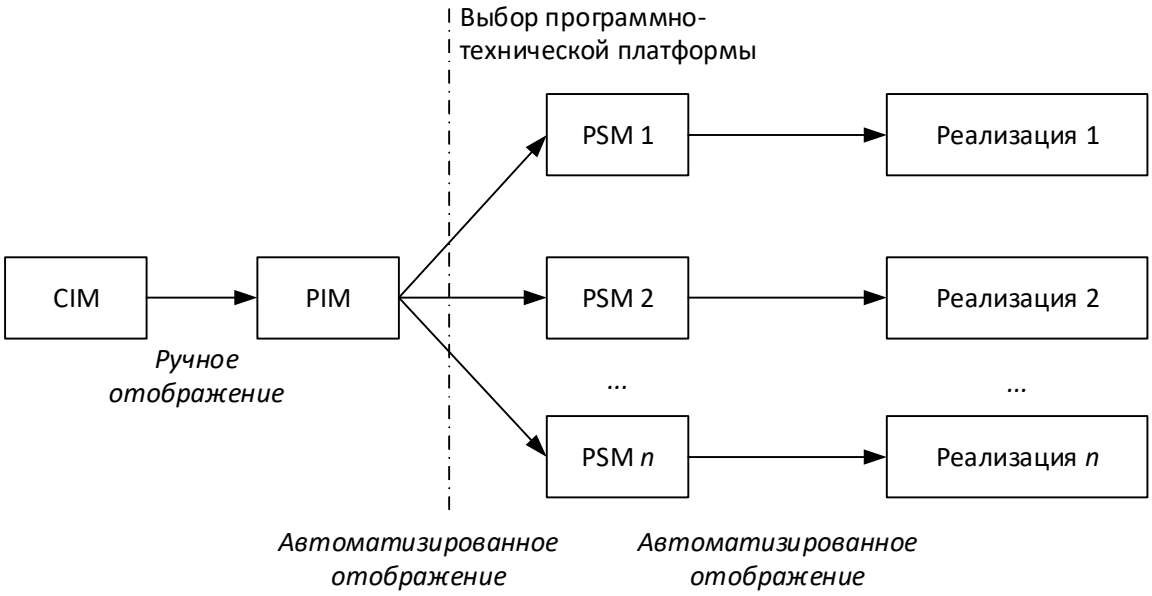


Рис. 1.11 Взаимосвязь моделей в MDA

Для обеспечения автоматизированного отображения из PIM в PSM модель платформы должна быть выполнена с учетом спецификации MDA на языке UML. Модель описывает технические характеристики, интерфейсы, функции платформы и может представлена в виде технических спецификаций и описаний, но в идеальном варианте должна быть представлена как часть CFW.

После формирования PSM модели можно выполнить процедуру генерации программного

кода и получить шаблон программного решения с учетом выбранной платформы и языковых средств. Степень проработки модели напрямую определяет полноту и качество полученного программного кода.

IBM Rational Rose / Rational Software Architect – разработанная компанией Rational Software среда для визуальной разработки программного обеспечения, которая впоследствии была приобретена компанией IBM(см рис. 1.12). Rational Rose и её современный вариант – Rational Software Architect, обеспечивают разработчиков средством разработки архитектур программных решений на основе визуальных моделей UML 2.1. Последние версии программы созданы на базе среды разработки Eclipce. Rational Software Architect позволяет разработать приложения на высоких уровнях абстракции, при этом координируя группу специалистов-разработчиков и лиц, заинтересованных в выполнении проекта. Также система позволяет применять отличные от UML методы моделирования и различные расширения UML. [29] [1] [30]

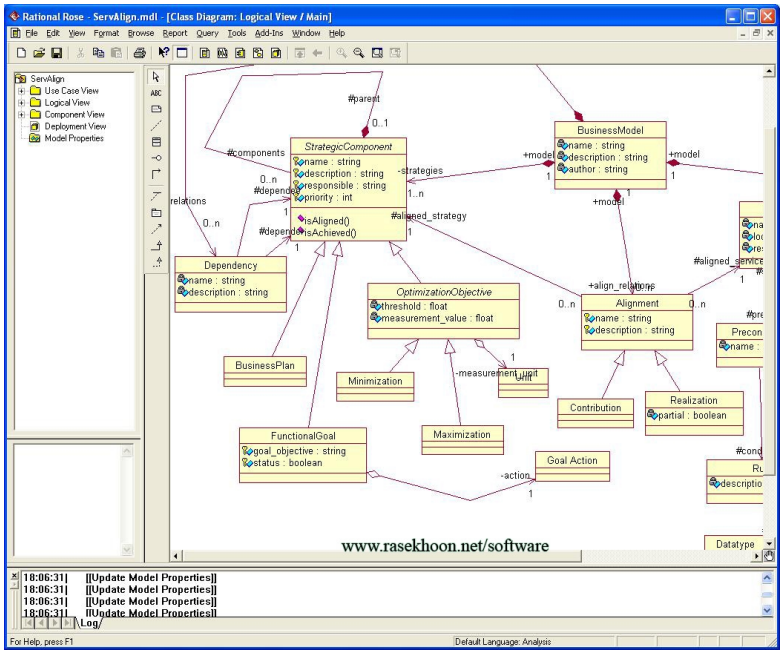


Рис. 1.12 Графический интерфейс системы «IBM Rational Rose»

Rational Software Architect плотно интегрируется со средой разработки и обеспечивает формирование шаблона проекта на самых распространённых языках (C#, C++, Java. и др.) для последующего уточнения. Имеется возможность генерации программного кода на основании файлов конфигураций и стандартизированных описаний интерфейсов.

Очевидным преимуществам Rational Rose является поддержка большого количества технологий и большой опыт применения при проектировании автоматизированных систем различного назначения.

Eclipse Modeling Framework (EMF) – платформа моделирования и средство генерации кода для создания инструментов и других приложений на основе структурированной модели данных. EMF предоставляет инструменты и поддержку среды выполнения и создания набора классов Java

на основании модели, описанной согласно XMI, а также набор классов-адаптеров и базовый редактор, которые позволяют просматривать и редактировать модель на основе команд. (рис. 1.13). [31]

EMF (core) - это общий стандарт для моделей данных, на котором основаны многие технологии и фреймворки. Сюда входят серверные решения, платформы сохраняемости, платформы пользовательского интерфейса и поддержка преобразований.

Ключевыми преимуществами Eclipse Modeling Framework является:

- поддержка большого количества технологий, сопряжённых с процессом визуального моделирования;
- активная поддержка и развитие рассматриваемого средства со стороны сторонних разработчиков.

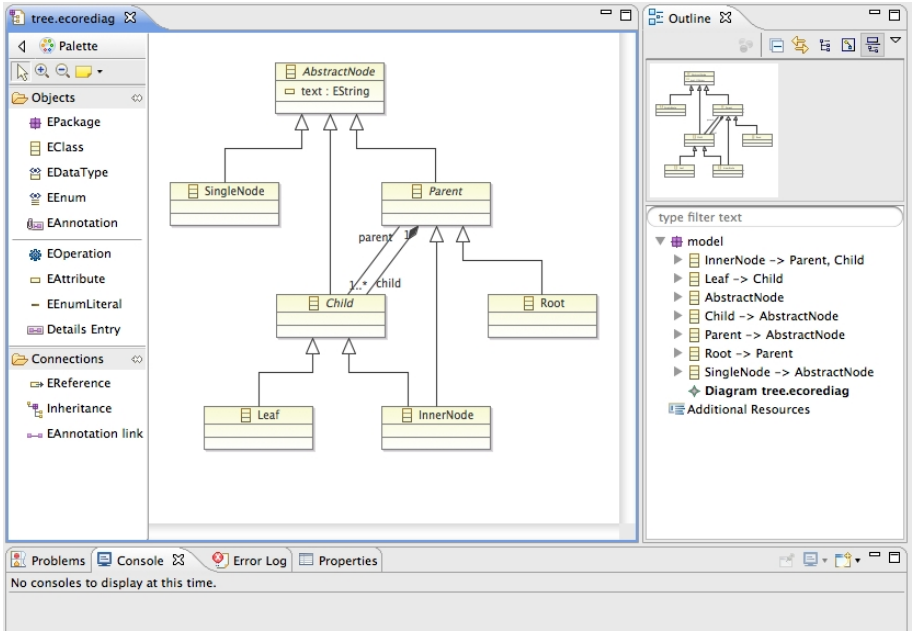


Рис. 1.13 Графический интерфейс среды «Eclipse»

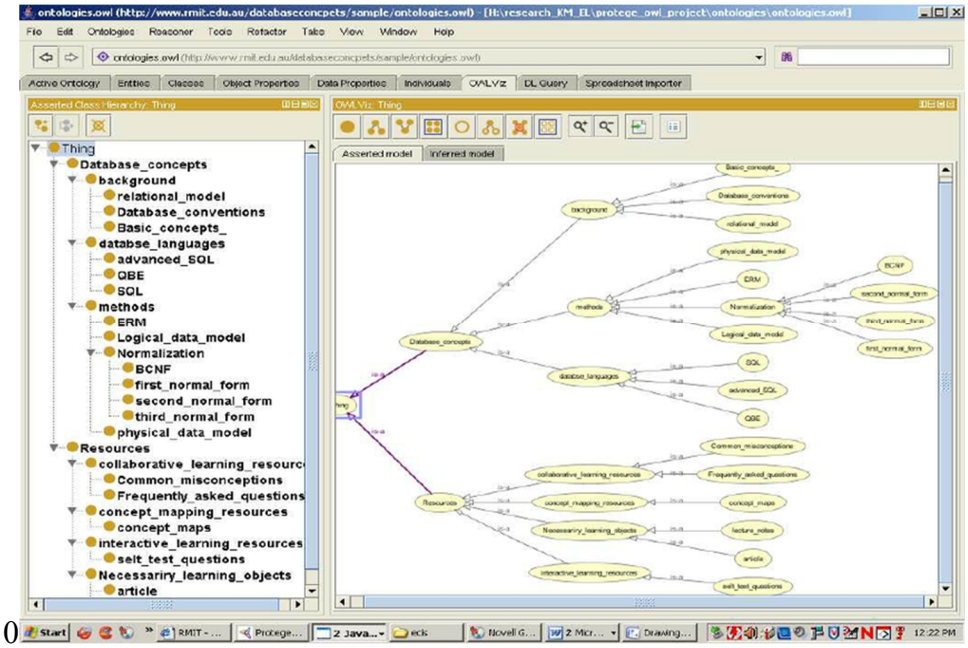
Visual Studio – созданная корпорацией Microsoft интегрированная среда разработки традиционно считается инструментом чистого программирования и отладки программного обеспечения. Однако с учетом современных тенденций в среду стали добавляться элементы манипулирования программным кодом через связанные с ним модели с более ранних этапов разработки. Visual Studio поддерживает процесс «обратного проектирования», позволяющий формировать модели по уже запрограммированной системе для перестановки процесса разработки на новые программно-технические «рельсы». [32]

Visual Studio предоставляет возможность генерации структуры программного кода, используя в качестве исходных данных диаграмму классов. Также среда содержит Entity Framework, реализующий технологию ORM (Object-Relational Mapping), позволяющий в визуальном редакторе проектировать гибрид реляционной и объектно-ориентированной базы данных, с которой можно

работать традиционным способом – с помощью языка SQL, так и с множеством экземпляров классов, как принято в объектно-ориентированном программировании. [33]

Особенностью Visual Studio является её ориентация именно на процесс программирования, а функциональность, связанная с моделированием АС, является расширением её базовых функций. Также среда поддерживает расширение своих возможностей путем установки плагинов.

Protégé является свободным программным средством с открытым исходным кодом для редактирования онтологий. Интерфейс графического редактора онтологий представлен на рис. 1.14.



0

Рис. 1.14 Графический интерфейс среды «Protégé»

Protege основан на фреймовой модели представления знаний. Структура онтологии сделана по типу древовидного каталога. В системе есть модуль, позволяющий на основе сформированной онтологии создавать интерфейсные формы. Созданные формы можно использовать для введения экземпляров классов и подклассов, формируя, тем самым, систему знаний.

Protege основан на фреймовой модели представления знаний. Структура онтологии сделана по типу древовидного каталога. В системе есть модуль, позволяющий на основе сформированной онтологии создавать интерфейсные формы. Созданные формы можно использовать для заполнения атрибутов экземпляров классов и подклассов, формируя, тем самым, систему знаний.

Особенностью системы является её открытый код и поддержка процесса разработки расширений. В базовой комплектации система снабжена набором расширений, добавляющих поддержку моделей в разных форматах (стандартный текстовый, базы данных, UML, языки XML, XOL, SHOE, RDF и RDFS, DAML+OIL, OWL).

CASEBERRY – среда автоматизированного проектирования и разработки приложений, созданная Институтом Информационных Систем. Продукт предоставляет средства построения UML моделей (рис. 1.15) и функции кодогенерации под платформы .Net на языках C# и Visual Basic,

также присутствует возможность генерации сущностей для реляционных баз данных Ado.Net. Среда является расширяемой посредством установки плагинов, добавляющих интеграцию со многими распространёнными источниками данных.

Отличительной возможностью CASEBERRY является возможность генерации интерфейса под разработанную структуру данных и управление им еще на этапе моделирования АС. [34]

Из недостатков рассматриваемой системы можно выделить её ориентацию на задачи небольшой сложности и отсутствие поддержки расширений, позволяющих добавлять новые языки программирования.

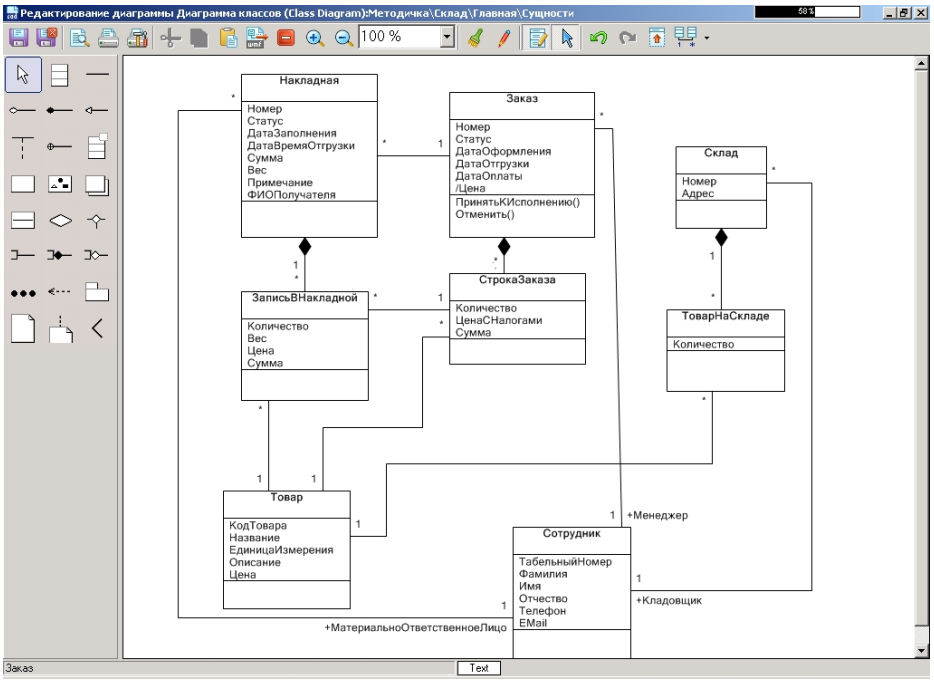


Рис. 1.15 Графический интерфейс среды «CASEBERRY»

1.2.3 Анализ средств поддержки методов когнитивного подхода

Model – средство поддержки этапа формирования начальной модели по полуавтоматизированной технологии, т.е. автоматическое формирование контрольных диаграмм на основании введённых спецификаций (рис. 1.16).

Средство применяется на этапе предпроектного обследования и помимо формирования информационной и функциональной частей модели позволяет выполнять их реструктуризацию.

Model позволяет формировать систему предметных действий и настраивать их информационную связность в рамках начальной модели МАИТ.

Разработанный на базе Microsoft Access 2007 программный комплекс Model может быть

интегрирован с другими средами из пакета Microsoft Office для формирования отчетной документации.

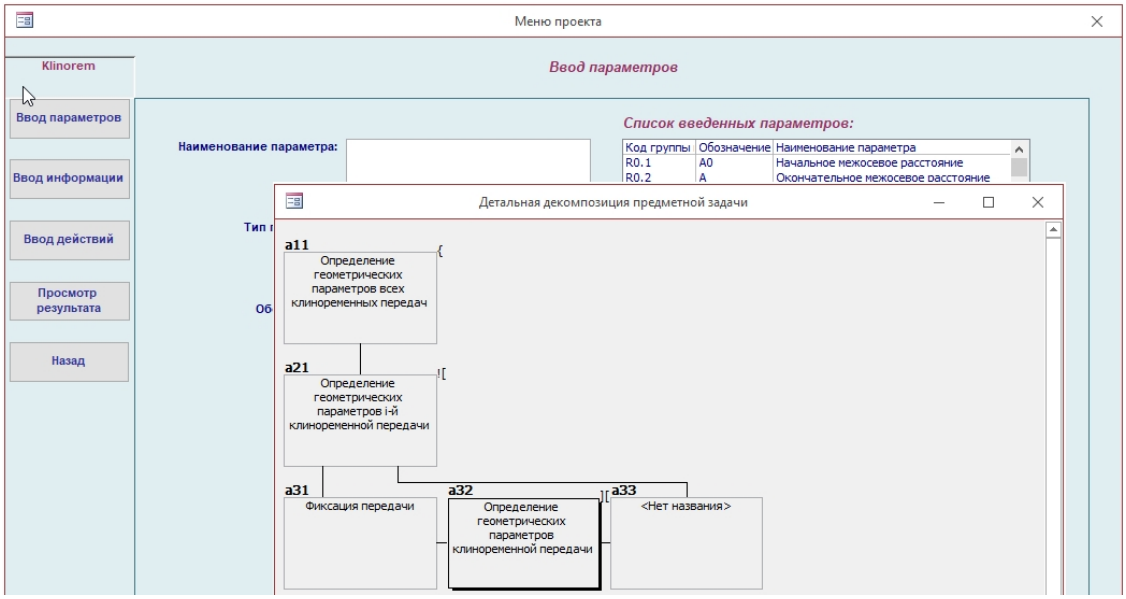


Рис. 1.16 Интерфейс программного комплекса «Model»

К существенным недостаткам можно отнести ориентацию на полуавтоматизированную технологию формирования моделей, не предусматривающую возможность визуального моделирования.

Concept – средство поддержки полуавтоматизированного концептуального моделирования. Основная задача программного средства – формирование концептуальной модели в вычислительной среде путем ввода данных спецификаций модели. На основании введённых данных программа в автоматическом режиме формирует контрольные диаграммы (рис. 1.17), которые в последствие сравниваются экспертами с первоначальным «ручным вариантом». На основе сравнения делается заключение о правильности модели.

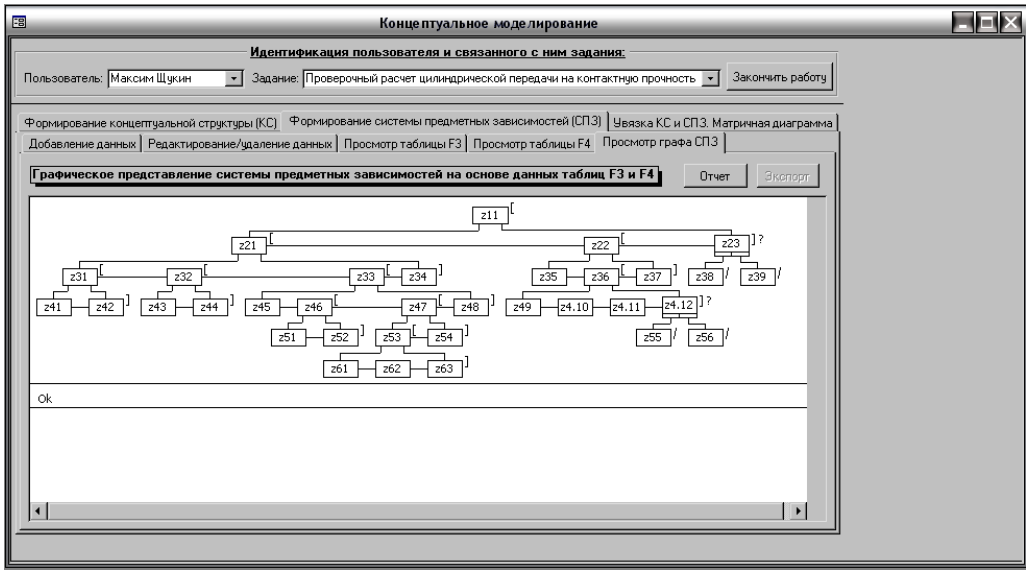


Рис. 1.17 Интерфейс программного комплекса «Concept»

Средство позволяет формировать основные структуры концептуальной модели (концептуальную структуру, систему предметных зависимостей первого рода, матричную диаграмму), формировать системы зависимостей второго рода и некоторые операции аналитической обработки моделей (расслоение концептуальной структуры под постоянную и переменную информацию).

Средство поддержки реализовано на платформе Microsoft Access, которое позволяет работать с базами данных. На данный момент является самым функциональным средством поддержки концептуального моделирования в рамках МАИТ, однако существенный недостаток – ориентация на полуавтоматизированную технологию формирования моделей, не предусматривающую возможность визуального моделирования.

CODA – среда визуального концептуального моделирования (рис. 1.18). Специфика рассматриваемого средства заключается в особенностях процесса его функционирования, а именно – применение графических редакторов для ввода структур концептуальной модели и последующая автоматическая генерация спецификаций модели на основании введенных данных. [35]

Программный комплекс CODA позволяет формировать основные структуры концептуальной модели (концептуальную структуру, систему предметных зависимостей первого рода, матричную диаграмму).

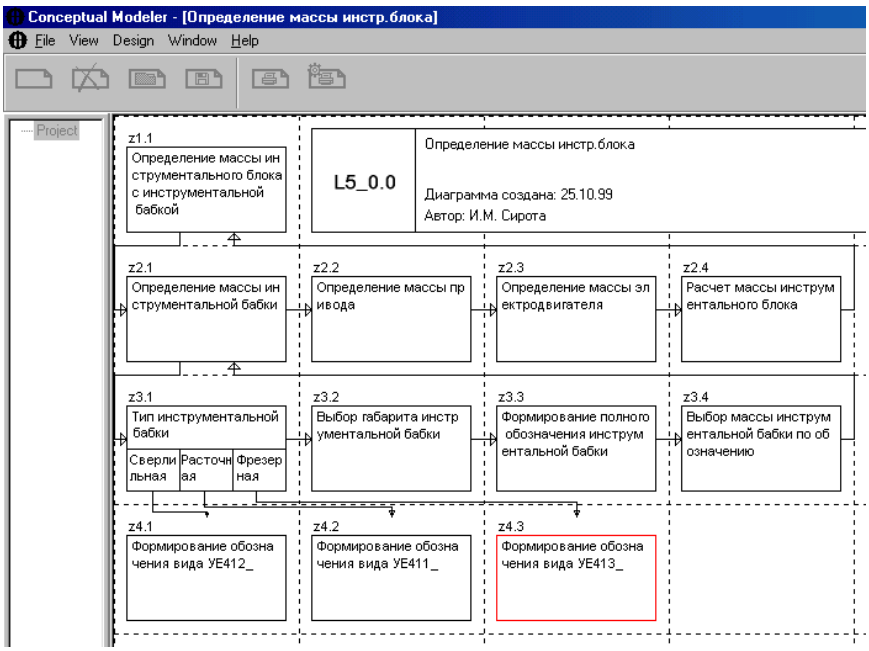


Рис. 1.18 Интерфейс графического редактора системы предметных зависимостей 1-го программного комплекса CODA

При формировании каждой структуры у пользователя доступны функции:

- анализа параметров компонентов модели, обеспечивающий корректность ввода модельных данных;
- визуального моделирования;

– формирования изображения;

– работы с базой данных проекта;

– различные вспомогательные функции.

В программном комплексе предусмотрена возможность документирования разработанных структур с предварительной настройкой размещения элементов модели на диаграммах, а диаграмм на листах. Присутствуют функции управления проектами.

Данное средство является единственным представителем систем графического моделирования в семействе инструментальных средств поддержки методологии автоматизации интеллектуального труда.

Infology – средство поддержки полуавтоматизированного инфологического моделирования, выполненное в среде MS Access. Модули программы позволяют работать с основной инфологической структурой, системой предметных манипуляций и доступов первого рода, моделью в целом. Также есть возможность выполнения различных процедур аналитической обработки инфологических моделей.

Модули форм и стандартные модули рассматриваемого средства выполняют следующие функции: [36]

– заполнение спецификаций;

– удаление данных из спецификаций;

– редактирование данных в спецификациях;

– визуализация инфологической модели.

Формирование диаграмм в модели происходит на основании данных, введённых в специальные формы. При этом процесс визуализации различных структур модели (рис. 1.19) доступен даже при неполных или частично некорректных данных.

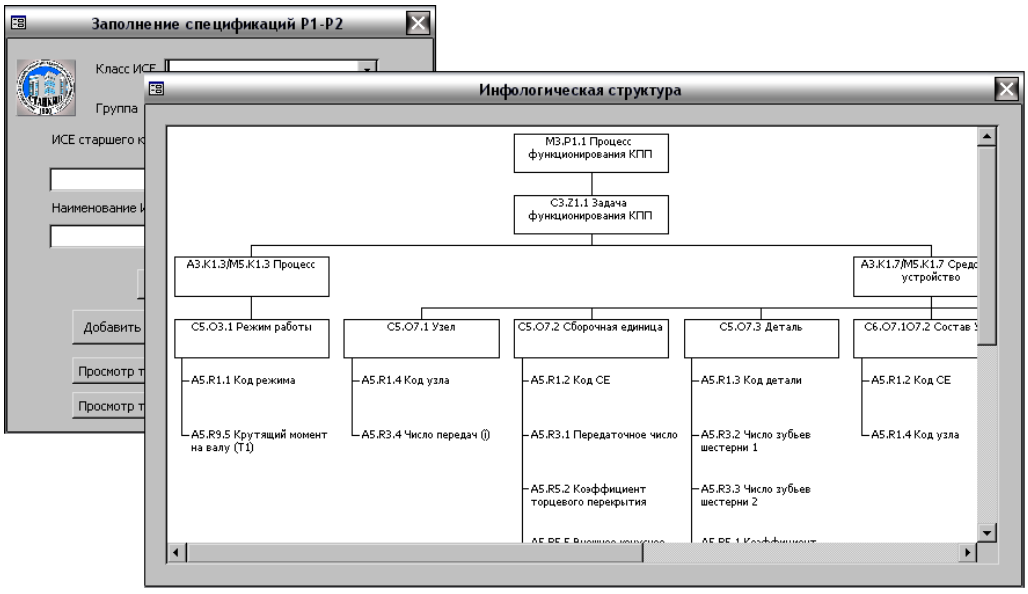


Рис. 1.19 Интерфейс программного комплекса Infology

Интегрированная среда (ИС) – среда, разработанная в МГТУ «СТАНКИН», объединяет в себе средства поддержки этапов промышленного жизненного цикла, выстроенного по методологии автоматизации интеллектуального труда. Среда объединяет в себе все созданные ранее (на момент разработки ИС) средства в виде функциональных модулей и обеспечивает их информационное взаимодействие за счет единообразного описания информационных, функциональных и интерфейсных компонентов моделей. Поэтапное развитие среды обеспечивается системой управления конфигурацией, позволяющей проводить наращивание функциональности среды путем изменения её файлового состава (рис. 1.20).

Интегрированная среда сформирована из трех основных компонентов: сервер баз данных, сервер приложений и клиентский модуль.

Сервер баз данных являться централизованным хранилищем информации для всей ИС. Он обеспечивает задачи информационного обмена между модулями, реализующими функциональные процедуры МАИТ, и между различными категориями пользователей среды, имеющих разные права доступа к одной и той же информации.

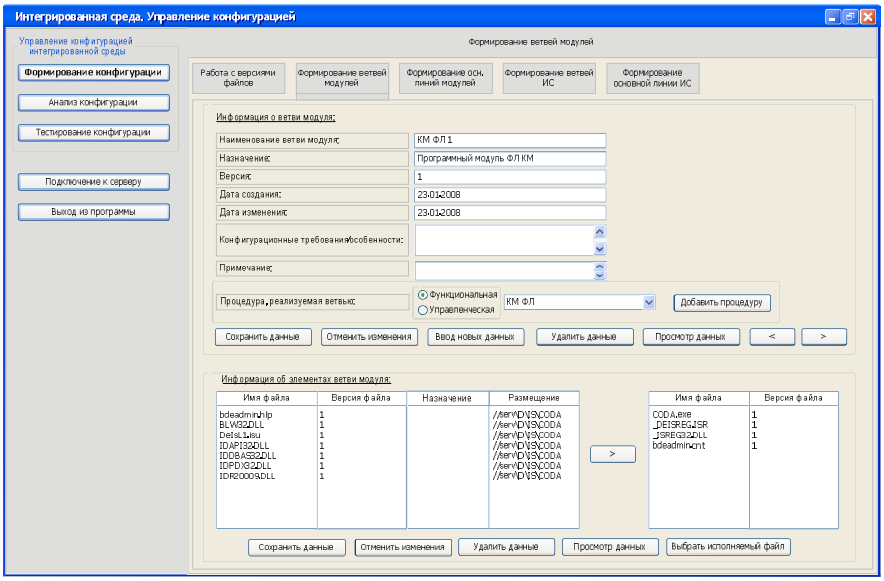


Рис. 1.20 Интерфейс средства управления конфигурацией интегрированной среды

Сервер приложений содержит набор средств поддержки процедур обработки моделей МАИТ. Вычислительно-сложные, сильно формализованные функциональные процедуры этапов создания АС по МАИТ наиболее эффективно выполнять на сервере приложений, развернутом на высокопроизводительной ЭВМ в режиме тонкого клиента, выводя пользователю лишь необходимую информацию о ходе работы. В то же время некоторые другие слабо-формализованные процедуры требуют постоянного диалога с пользователем. Выполнение таких процедур более эффективно производить на клиентском рабочем месте в режиме толстого клиента. Также сервер приложений обеспечивает управление проектной деятельностью, предоставляя возможности назначения прав доступа и ведя журналирование действий пользователя.

Клиент предоставляет пользователю возможности для взаимодействия с графическим интерфейсом интегрированной среды в режиме толстого или тонкого клиента. Клиент позволяет пользователю осуществлять разработку АС, управление разработкой АС и управление конфигурацией среды.

1.2.4 Сравнительный анализ рассмотренных инструментальны средств

Сравнительный анализ сред поддержки объектно-ориентированного и когнитивного подходов выполнялся в соответствии с основным критерием: полнота охвата процесса создания автоматизированной системы промышленным способом.

Распределение рассматриваемых средств поддержки по этапам промышленного способа создания АС показана на рисунке 1.21. На схеме приведены варианты сопряжения различных программных продуктов согласно их месту в жизненном цикле разрабатываемой системы.

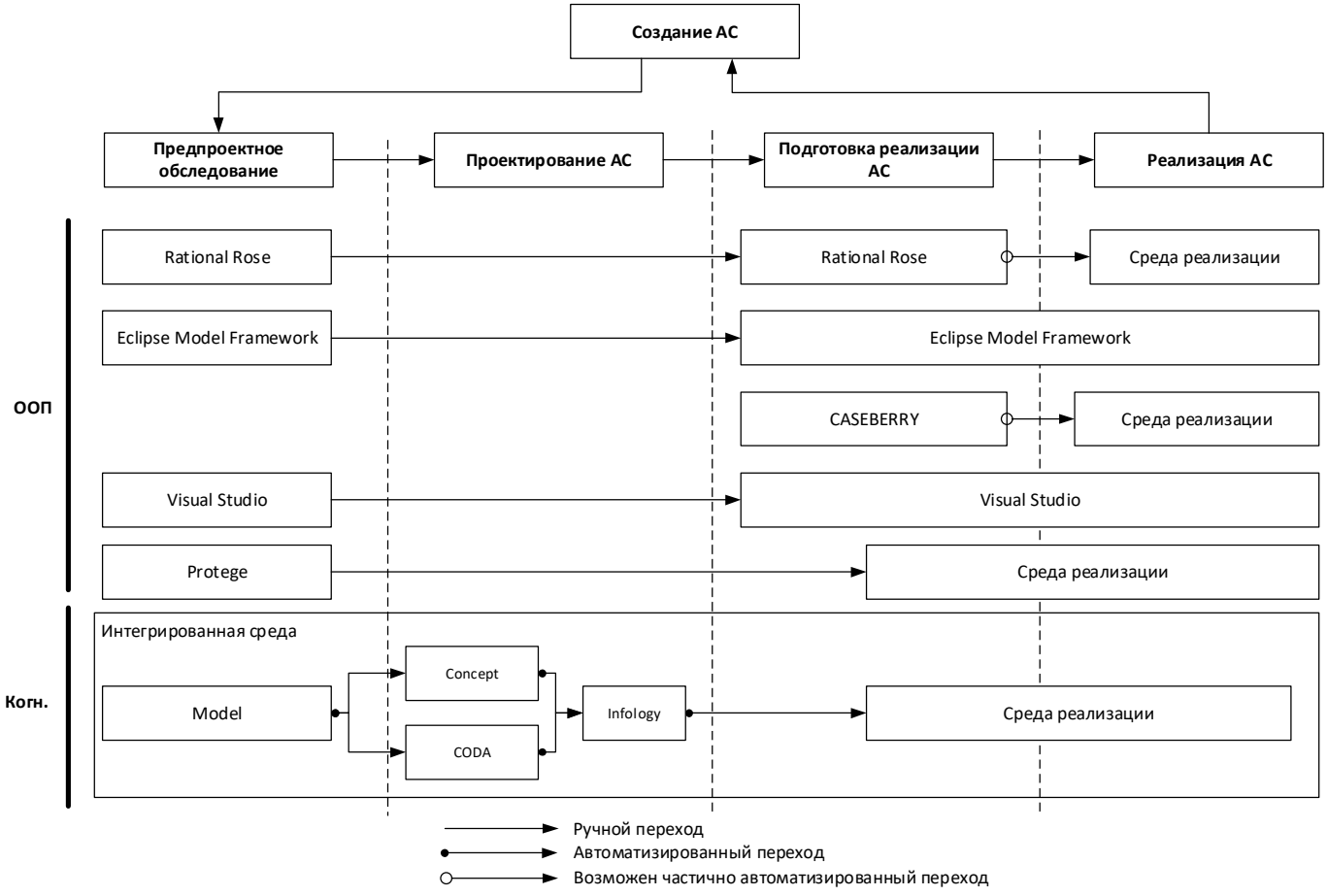


Рис. 1.21 Применение средств поддержки ОО и когнитивного подхода на различных этапах создания АС

Результаты сравнительного анализа функциональных возможностей рассматриваемых средств приведены в таблице 1.2.

Основные представители средств поддержки объектно-ориентированного подхода представляют собой крупные интегрированные среды разработки программного обеспечения (Visual Studio, Eclipse), которые в процессе своего развития приобрели возможность предварительного моделирования создаваемых систем перед их непосредственной реализацией. Так как эти среды поддерживают и определяют спектр технологий средств реализации, то и формируемые в них модели также будут ориентированы на эти среды, что не удовлетворяет промышленному способу создания.

Среды формирования онтологий, на примере редактора Protégé, используются в основном для создания обширных систем знаний на этапе исследования предметной области, которые, в последствии, могут быть использованы практически с любой средой реализации. Однако онтологии ориентированы, в первую очередь, на структурирование и систематизацию предметной области, а среды разработки позволяют формировать решения высокой степени интероперабельности, поэтому одна и та же онтология может быть совмещена с большим количеством сред реализации.

В процессе аналитического обзора средств поддержки ООП было показано, что практически все рассмотренные средства ориентированы на поздние этапы создания, на которых формируется модель АС, ориентированная на конкретную технологию реализации, и возможна процедура кодогенерации под выбранную платформу.

Средства организации и управления процессом групповой разработки позволяют вовлекать в процесс разработки одновременно большое количество различных специалистов и обеспечивают среду их взаимодействия.

В качестве основного достоинства средств поддержки ООП можно выделить ориентацию на визуальное моделирование, которое предоставляет разработчикам возможность создания моделей в виде удобочитаемых диаграмм. Однако такое представление информации затрудняет дальнейшую обработку возможностями ЭВМ. Поэтому для эффективного применения объектно-ориентированных моделей необходимо разрабатывать единый регламент работы с ними в рамках каждого коллектива разработчиков, состоящего из специалистов в разных областях для каждого отдельного проекта.

Совокупным недостатком всех средств поддержки объектно-ориентированного подхода является отсутствие интеграции и информационного сопряжения моделей как одинаковых компонентов различных моделей, так и различных компонентов в рамках одной модели автоматизированных систем.

Средства поддержки когнитивного подхода, а именно методологии автоматизации интеллектуального труда, были интегрированы в единую среду, которая увязывает все этапы

Таблица 1.2 Сравнительный анализ средств создания АС

подход	**ООП**					**Когн. подход**				
Система	IBM Rational Rose	Eclipse Model Framework	Visual Studio	CASEBERRY	Protege	Model	Concept	Infology	CODA	Интегриров. среда поддержки создания САПР
Класс системы	CASE	ИС	ALM	CASE	CASE	CASE	CASE	CASE	CASE	ИС
Охват этапов ЖЦ	Подготовка реализации; реализация	Подготовка реализации; реализация	Подготовка реализации; реализация	Подготовка реализации; реализация	Проектирование	Предпроектное обследование	Проектирование	Проектирование	Проектирование	Предпроектное обследование, проектирование, подготовка реализации, реализация
Компоненты предметных задач	Инф. Функ.	Инф. Функ.	Инф. Функ.	Инф. Функ.	Инф.	Инф. Функ МвЦ	Инф. Функ МвЦ	Инф. Функ МвЦ	Инф. Функ МвЦ	Инф. Функ, МвЦ
Увязка компонентов модели	**+**	**+**	**-**	**+**	**-**	**+**	**+**	**+**	**+**	**+**
Способы ввода данных	Граф.	Текст / Граф.	Текст / Граф.	Граф.	Граф.	Табл.	Табл.	Табл.	Граф.	Табл.
Возможность сопряжения граф. и табл. описаний модельных представлений	**-**	**-**	**-**	**-**	**-**	**+**	**+**	**+**	**+**	**+**

Сокращения названий компонентов: Инф. – информационный, Функ. – функциональный, МвЦ – модель в целом, увязка информационного и функционального компонентов.

процесса создания в единый процесс разработки за счет единообразного описания всех моделей автоматизированных систем, формируемых в соответствии с МАИТ, и их формализации. Одним из преимуществ методологии является двухэтапное проектирование АС, на котором создается семантическое представление (с точки зрения специалиста предметной области) и синтаксическое (ориентированное на последующее представление в вычислительной среде).

Однако интегрированная среда имеет ряд недостатков:

- Используемые средства поддержки этапов ЖЦ ориентированы на табличные формы ввода информации, что затрудняет процесс моделирования;
- Слабая поддержка заключительных этапов процесса создание и отсутствие этапа кодогенерации.
- Среда и её средства поддержки разработаны с использованием устаревших средств реализации, что затрудняет её дальнейшую модернизацию и развитие.

1.3 Выводы по главе

1. Анализ существующих подходов к автоматизации информационных и интеллектуальных задач и процессов, применяемых в них методов и инструментальных их средств поддержки, позволил выявить комплекс проблем, сдерживающих темпы внедрения и развития автоматизированных систем:

- достаточно большое разнообразие специалистов в области создания АС;
- необходимость постоянного взаимодействия разработчиков и заказчиков АС;
- высокая степень итеративности процесса разработки АС;
- необходимость внесения изменений в процессе разработки и совершенствования АС;
- отсутствие модельных представлений АС на ранних этапах её создания;
- обеспечение требуемого качества АС.

Существенными особенностями МАИТ по сравнению с методологиями структурного и объектно-ориентированного подхода являются:

– наличие системы взаимосвязанных модельных представлений автоматизируемых задач – от начального до даталогического представления, которые содержательно увязаны на основе семантического представления этих задач;

– наличие модельных представлений задач как инвариантных, так и ориентированных на программно-техническую среду и средства реализации АС.

2. Выполненный анализ инструментальных средств поддержки методов объектно-

ориентированного подхода позволил выделить ряд проблем:

– поддержка технологии визуального моделирования компонентов программных систем не обеспечивает дальнейшую их аналитическую обработку и/или отображение;

– развитие средств поддержки методов О-О подхода ориентировано на взаимоувязанное даталогическое моделирование разрабатываемой системы и процесса её программирования;

– наличие средств организации и управления процессом групповой разработки позволяет частично устранить трудности интеграции составляющих модельных представлений.

3. Разработанные ранее программные средства поддержки МАИТ оказывают независимую поддержку этапов предпроектного обследования и проектирования АС. Большинство этих комплексов позволяют формировать модели в виде табличных описаний всех составляющих моделей каждого этапа. При этом совокупность функциональных и дополнительных возможностей этих комплексов имела значительные пересечения. Ориентация только на табличное описание создаваемых модельных представлений усложняет процесс создания автоматизированных комплексов, что является их существенным недостатком. Для обеспечения возможности визуального моделирования и увязки программных модулей в рамках единого программного комплекса, обеспечивающего их взаимодействие, требуется разработка принципиально новой интегрированной среды, позволяющей формировать и обрабатывать конструкции различных моделей АС с максимальной степенью унификации.

Выявленные факторы определили необходимость разработки метода моделирования визуальной интегрированной среды, обеспечивающего эффективное проектирование АС, и средств поддержки процесса её развития, обеспечивающих дальнейшее поэтапное развитие самой среды. Это позволило сформулировать цель работы и поставить научную задачу для её достижения.

ГЛАВА 2
РАЗРАБОТКА МЕТОДОВ МОДЕЛИРОВАНИЯ ВИЗУАЛЬНОЙ ИНТЕГРИРОВАННОЙ СРЕДЫ

2.1 Особенности и требования к визуальной интегрированной среде

2.1.1 Особенности методологии автоматизации интеллектуального труда

Методология автоматизации интеллектуального труда представляет собой совокупность методов и методик для разработки автоматизированных систем и определяет промышленный способ их создания, разделенный на несколько этапов.

После завершения каждого этапа методологии формируется модельное представление АС, представленное в виде диаграмм и спецификаций (рис. 2.1). Модельные представления предыдущего этапа являются входными данными для следующего. Такой процесс переноса данных между моделями называется отображением и может выполняться в автоматическом режиме.

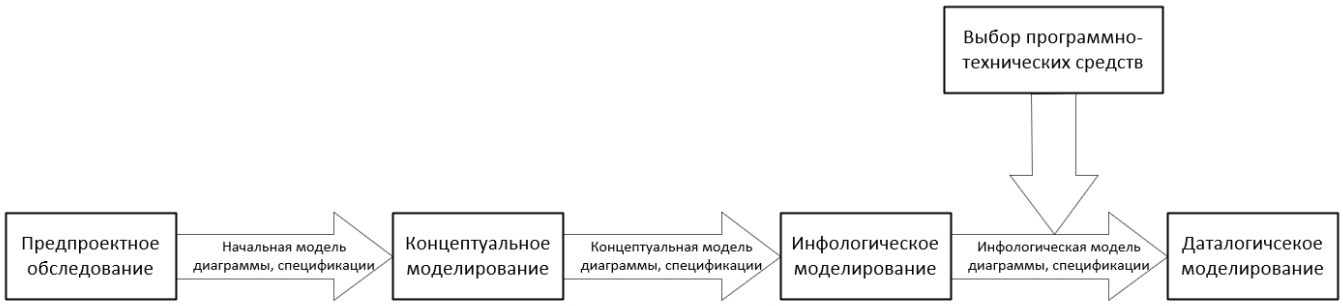


Рис. 2.1 Взаимосвязь этапов МАИТ

На этапе предпроектного обследования выполняется анализ предметной области. Анализ позволяет выявить иерархию задач, которые подлежат автоматизации. Так как отдельно взятые задачи могут являться комплексными, а значит могут быть разложены на более простые, необходимо выявить их приоритет в общей структуре проекта, на основании которого формируется последовательность работ автоматизации задач (рис. 2.2). В процессе обследования формируется начальная модель проекта. Для сложных задач формируется обобщённая модель, для простых – локальная, которая затем интегрируется в комплекс. Некоторые сложные задачи могут быть в начале выполнения проекта рассмотрены как простые, с последующей их проработкой. Также на этапе начального моделирования для каждой задачи выполняется классификация и систематизация

информации. После завершения предпроектного обследования начинается проектирование автоматизированной системы.

На этапе проектирования формируется проект АС, инвариантный к среде и средствам реализации. Особенностью этапа в МАИТ является его разделение на два последовательных подэтапа – концептуальное и инфологическое моделирование.

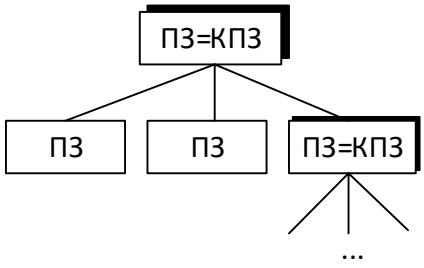


Рис. 2.2 Пример выявленной организации задач на этапе анализа предметной области

На этапе концептуального моделирования выполняется уточнение начальных моделей предпроектного обследования и формирование системы знаний о предметной области автоматизированной системы, содержащей понятия и их отношения. Сформированная концептуальная модель, зафиксированная в специальном виде, является входной информацией для инфологического моделирования.

Инфологическое моделирование подразумевает получение такого представления автоматизированной системы, которое описывает организацию информации, манипуляции и доступ к информации, но не учитывает особенности конкретных программно-технических сред и других особенностей реализации АС. После формирования проекта АС необходимо перейти к подготовке реализации.

Первым шагом подготовки реализации является выбор программно-технической среды и средств реализации, под которые разрабатываются даталогические модели. Одна инфологическая модель может быть исходной для множества даталогических, разработанных под различные программные платформы.

Заключительным этапом создания АС является процесс её реализации, включающий:

- настройку вычислительной среды (настройку графического интерфейса, настройку сетевого и межпрограммного взаимодействия);
- организацию процесса программирования;
- отладку и тестирование АС.

На каждом рассмотренном этапе моделирования выполняется однотипный набор процедур, организованный в определенную структуру, представленный на рисунках 2.4 - 2.9. Моделированию подлежат как комплексы предметных задач, так и отдельные предметные задачи, входящие в состав комплекса.

Представленные рисунки описаны в компактной графической нотации, созданной на основе методик формирования и описания моделей МАИТ, с применением базовых конструкций (рис. 2.3) и обозначений (табл. 2.1), применяемых в этих методиках. [37]

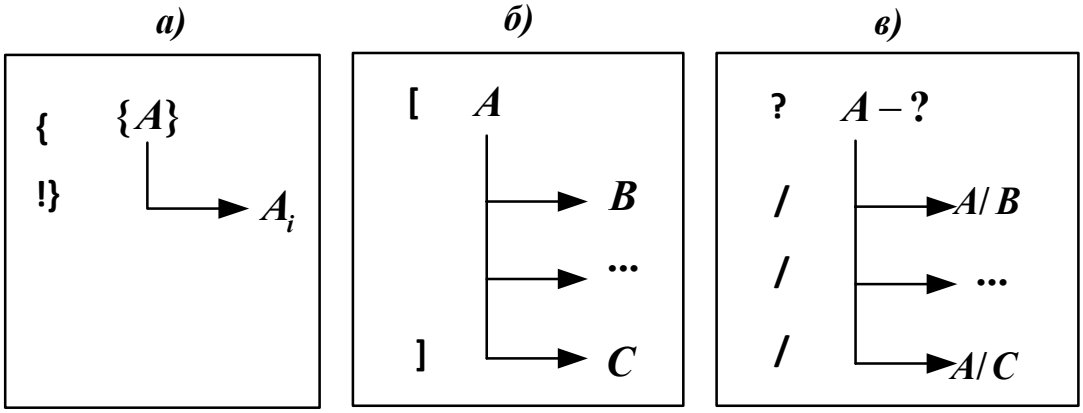


Рис. 2.3 Обозначение типовых конструкций. а) – итерация, б)–последовательность, в) – альтернатива

Типовая конструкция «итерация» (рис. 2.3а). ***{A}*** – заголовок итерации, формирование множества объектов ***A***. A_i – тело итерации, формирование ***i***-го элемента множества ***A***.

Типовая конструкция «последовательность» (рис. 2.3б). ***A*** – заголовок последовательности. ***B***, … , ***C*** – последовательно выполняемые действия.

Типовая конструкция «альтернатива» (рис. 2.3в). ***A –*** заголовок альтернативы, формирование объекта ***A*** альтернативными способами. ***A/B, … , A/C –*** альтернативные действия. ***A/B –*** формирование объекта ***A*** при выполнении условия ***B***, … , формирование объекта ***A*** в зависимости от условия ***C***.

Таблица 2.1 Условные обозначения на рис. 2.4 -2.9

Обозначение	Структурное свойство
{	Заголовок цикла
!	Фиксация аргумента итерации
[	Заголовок последовательности
]	Завершение последовательности
}	Завершение цикла
?	Заголовок альтернативы
/	Альтернативное действие

Таким образом, структуру процедур любого этапа создания АС можно представить как сложную иерархическую суперпозицию типовых конструкций, которые позволяют смоделировать такие типовые алгоритмические структуры как цикл и переключатель.

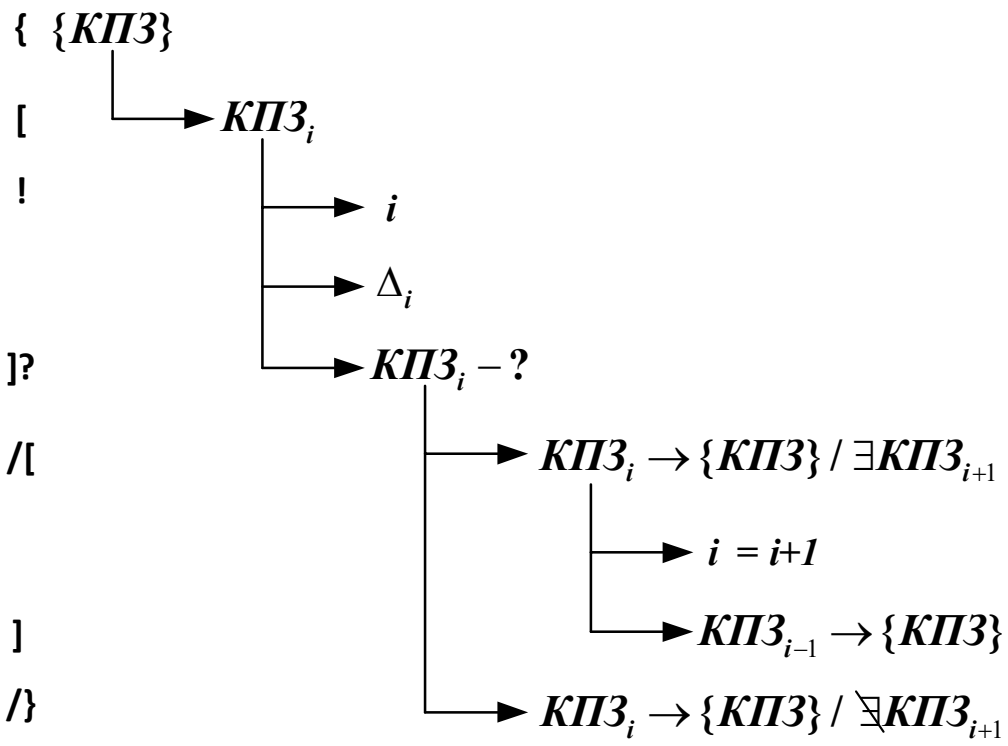


Рис. 2.4 Инвариантный набор процедур любого этапа создания АС
{КПЗ} *– моделирование всех комплексов в цикле,* ***КПЗ$_i$*** *– моделирование* ***i****-го КПЗ,* ***i*** *– фиксация* ***i****-го КПЗ,* Δ_i *– обработка* ***i****-го КПЗ,* ***КПЗ$_i$ – ?*** *– альтернатива завершения итерации,* ***КПЗ$_i \to$ {КПЗ} / ∃КПЗ$_{i+1}$*** *– фиксация результатов итерации в массиве комплексов при наличии следующего КПЗ ,* $i = i + 1$ *– инкрементация счетчика цикла,* ***КПЗ$_i \to$ {КПЗ} / ∄КПЗ$_{i+1}$*** *– фиксация результатов последней итерации в массиве комплексов.*

Обработка КПЗ заключается в применении к нему процедур формирования, разработки локальных моделей, аналитической обработки комплекса и его документирование. Под формированием понимается создание первичной обобщенной модели КПЗ, в которой каждой отдельной задаче соответствуют отдельные элементарные действия, являющиеся точками интеграции. При дальнейшей разработке локальные модели интегрируется в обобщенную модель в этих точках, образуя полную обобщенную модель. Процедура аналитической обработки комплекса обеспечивает структурную и временную оптимизацию модели и позволяет выявить дополнительные закономерности при дальнейшей автоматизации КПЗ. Процедура документирования обеспечивает формирование отчетных документов о выполненном этапе моделирования, содержащих описания всех компонентов модели КПЗ в табличном и графическом виде.

Моделирование всех локальных предметных задач подразумевает применение регламентированного перечня процедур обработки локальных моделей под каждую предметную задачу комплекса.

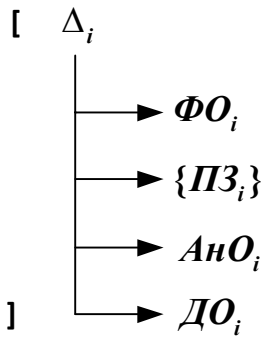


Рис. 2.5 Процедуры обработки i-го КПЗ:
$ФО_i$ – формирование начальной обобщенной модели комплекса, $\{ПЗ_i\}$ – моделирование всех предметных задач комплекса, $АнО_i$ – аналитическая обработка обобщенной модели комплекса, $ДО_i$ – документирование обобщенной модели комплекса.

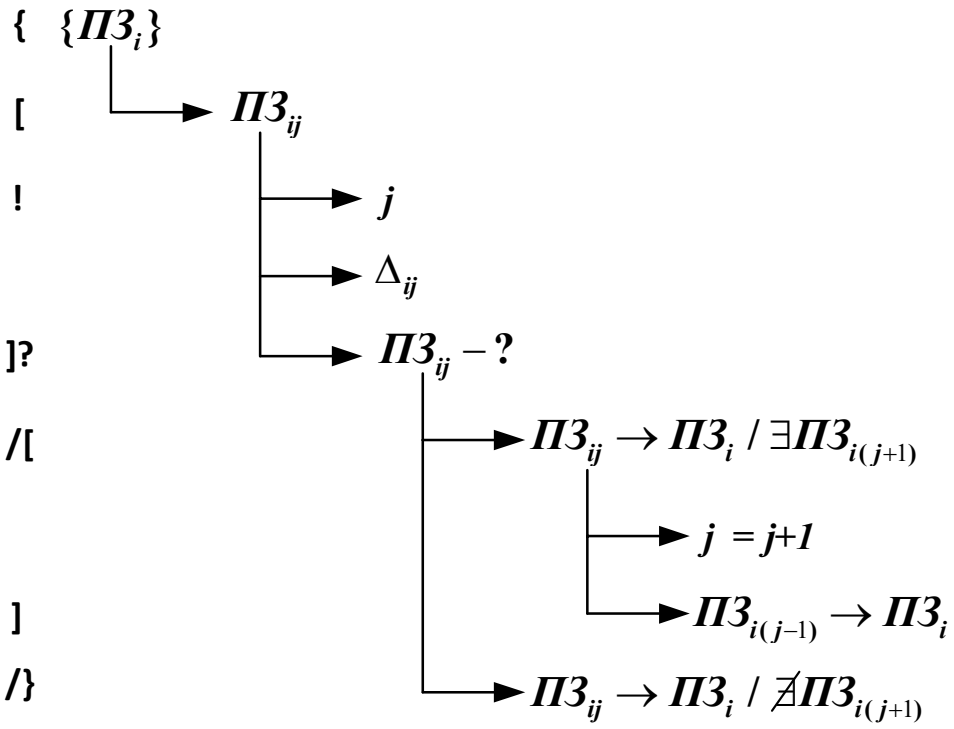


Рис. 2.6 Моделирование всех предметных задач i-го комплекса
*$ПЗ_{ij}$ – моделирование **j**-ой ПЗ текущего КПЗ, **j** – фиксация **j**-ой ПЗ, Δ_{ij} – обработка текущей ПЗ, $ПЗ_{ij} - ?$ – условие завершения цикла, $ПЗ_{ij} \rightarrow ПЗ_i / \exists ПЗ_{i(j+1)}$ – фиксация результатов итерации при наличии следующей ПЗ, $j = j + 1$ – инкрементация счетчика цикла, $ПЗ_{ij} \rightarrow ПЗ_i / \nexists ПЗ_{i(j+1)}$ – фиксация результатов последней итерации.*

Разработка локальных моделей подразумевает применение следующего перечня процедур на каждом этапе создания АС. Формирование локальной модели – разработка перечня версий модели на определенном этапе создания. Финальная версия модели выбирается экспертом на основании её анализа. Синтез локальной модели с обобщенной обеспечивает полное включение локальной модели в советующую ей точку интеграции в обобщённой. Документирование регламентирует описание модельных структур в графической и табличной формах.

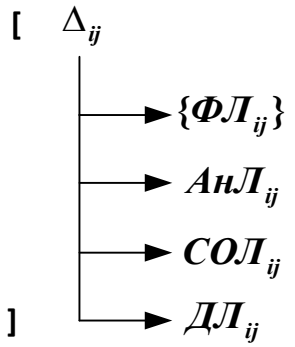


*Рис. 2.7 Процедуры обработки **j**-ой ПЗ **i**-го КПЗ*
$\{ФЛ_{ij}\}$ – формирование множества версий локальной модели задачи, $АнЛ_{ij}$ – аналитическая обработка локальной модели задачи, $СОЛ_{ij}$ – синтез локальной модели ПЗ с обобщений моделью КПЗ, $ДЛ_{ij}$ – документирование локальной модели задачи.

Формирование всех версий локальной модели подразумевает последовательное применение процедур разработки новой версии модели ПЗ и её анализ. Анализ версии локальной модели позволяет оценить полноту и корректность модельных данных на основании специальных контрольных диаграмм и знаний эксперта.

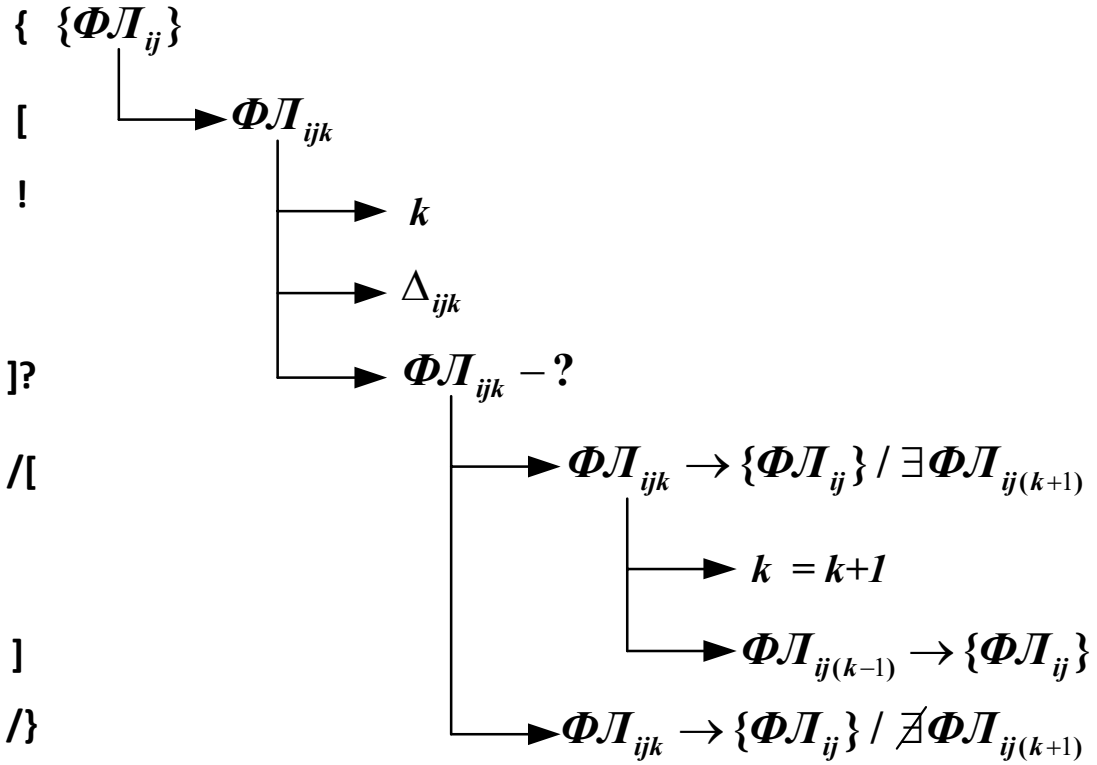


*Рис. 2.8 Моделирование всех версий локальной модели **j**-ой ПЗ **i**-го КПЗ*
*$\{ФЛ_{ij}\}$ – моделирование всех версий задач текущего КПЗ, $ФЛ_{ijk}$ – моделирование k-ой версии текущей ПЗ, **k** – фиксация **k**-ой версии ПЗ, Δ_{ijk} – обработка текущей версии ПЗ, $ФЛ_{ijk} - ?$ – условие завершения цикла, $ФЛ_{ijk} \to \{ФЛ_{ij}\} / \exists ФЛ_{ij(k+1)}$ – фиксация результатов итерации при наличии следующей версии ПЗ, $k = k + 1$ – инкрементация счетчика цикла, $ФЛ_{ijk} \to \{ФЛ_{ij}\} / \nexists ФЛ_{ij(k+1)}$ – фиксация результатов последней итерации.*

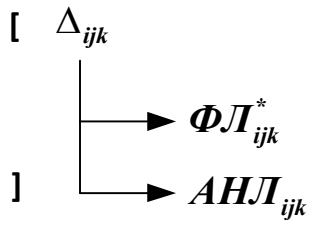


*Рис. 2.9 Процедура обработки **k**-ой версии модели **j**-ой задачи **i**-го комплекса*
***ФЛ**$^{*}_{ijk}$ – формирование текущей версии модели ПЗ, **АНЛ**$_{ijk}$ – анализ текущей версии модели ПЗ.*

Процессная унификация при обработке моделей.

Так как структуры моделей для комплекса задач и отдельной задачи аналогичны, то можно выделить 5 базовых процедур обработки моделей: формирование (Ф), анализ (АН), аналитическая обработка (Ан), синтез (С), документирование (Д). Исходя из этого, процесс создания автоматизированной системы можно представить как процесс применения этих процедур для задач различного уровня сложности. Такое представление процесса моделирования с выделением информационных связей между процедурами приведено на рисунке 2.10. [38]

.

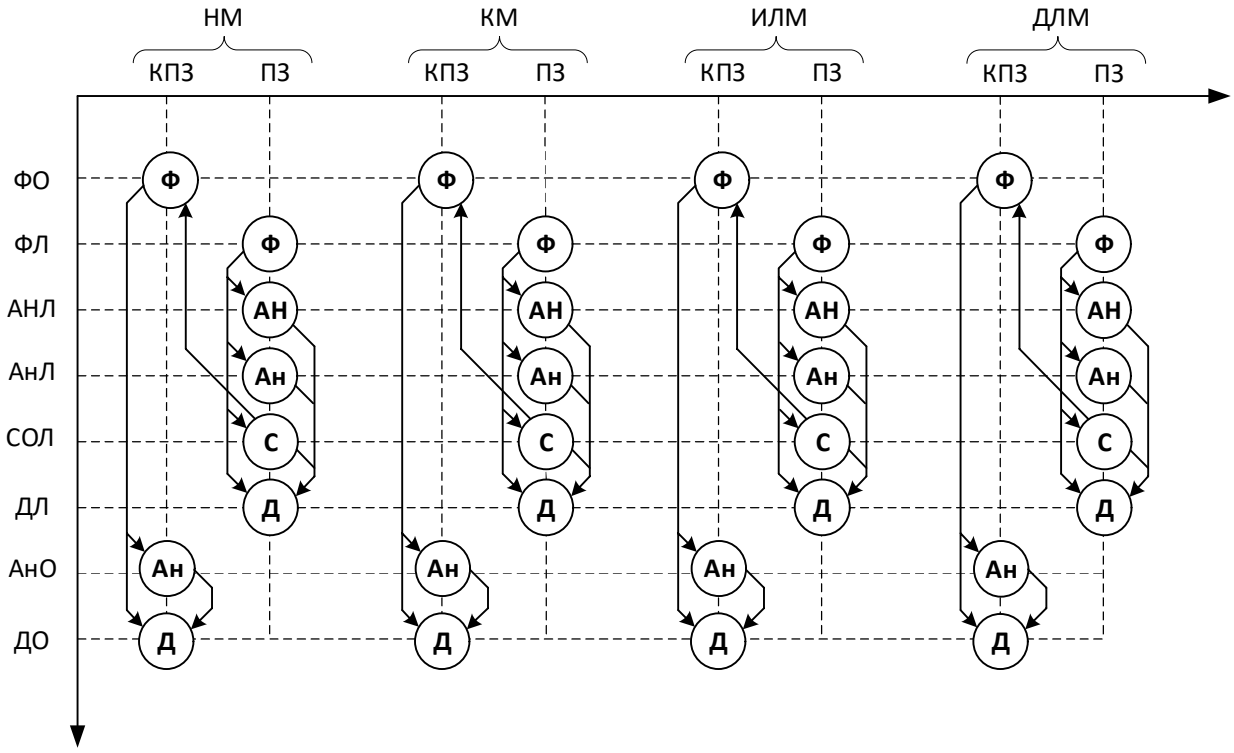


Рис. 2.10 Информационная взаимосвязь между этапами МАИТ

Структурная унификация моделей**.**

Каждая составляющая содержит набор однотипных составляющих, которые в процессе моделирования фиксируются в различных формах: в виде диаграмм и в виде спецификаций. [39] Структуры, которые описываются составляющими модели, формально представлены графами (рис. 2.11). Основываясь на методиках построения и описания моделей АС, выполненных в соответствии

с МАИТ, можно выделить 4 унифицированных вида структур моделей: основную структуру на элементах моделей, производную структуру на схемах элементов, функциональную структуру первого рода, функциональную структуру второго рода.

Унификация представления моделей.

Список формируемых в процессе моделирования документов приведен в таблице 2.2 . Сокращения к таблице приведены в таблице 2.3. [37] [40] [41]. Исходя из однотипности составляющих моделей и форм их представления, для каждого вида унифицированной структуры можно сформировать представление.

Таблица 2.2 Формы документирования моделей АС

Компонент	**НМ**		**КМ**		**ИЛМ**		**ДЛМ-Л**	
	граф.	*табл.*	*граф.*	*табл.*	*граф.*	*табл.*	*граф.*	*табл.*
Ст.	-	Форма B1 Форма B2 Форма B3	Диаграмма ОКС	Форма F1 Форма F2	Диаграмма ИЛС	Форма P1 Форма P2	Схема данных	Форма G1 Форма G2
Дн.	-	Форма A1 Форма A2	Диаграмма СПЗ-1	Форма F3 Форма F4	Диаграмма СПд-1	Форма P3 Форма P4	Диаграмма СЛД-1	Форма G3 Форма G4
Ф.	Диаграмма СПД	-	-	-	Диаграмма СПМ-1	Форма P5 Форма P6	Диаграмма СЛО-1	Форма G5 Форма G6
Инт.	-	-	-	-	-	-	Граф интерфейса	Форма G7 Форма G8
МвЦ.	Матричная диаграмма НМ	Форма D1 Форма D2 Форма D3	Матричная диаграмма КМ	Форма F6	Матричная диаграмма ИЛМ	Форма P8	Матричная диаграмма ДЛМ-Л	Форма G9

Таблица 2.3 Сокращения к таблице 2.2

Сокращение	**Обозначение**
Ст	Статическая составляющая
Дн	Динамическая составляющая
Ф.	Функциональная составляющая
Инт	Интерфейсная составляющая
Мвц	Модель в целом
НМ	Начальная модель
КМ	Концептуальная модель
ИЛМ	Инфологическая модель
ДЛМ-Л	Даталогическая модель на логическом уровне абстракции
СПД	Система предметных действий
СПЗ-1	Система предметных зависимостей 1-го рода
ОКС	Основная концептуальная структура
ИЛС	Основная инфологическая структура
СПд-1	Система предметных доступов 1-го рода
СПМ-1	Система предметных манипуляций 1-го рода
СЛД-1	Система логических доступов 1-го рода
СЛО-1	Система логических обработок 1-го рода

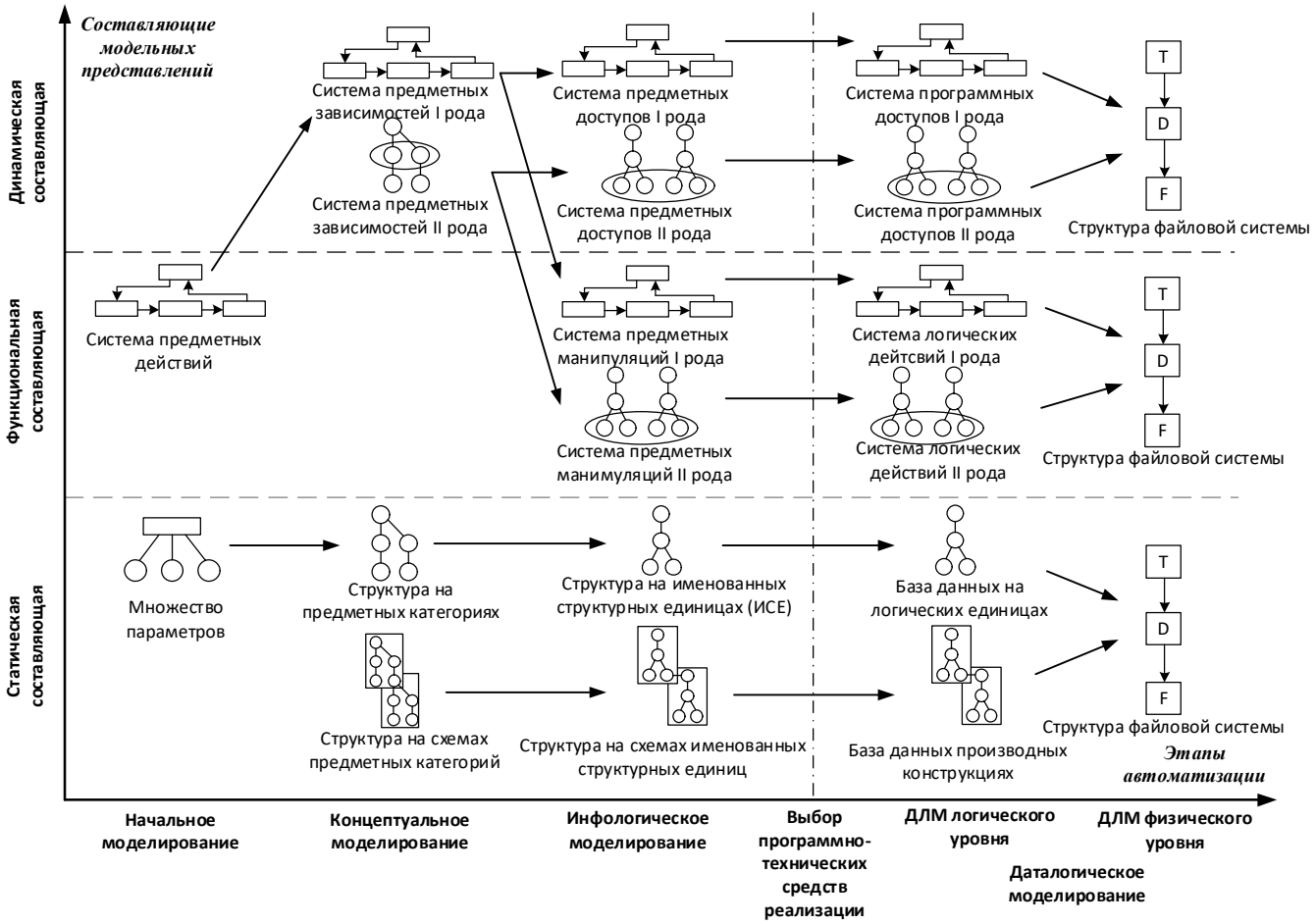


Рис. 2.11 Структура моделей автоматизированных систем на всех этапах их создания

Спецификации одних компонентов различных моделей схожи между собой, так как описывают графовые структуры. В общем виде их можно представить как таблицы элементов и таблицы связей. Каждая строка, описывающая элемент, представляет совокупность ключевого идентификатора – кода, наименования и основных свойств. Связи увязывают элементы на основе их кодов. Так как во всех моделях МАИТ используются связи одного вида, а свойства элементов различных моделей частично пересекаются, их представление можно унифицировать.

2.1.2 Разработка требований к ВИС

Выделенные особенности методологии автоматизации интеллектуального труда позволили сформулировать требования для визуальной интегрированной среды, её поддержки по следующим группам.

Требования к системе в целом:

-наличие следующих режимов функционирования: 1. режим моделирования, позволяющий

производить автоматизацию задач в соответствии с МАИТ, причем, в зависимости от типа задач можно выделить 2 подрежима: 1.1. ВИС обеспечивает моделирование проектно-конструкторских, управленческих задач различных предметных областей; 1.2 ВИС обеспечивает моделирование задач, направленных на собственное развитее, т.е. создание функциональных модулей будущих версий ВИС; 2. режим конфигурирования, реализующий плавный переход текущей версии ВИС на новую.

-возможность визуального моделирования автоматизируемых задач на этапах предпроектного обследования и проектирования *АС;*

-использование единых программных элементов для реализации процедур обработки однотипных аспектов различных моделей;

-осуществление информационной взаимосвязи между средствами поддержки различных этапов моделирования;

-предоставление возможности взаимодействия с внешними программными продуктами для экспорта результатов моделирования;

-обеспечение хранения версий реализации как отдельных элементов среды, так и всей среды в совокупности;

-поддержка процесса реконфигурирования системы путем изменения версий её элементов для постепенного наращивания функциональных возможностей ВИС.

Требования к модулям визуального формирования статических составляющих моделей:

-возможность размещения элементов в виде графических блоков на листе графического редактора, блоки размещаются на шести горизонтальных не пересекающихся уровнях;

-описание блока включает наименование элемента (размещается по центу блока) и его код (размещается в нижнем правом углу блока);

-обеспечение возможности установления связей (вертикальных и горизонтальных) между блоками в виде линий с указанием типа связи (символа, размещающегося по центру линии) и её направления;

- нижний уровень статической структуры, предоставляющий списочную информацию, должен вводиться на отдельной форме в табличном виде;

-наличие справочной документации пользователя.

Требования к модулям визуального формирования функциональной/динамической составляющих моделей:

-обеспечение возможности построения диаграмм из блоков (элементов) и линий (связей) на основе базовых и типовых алгоритмических конструкций в соответствии с правилами

моделирования;

-описание блока должно включать: наименование элемента по центру блока, код элемента в нижнем правом углу, код функции и структурный символ в нижнем левом углу;

-описание функциональной/динамической составляющих моделей должно формироваться на листах, листы включают в себя страницы, на которых размещаются диаграммы;

-блоки должны размещаться на страницах в ячейках, где каждый отдельный блок вписан в ячейку; при необходимости сложная структура может быть выполнена на нескольких страницах;

-в рамках страницы блоки размещаются ниже и правее родительского блока, в случае достижения правой границы листа или отсутствия свободных ячеек, дочерняя конструкция сдвигается левее, при этом размерность страницы – 6x6 ячеек;

-при необходимости редактирование объекта модели (изменение типа конструкции, добавление/удаление блока и т.д.), изменения контекста объекта редактирования должно сводиться к минимуму;

-наличие справочной документации пользователя.

Требования к модулям визуального формирования увязки составляющих моделей:

- обеспечение фиксации взаимосвязей элементов основной статической структуры и функциональной/динамической структуры первого рода за счет редактирования свойств блоков визуального редактора функциональной/динамической составляющей;

- обеспечение выполнения данной процедуры только для фиксации ролей параметров для каждого элементарного действия; на основании введенных данных формировать взаимосвязи промежуточных элементов различных составляющих и вычислять их структурные свойства автоматически;

-наличие справочной документации пользователя.

Требования к процедурам конфигурирования и развития среды:

-осуществление реконфигурации среды путем изменения её физического (файлового) состава и структуры;

-наличие автоматизированной процедуры анализа конфигурации на совместимость находящихся в ней файлов;

-наличие средств интеграции новых модулей среды: набора средств разработки (англ. software development kit, SDK) и документации к ним.

Сформированные требования к визуальной интегрированной среде позволяют перейти к разработке метода её моделирования на основе системного подхода.

2.2 Разработка формального описания визуальной интегрированной среды

2.2.1 Разработка формального описания визуальной интегрированной среды на основе системного подхода

Системный подход рассматривает исследуемый объект в виде системы, состоящей из совокупности компонентов – элементов и их взаимосвязей. По сути, множество элементов является разнородным, что подразумевает выделение подмножеств различных типов и описание взаимосвязей. Таким образом, функционирование визуальной интегрированной среды (ВИС) может быть представлено как система:

$$S_1^F =< Y^F, X^F, q(Y^F), q(X^F), R_{XY}^F >,$$

где Y^F – система процессов, выполняемых ВИС, X^F – совокупность объектов, участвующих в этих процессах, R_{XY}^F – множество взаимосвязей между объектами и процессами, $q(Y^F)$ – свойства процессов, $q(X^F)$ – свойства объектов.

Полученная система может быть рассмотрена в контексте окружающей среды – набора существующих объектов, которые взаимодействуют с системой. С учетом среды систему можно представить следующим образом:

$$S_2^F =< S_1^F, Z^F, q(Z^F), R_{XYZ}^F >,$$

где Z – среда, т.е. совокупность кадровых ***(SUB)*** и программно-технических ***(SW)*** ресурсов:

$$Z^F = SUB \bigcup SW,$$

$$q(Z^F) = q(SW) \bigcup q(SUB),$$

где ***q(Z), q(SW), q(SUB)*** – множества свойств соответствующих компонентов. [42] [43]

Элементы формального описания системы.

1. **Множество элементов** X^F представляет собой совокупность моделей различных видов и их составляющих, формируемых и обрабатываемых в процессе функционирования визуальной интегрированной среды.

$$X^F =< X_1, X_2, X_3, X_4, X_5, R_{12345}^X >,$$

где $X_1 \equiv X_{MP}(n)$ – начальное модельное представление n-ой предметной задачи, $X_2 \equiv X_{KP}(n)$ – концептуальное модельное представление, $X_2 \equiv X_{ILP}(n)$ – инфологическое модельное представление, $X_4 \equiv X_{DLPL}(n)$ – даталогическое представление логического уровня,

$X_5 \equiv X_{DLPF}(n)$ – даталогическое представление физического уровня, $R^X_{12345} = \{R^X_{12}, R^X_{23}, R^X_{34}, R^X_{45}\}$ – взаимосвязи между соответствующими модельными представлениями.

В работе рассматриваются начальное, концептуальное и инфологическое модельные представления ($\overline{X}$):

$$\overline{X}^F \subset X^F, \quad \overline{X}^F =< X_1(n), X_2(n), X_3(n), R^X_{123} >,$$

$$R^X_{123} = \{R^X_{12}, R^X_{23}\},$$

где R^X_{12} – взаимосвязи между начальным и концептуальным модельными представлениями, R^X_{23} – взаимосвязи между концептуальным и инфологическим модельными представлениями.

В общем случае ***n***-предметная задача может являться комплексом более простых задач

$$n = \left(\{n1, n2, \ldots\}, R(n)\right),$$

где $R(n)$ – структура задач в комплексе.

С учетом этого, на любом этапе обследования и проектирования АС выполняется формирование и обработка совокупности модельных представлений:

$$X_i(n) = \{X_i^0(n), X_i(n1), \ldots, X_i(nj), \ldots\},$$

где $X_i^0(n)$ – модельное представление на ***i***-м этапа ***n***-го комплекса предметных задач, $X_i(nj)$ – модельное представление отдельной ***j***-ой ПЗ, составляющей комплекс. В свою очередь ***nj***-ая также может быть представлена комплексом и подвергнута дальнейшей декомпозиции:

$$X_i(nj) = \{X_i^0(nj), X_i(nj1), \ldots, X_i(njl), \ldots\},$$ и т.д.

Также стоит учитывать, что любое модельное представление может существовать в нескольких версиях, из массива которых выбирается одна, утвержденная экспертом:

$$X_i(n) \subset \{X_{i1}(n), \ldots, X_{if}(n), \ldots\}$$

При этом модельные представления и для КПЗ, и для отдельных ПЗ имеют одинаковую структуру на каждом этапе, поэтому, вне зависимости, является ли ***n***-ая задача комплексом задач или локальной задачей:

$$X_i^0(n) \equiv X_i(n) =< M2_i(n), \overline{M3}_i(n), R_i^{23}(n) >,$$

где $M2_i(n)$ – модель объектного уровня ***n***-ой задачи на ***i***-ом этапе создания, $\overline{M3}_i(n) = \{\overline{M3}_i(nm)\}$ – совокупность моделей конкретного уровня для ***n***-ой задачи на ***i***-ом этапе создания, $\overline{M3}_i(nm)$ – модель конкретного уровня ***m***-ой реализации ***n***-ой задачи, $R_i^{23}(n)$ – связи моделей разных уровней абстрагирования.

Исходя из представленной структуры (рис. 2.11), объединённую модель объектного уровня ***n***-ой ПЗ ***i***-го этапа создания можно представить:

$$M2_i(n) =< EL2_i(n), ST2_i(n), DN2_i(n), FN2_i(n), R2_i^{SDF}(n) >,$$

где $EL2_i(n)$ – множество элементов модели ПЗ, $ST2_i(n)$ – множество статических отношений на элементах модели объектного уровня (статическая составляющая), $DN2_i(n)$ – множество динамических ограничений на элементах модели (динамическая составляющая), $FN2_i(n)$ – множество функциональных отношений на элементах модели (функциональная составляющая), $R2_{SDF}^{i}(n)$ – увязка составляющих. Каждая составляющая включает две структуры: основную и производную для статической составляющей, системы ограничений на элементы основной и производной статических структур, и системы функциональных отношений на элементах основной и производной статических структур (таблица 2.4). [44] [45] [46] [47]

С точки зрения графического описания конструкций для моделей объектного уровня ***n***-ой предметной задачи все составляющие могут быть формально представлены графами: статическими, динамическими, функциональными и содержательными (граф увязки составляющих):

$$M2_i(n) \to \{G_i^{tr}(n)\}, \quad G_i^{tr}(n) = (V_i^{tr}(n), E_i^{tr}(n)),$$

где индексы ***t, r*** определяют тип графа и его составляющих: $t \in \{'ST','DN','FN','R'\}$ обозначает вид составляющей, представленной графом, $r \in [1,2]$ определяет индекс структуры.

С точки зрения табличного описания конструкций каждая модель объектного уровня ***n***-ой предметной задачи фиксируется в виде спецификаций, которые могут быть формально представлены реляционными отношениями. Каждая структура модели может содержать спецификации для описания элементов структуры, для описания бинарных и/или тернарных связей на элементах структуры.

С точки зрения табличного описания конструкций каждая модель объектного уровня ***n***-ой предметной задачи фиксируется в виде спецификаций, которые могут быть формально представлены реляционными отношениями. Каждая структура модели может содержать спецификации для описания элементов структуры, бинарных и тернарных связей на элементах структуры:

$$M2(n) \to \{R_i^{tre}(n)\},$$

$$R_i^{rte}(n) \subset D_i^{tre1} \times D_i^{tre2} \times ... \times D_i^{trem}, \quad R_i^{tre}(n) = \{r_i^{tren} : r_i^{tren} = (d_i^{tre1-x}, d_i^{tre2-y}, ..., d_i^{trem-z})\},$$

где индексы ***t, r*** определяют вид спецификации, описывающую ***tr***-ю структуру, индекс ***e*** определяет индекс спецификации: ***e*** = 1 для таблицы с элементами, ***e*** = 2 для таблицы с бинарными связями, ***e*** = 3 для таблицы с тернарными связями, ***e***>3 для дополнительных спецификаций.

Таким образом определяются объекты множества ***X.***

Таблица 2.4 Состав моделей объектного уровня для проектирования АС

Компонент			Состав моделей по этапам (***i***)		
Вид	Стр-ра	Объект ст-ры	**НМ**	**КМ**	**ИЛМ**
ST2	**1**	*эл.* ***EL2***	Множество параметров: $\boldsymbol{X(n)=\{x_{kq}\}}$.	Предметные категории (ПК): $\boldsymbol{A(n)=\{a_{lm}\}}$.	Именованные структуры единицы (ИСЕ): $\boldsymbol{C(n)=\{c^{p}_{lm}\}}$.
		св.	-	Бинарные связи на ПК: $\boldsymbol{T2(n)=\{(a_{lm},a_{kn})\}}$; тернарные связи на ПК: $\boldsymbol{H2(n)=\{(a_{lm},a_{kn},a_{pt})\}}$.	Бинарные связи на ИСЕ: $\boldsymbol{D2(n)=\{(c^{r}_{il},c^{p}_{jm})\}}$; тернарные связи на ИСЕ: $\boldsymbol{E2(n)=\{(c^{r}_{il},c^{p}_{jm},c^{q}_{kn})\}}$.
	2	*эл.*	-	Схемы ПК: $\overline{\boldsymbol{H2(n)=\{h^{2}_{il}\}}}$.	Схемы ИСЕ: $\overline{\boldsymbol{E2(n)=\{e^{2}_{suil}\}}}$.
		св	-	Бинарные связи схем ПК: $\overline{\boldsymbol{Th2(n)=\{(h^{2}_{il},h^{2}_{jm})\}}}$.	Бинарные связи схем ИСЕ: $\overline{\boldsymbol{De2(n)=\{(e^{2}_{suil},e^{2}_{gvkn})\}}}$.
DN2	**1**	*эл.*	-	Предметные зависимости 1-го рода (ПЗ-1): 1-го рода003A $\overline{\boldsymbol{W(n)=\{w_{ps}\}}}$.	Предметные доступы 1-го рода (Пд-1): $\overline{\boldsymbol{R(n)=\{r_{sp}\}}}$.
		св	-	Бинарные связи ПЗ-1: $\overline{\boldsymbol{FV2(n)=\{(w_{ps},w_{jn})\}}}$.	Бинарные связи Пд-1: $\overline{\boldsymbol{QS2(n)=\{(r_{sp},r_{tr})\}}}$.
	2	*эл.*	-	Предметные зависимости 2-го рода (ПЗ-2): $\overline{\boldsymbol{Q2(n)=\{q^{2}_{st}\}}}$.	Предметные доступы второго рода (Пд-2): $\overline{\boldsymbol{N2(n)=\{n^{2}_{il}\}}}$.
		св	-	Бинарные связи ПЗ-2: $\overline{\boldsymbol{UQ2(n)=\{(q^{2}_{st},q^{2}_{rp})\}}}$.	Бинарные связи Пд-2: $\overline{\boldsymbol{QN2(n)=\{(n^{2}_{il},n^{2}_{jm})\}}}$.
FN2	**1**	*эл.*	Предметные действия (ПД): $\boldsymbol{Y(n)=\{y_{ps}\}}$.	-	Предметные манипуляции 1-го рода (ПМ-1): $\overline{\boldsymbol{X2}}=\{\overline{x^{2}_{sp}}\}$.
		св	Бинарные связи ПД: $\boldsymbol{FP2(n)=\{(y_{ps},y_{rt})\}}$.	-	Бинарные связи ПМ-1: $\overline{\boldsymbol{QX2(n)=\{(x^{2}_{il},x^{2}_{jm})\}}}$.
	2	*эл.*	-	-	Предметные манипуляции второго рода (ПМ-2): $\overline{\boldsymbol{Y2(n)=\{y^{2}_{il}\}}}$.
		св.	-	-	Бинарные связи ПМ-2: $\overline{\boldsymbol{QN2(n)=\{(n^{2}_{il},n^{2}_{jm})\}}}$.
R2	**1**	*св.*	Взаимосвязи параметров и ПД: $\boldsymbol{R_{XY}(n)\subset \subset X(n)\times...\times X(n)\times \times Y(n).}$	Взаимосвязи ПК и ПЗ-1: $\boldsymbol{R_{AW}(n)\subset A(n)\times...\times \times A(n)\times\overline{W}(n).}$	Взаимосвязи ИСЕ, Пд-1 и ПМ-1: $\boldsymbol{R_{CRX}(n)\subset C(n)\times...\times C(n)\times \times\overline{R}(n)\times\overline{X2}(n).}$
	2	*св.*	-	Взаимосвязи ПК и ПЗ-2 в рамках схем ПК: $\boldsymbol{R_{AHQ}(n)\subset A(n)\times...\times A(n)\times \times\overline{H}(n)\times...\times\overline{H}(n)\times\overline{Q2}(n).}$	Взаимосвязи ИСЕ, Пд-2, ПМ-2 в рамках схем ИСЕ: $\boldsymbol{R_{ENY}(n)\subset C(n)\times...\times C(n)\times \times E(n)\times...\times E(n)\times \times\overline{N2}(n)\times\overline{Y2}(n).}$

2. **Структура процесса** создания моделей Y формально может быть представлена следующим образом:

$$Y = (P, TP), \quad P = \bigcup_u P_u, \quad P_u \cap P_k = \varnothing,\ u \neq k,\ P_u = \{p_{uv}\},$$

$$TP \subset P \times P,\ TP = \{LP_u, LP_u P_w, LP_w / P_u\},$$

где P – множество действий, TP – множество связей между действиями, описывающими структуру действий. P_u – множество действий u-го уровня сложности, p_{uv} – v-ое действие на u-го уровне сложности в общем виде.

Множество связей типа «упорядочивание» определяет отношения порядка между действиями на одном уровне сложности:

$$LP_u = \{(p_{uv}, p_{uk}) \mid p_{uv}, p_{uk} \in P_u\}.$$

Связи типа «состав» определяют отношения иерархии между действиями на разных уровнях:

$$LP_u P_w = \{(p_{uv}, p_{wh}) \mid p_{uv} \in P_u, p_{wk} \in P_w\}.$$

Связи типа «компоновка» определяют отношения локального порядка между действиями на одном уровне сложности, но имеющими связь типа «состав» с общим вышестоящим действием:

$$LP_w / P_u = \{(p_{wh}, p_{wf}) \mid \exists p_{uv} \in P_u : (p_{uv}, p_{wh}) \in LP_u P_w \,\&\, (p_{uv}, p_{wf}) \in LP_u P_w\},\ w = u + 1$$

В структуре процесса создания АС можно выделить следующие уровни сложности (рис 2.12): процессов, этапов, процедур, операций, действий.

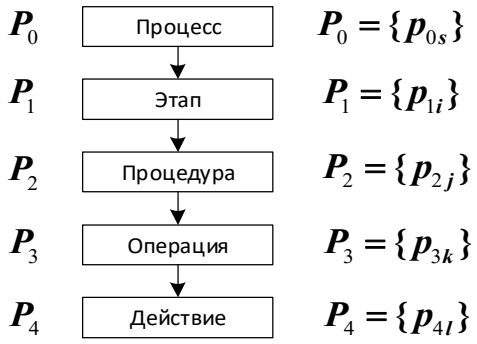


Рис. 2.12 Уровни сложности структуры процесса создания АС

Тогда, с учетом места каждого действия в общей структуре, элементы различных уровней декомпозиции формально можно представить так:

$\overline{p}_{1i(s)} = p_{1i} \,\&\, (p_{0s}, p_{1i})$ – i-ый этап s-го процесса,

$\overline{p}_{2j(si)} = p_{2j} \,\&\, (p_{1i}, p_{2j}) \,\&\, (p_{0s}, p_{1i})$ – j-ая процедура i-го этапа s-го процесса,

$\overline{p}_{3k(sij)} = p_{3k} \,\&\, (p_{2j}, p_{3k}) \,\&\, (p_{1i}, p_{2j}) \,\&\, (p_{0s}, p_{1i})$ – k-ая операция j-ой процедуры i-го этапа s-го процесса,

$\overline{p}_{4l(sijk)} = p_{4l} \,\&\, (p_{3k}, p_{4l}) \,\&\, (p_{2i}, p_{3k}) \,\&\, (p_{1j}, p_{2j}) \,\&\, (p_{0s}, p_{1i})$ – l-ое действие k-ой операции j-ой

процедуры ***i***-го этапа ***s***-го процесса.

Таким образом определены объекты множества Y как родословная ***l***-ого действия.

3. **Визуальная интегрированная среда является средством, т.е. программно-техническим ресурсом**, обеспечивающим создание автоматизированных систем различного назначения. При этом, в силу специфики предметной области разрабатываемой системы, процесс функционирования ВИС тождественен её процессу создания. Среда может формировать и обрабатывать модели в двух режимах: 1 – режим формирования модельных представлений «сторонних» автоматизированных систем; 2 – режим формирования модельных представлений будущих версий самой себя. Так как ВИС сама является автоматизированной системой, к её созданию может быть применена МАИТ. ВИС как программно-технический ресурс может быть смоделирована как модель реализации специализированной автоматизированной системы, т.е. в виде даталогических моделей – логической и физической. Поэтому моделирование визуальной интегрированной среды заключаются в формировании её взаимосвязанных моделей на трех уровнях абстракции, соответствующих её функционально-модульному описанию, компонентному описанию (логическое представление даталогической модели), файловому описанию (физическое представление даталогической модели), как показано на рис. 2.13: [48]

$$SW \equiv VIS = < VIS1, VIS2, VIS3, R^{123} >,$$

где $R^{123} = \{R^{12}, R^{23}\}$ – множество взаимосвязей между элементами моделей ВИС различных уровней абстракции.

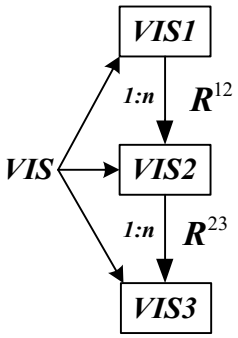


Рис. 2.13 Представление модели реализации ВИС с учетом уровней абстракции

Для каждого действия $\overline{p}_{4l(sijk)}$ может существовать множество версий средств поддержки, реализованных в виде функциональных модулей M^{sijkl}:

$$\overline{p}_{4l(sijk)} \rightarrow M^{sijkl} = \{m_f^{sijkl}\},$$

где m_f^{sijkl} – ***f***-ая версия функционального модуля.

Тогда, для любой операции $\overline{p}_{3k(sij)}$:

$$\overline{p}_{3k(sij)} \to M^{sijk} = \bigcup_{l} M^{sijkl},$$

где M^{sijk} – множество средств поддержки k-ой операции. Конфигурация операции $\overline{p}_{3k(sij)}$ формально определяется как множество версий средств поддержки каждого входящего в нее действия:

$$R_o^{sijk} \equiv r(M^{sijk}, \overline{o}(M^{sijk})) = \bigcup_{l} r(M^{sijkl}, \overline{f}(M^{sijkl})),$$

при этом $\overline{o}(M^{sijk}) = o(s,i,j,k)$ – индекс версии совокупности средств поддержки k-ой операции j-ой процедуры i-го этапа s-го процесса.

Для любой процедуры $\overline{p}_{2j(si)}$:

$$\overline{p}_{2j(si)} \to M^{sij} = \bigcup_{k} M^{sijk},$$

где M^{sij} – множество средств поддержки j-ой.

Для любого этапа $\overline{p}_{1i(s)}$:

$$\overline{p}_{1i(s)} \to M^{si} = \bigcup_{j} M^{sij},$$

где M^{si} – множество средств поддержки i-го этапа.

Для любого процесса $\overline{p}_{0s}$:

$$\overline{p}_{0s} \to M^{s} = \bigcup_{i} M^{si},$$

где M^{s} – множество средств поддержки s-го процесса.

Геометрическая интерпретация вышеописанных множеств приведена на рис. 2.14.

Конфигурация действия $\overline{p}_{4l(sijk)}$ определяется как ограничение на множество взаимосвязей версий средств поддержки M^{sijkl}, сопоставляющее каждому действию только одну версию средств поддержки, поэтому f-ая версия конфигурации:

$$R_f^{sijkl} \equiv r(M^{sijkl}, \overline{f}) = m_f^{sijkl}, m_f^{sijkl} \in M^{sijkl},$$

где $\overline{f} = f(s,i,j,k,l)$ – индекс f-ой версия средства поддержки l-го действия k-ой операции j-ой процедуры i-го этапа s-го процесса.

Конфигурация процедуры формально описывается как интеграция конфигураций всех операций процедуры или как ограничение на множество средств поддержки j-ой процедуры i-го этапа s-го процесса:

$$R_p^{sij} \equiv r(M^{sij}, \overline{p}) = \bigcup_{k} r(M^{sijk}, \overline{f}),$$

где $\bar{q} = q(s,i,j)$ – индекс версии конфигурации ***j***-ой процедуры ***i***-го этапа ***s***-го процесса.

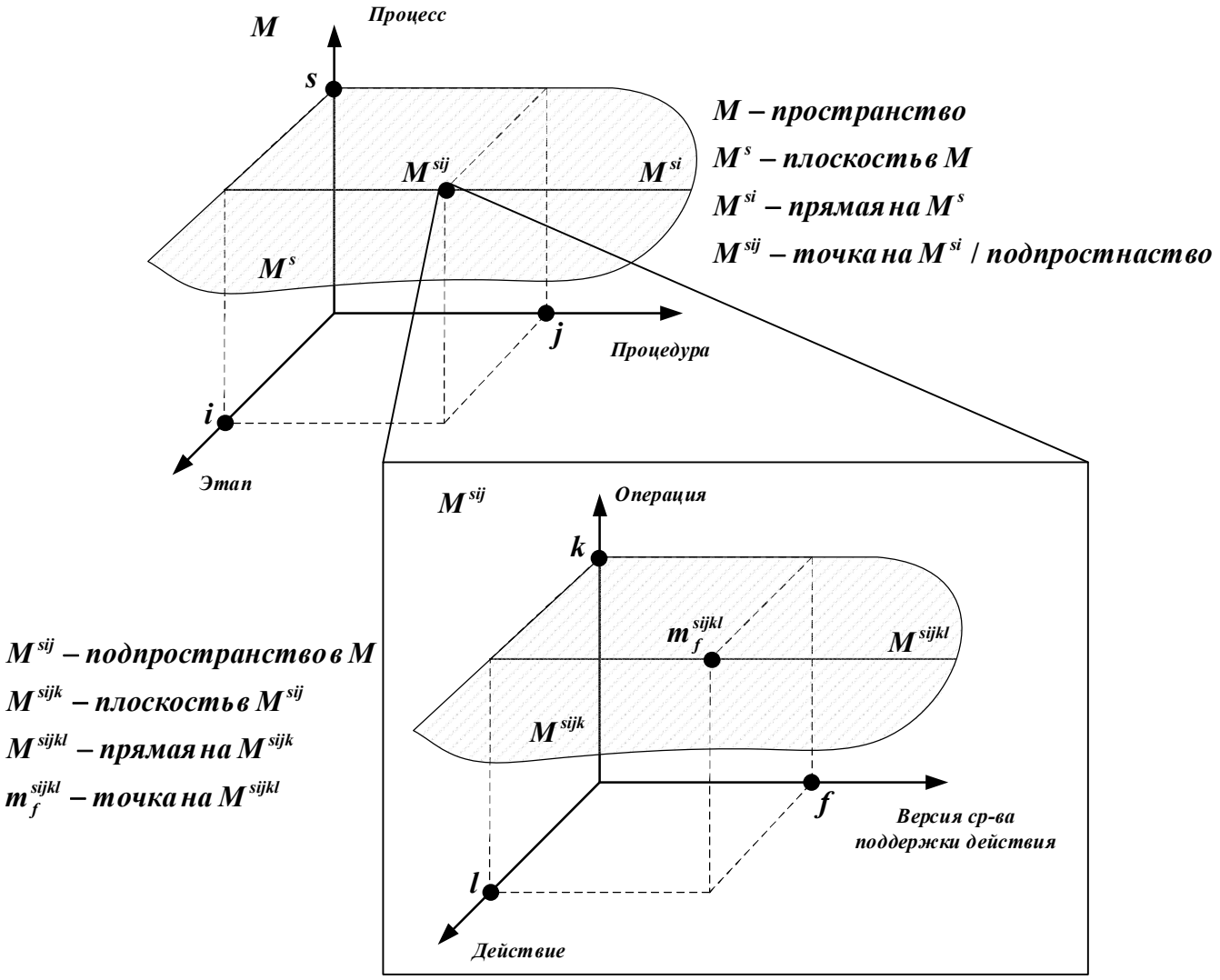


Рис. 2.14 Геометрическая интерпретация организации множества функциональных модулей ВИС

Формально конфигурация этапа определяется как интеграция конфигураций всех процедур этапа:

$$R_t^{si} \equiv r(M^{si}, \bar{t}) = \bigcup_j r(M^{sij}, \bar{q}),$$

где $\bar{t} = t(s,i)$ – индекс версии конфигурации ***i***-го этапа ***s***-го процесса.

$$R_u^s \equiv r(M^s, \bar{u}) = \bigcup_i r(M^{si}, \bar{t}),$$

$\bar{u} = u(s)$ – индекс конфигурации **s**-го процесса

$$R_v \equiv r(M\ , \bar{v}) = \bigcup_s r(M^{si}, \bar{u})$$

$\bar{v} \in \{v_1, v_2, ..., v_d, ...\}$ – множество конфигураций ВИС для всех процессов.

Исходя из вышесказанного, визуальную интегрированную среду можно представить, как

систему функциональных модулей:

$$VIS1 =< M, R1 > .$$

где M – множество функциональных модулей, $R1 \subset M \times M ... \times M = \{R_v\}$ – множество взаимосвязей между функциональными модулями.

Каждую версию функционального модуля m_f^{sijkl} поддержки действия $\overline{p}_{4l(sijk)}$ можно представить как совокупность компонентов – информационных, интерфейсных и функциональных (рис 2.15).

$$m_f^{sijkl} \rightarrow < INF_f^{sijkl}, FUN_f^{sijkl}, INT_f^{sijkl}, R2_f^{sijkl} >,$$

где INF_f^{sijkl} – версия информационного компонента функционального модуля m_f^{sijkl}, FUN_f^{sijkl} – версия функционального компонента, INT_f^{sijkl} – версия интерфейсного компонента, $R2_f^{sijkl}$ – взаимосвязи между версиями компонентов в рамках одной версии модуля m_f^{sijkl}.

При этом взаимосвязи между уровнем функционального представления ВИС и уровнем представления в виде логических компонентов формируются следующим образом:

$$R_{sifklf}^{12-INF} = \left\{(m_f^{sijkl}, INF_f^{sijkl})\right\}, R_{sifklf}^{12-FUN} = \left\{(m_f^{sijkl}, FUN_f^{sijkl})\right\}, R_{sifklf}^{12-INT} = \left\{(m_f^{sijkl}, INT_f^{sijkl})\right\}.$$

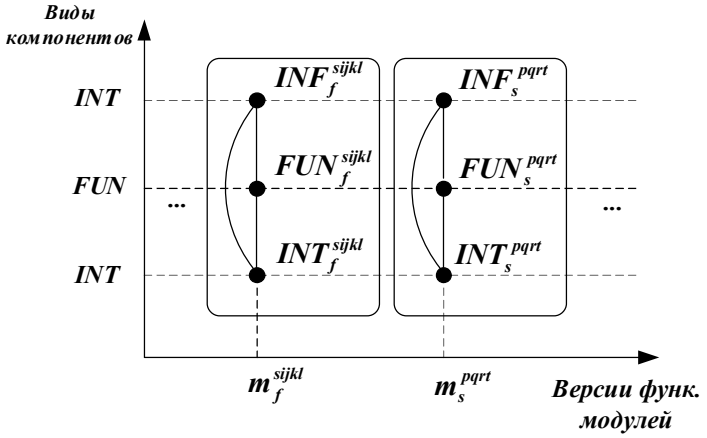


Рис. 2.15 Графическое представления версии функционального модуля как совокупности компонентов

Тогда совокупность версий функциональных модулей M^{sijkl}, реализующих действие $\overline{p}_{4l(sijk)}$, можно представить так:

$$M^{sijkl} \rightarrow < INF^{sijkl}, FUN^{sijkl}, INT^{sijkl}, R2^{sijkl} >,$$

$$R_{sifkl}^{12-INF} = \bigcup_f R_{sifklf}^{12-INF}, R_{sifkl}^{12-FUN} = \bigcup_f R_{sifklf}^{12-FUN}, R_{sifkl}^{12-INT} = \bigcup_f R_{sifklf}^{12-INT}$$

Для совокупности модулей M^{sijk}, реализующих k-ую операцию $\overline{p}_{3k(sij)}$:

$$M^{sijk} \rightarrow < INF^{sijk}, FUN^{sijk}, INT^{sijk}, R2^{sijk} >,$$

$$R_{sifk}^{12-INF} = \bigcup_{l} R_{sifkl}^{12-INF}, R_{sifk}^{12-FUN} = \bigcup_{l} R_{sifkl}^{12-FUN}, R_{sifk}^{12-INT} = \bigcup_{l} R_{sifkl}^{12-INT}.$$

Для совокупности модулей M^{sij}, реализующих j-ую процедуру $\overline{p}_{2j(si)}$:

$$M^{sij} \rightarrow \ < INF^{sij}, FUN^{sij}, INT^{sij}, R2^{sij} >,$$

$$R_{sif}^{12-INF} = \bigcup_{k} R_{sifk}^{12-INF}, R_{sif}^{12-FUN} = \bigcup_{k} R_{sifk}^{12-FUN}, R_{sif}^{12-INT} = \bigcup_{k} R_{sifk}^{12-INT}.$$

Для совокупности модулей M^{si}, реализующих i-ый этап $\overline{p}_{1i(s)}$:

$$M^{si} \rightarrow \ < INF^{si}, FUN^{si}, INT^{si}, R2^{si} >.$$

$$R_{si}^{12-INF} = \bigcup_{j} R_{sij}^{12-INF}, R_{si}^{12-FUN} = \bigcup_{j} R_{sij}^{12-FUN}, R_{si}^{12-INT} = \bigcup_{j} R_{sij}^{12-INT}.$$

Для совокупности модулей M^{s}, реализующих s-ый процесс p_{0s}:

$$M^{s} \rightarrow \ < INF^{s}, FUN^{s}, INT^{s}, R2^{s} >.$$

$$R_{s}^{12-INF} = \bigcup_{i} R_{si}^{12-INF}, R_{si}^{12-FUN} = \bigcup_{j} R_{sij}^{12-FUN}, R_{si}^{12-INT} = \bigcup_{j} R_{sij}^{12-INT}.$$

Для совокупности модулей M, реализующих все процессы:

$$M \rightarrow \ < INF, FUN, INT, R2 >.$$

$$R^{12-INF} = \bigcup_{s} R_{s}^{12-INF}, R^{12-FUN} = \bigcup_{s} R_{s}^{12-FUN}, R^{12-INT} = \bigcup_{s} R_{s}^{12-INT}.$$

$$R^{12} = \{R^{12-INF}, R^{12-FUN}, R^{12-INT}\}.$$

Таким образом, на втором уровне абстракции – уровне логических компонентов, визуальную интегрированную среду можно представить как:

$$VIS2 =< INF, FUN, INT, R2 >.$$

Каждый логический компонент можно представить как совокупность физических элементов файловой системы (ФС): томов, директорий, файлов.

Информационный компонент INF_f^{sijkl} f-ой версии модуля m_f^{sijkl}, реализующий действие $\overline{p}_{4l(sijk)}$, на уровне файловой системы представляется следующим образом (рис. 2.16):

$$INF_f^{sijkl} \rightarrow < TI_f^{sijkl}, DI_f^{sijkl}, FI_f^{sijkl}, RI3_f^{sijkl} >,$$

где TI_f^{sijkl} – множество томов информационного компонента версии модуля или информационных томов, DI_f^{sijkl} – множество информационных директорий, FI_f^{sijkl} – множество информационных файлов, $RI3_f^{sijkl}$ – взаимосвязи между информационными томами, директориями и файлами.

$$RI3_f^{sijkl} = RI3_{1f}^{sijkl} \cup RI3_{2f}^{sijkl}$$

$$RI3_{1f}^{sijkl} = (RI3_{Tf}^{sijkl}, RI3_{Df}^{sijkl}, RI3_{Ff}^{sijkl}, RI3_{TDf}^{sijkl}, RI3_{DDf}^{sijkl}, RI3_{DFf}^{sijkl}),$$

$$RI3_{2f}^{sijkl} \subset FI_f^{sijkl} \times FI_f^{sijkl};$$

При этом взаимосвязи между уровнями логических компонентов и элементов файловой системы для информационного компонента формируются так:

$$RI_{sijklf}^{23-T} = \left\{(INF_f^{sijkl}, TI_f^{sijkl})\right\}, RI_{sijklf}^{23-D} = \left\{(INF_f^{sijkl}, DI_f^{sijkl})\right\}, RI_{sijklf}^{23-F} = \left\{(INF_f^{sijkl}, FI_f^{sijkl})\right\}.$$

Для информационного компонента множества версий функциональных модулей M^{sijkl}, реализующих действие $\overline{p}_{4l(sijk)}$:

$$INF^{sijkl} \rightarrow\!\!< TI^{sijkl}, DI^{sijkl}, FI^{sijkl}, RI3^{sijkl} >,$$

$$RI_{sijkl}^{23-T} = \bigcup_f RI_{sijklf}^{23-T}, RI_{sijkl}^{23-D} = \bigcup_f RI_{sijklf}^{23-D}, RI_{sijkl}^{23-F} = \bigcup_f RI_{sijklf}^{23-F};$$

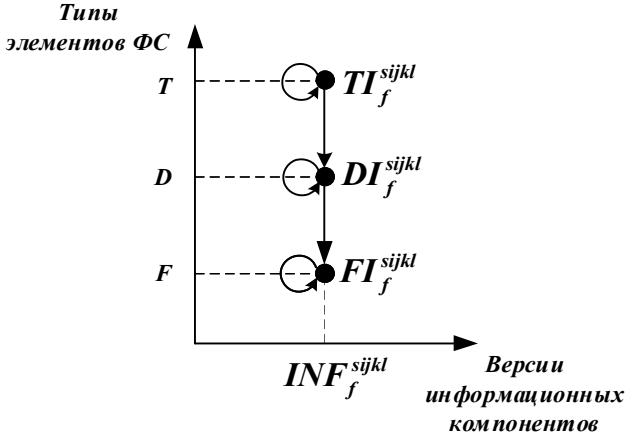


Рис. 2.16 Структура информационного компонента на физическом уровне

для информационного компонента множества функциональных модулей M^{sijk}, реализующих ***k***-ую операцию $\overline{p}_{3k(sij)}$:

$$INF^{sijk} \rightarrow\!\!< TI^{sijk}, DI^{sijk}, FI^{sijk}, RI3^{sijk} >;$$

$$RI_{sijk}^{23-T} = \bigcup_l RI_{sijkl}^{23-T}, RI_{sijk}^{23-D} = \bigcup_l RI_{sijkl}^{23-D}, RI_{sijk}^{23-F} = \bigcup_l RI_{sijkl}^{23-F};$$

для информационного компонента множества функциональных модулей M^{sij}, реализующих ***j***-ую процедуру $\overline{p}_{2j(si)}$:

$$INF^{sij} \rightarrow\!\!< TI^{sij}, DI^{sij}, FI^{sij}, RI3^{sij} >;$$

$$RI_{sij}^{23-T} = \bigcup_k RI_{sijk}^{23-T}, RI_{sij}^{23-D} = \bigcup_k RI_{sijk}^{23-D}, RI_{sij}^{23-F} = \bigcup_k RI_{sijk}^{23-F};$$

для информационного компонента множества функциональных модулей M^{si}, реализующих ***i***-ый этап $\overline{p}_{1i(s)}$:

$$INF^{si} \rightarrow\!\!< TI^{si}, DI^{si}, FI^{si}, RI3^{si} >;$$

$$RI_{si}^{23-T} = \bigcup_j RI_{sij}^{23-T},\ RI_{si}^{23-D} = \bigcup_j RI_{sij}^{23-D},\ RI_{si}^{23-F} = \bigcup_j RI_{sij}^{23-F};$$

для информационного компонента множества функциональных модулей M^s, реализующих s-ый процесс p_{0s}:

$$INF^s \rightarrow\!\prec TI^s, DI^s, FI^s, RI3^s \succ.$$

$$RI_s^{23-T} = \bigcup_i RI_{si}^{23-T},\ RI_s^{23-D} = \bigcup_i RI_{si}^{23-D},\ RI_s^{23-F} = \bigcup_i RI_{si}^{23-F};$$

Исходя из вышесказанного, информационный компонент ВИС формально определяется так:

$$INF \rightarrow\!\prec TI, DI, FI, RI3 \succ.$$

$$RI^{23-T} = \bigcup_s RI_s^{23-T},\ RI^{23-D} = \bigcup_s RI_s^{23-D},\ RI^{23-F} = \bigcup_s RI_s^{23-F};$$

$$RI^{23} = \{RI^{23-T}, RI^{23-D}, RI^{23-F}\}.$$

Функциональный компонент FUN_f^{sijkl} f-ой версии модуля m_f^{sijkl}, реализующий действие $\overline{p}_{4l(sijk)}$, на уровне файловой системы представляется так (рис. 2.17):

$$FUN_f^{sijkl} \rightarrow\!\prec TD_f^{sijkl}, DD_f^{sijkl}, FD_f^{sijkl}, TP_f^{sijkl}, DP_f^{sijkl}, FP_f^{sijkl},\quad RDP3_f^{sijkl} \succ,$$

где TD_f^{sijkl} – множество томов доступов функционального компонента модуля, DD_f^{sijkl} – множество директорий доступов, FD_f^{sijkl} – множество файлов доступов, TP_f^{sijkl} – множество томов процедур, DP_f^{sijkl} – множество директорий процедур, FP_f^{sijkl} – множество файлов процедур, $RDP3_f^{sijkl}$ – взаимосвязи между элементами вышеперечисленных множеств.

При этом взаимосвязи между уровнями логических компонентов и элементов файловой системы для функционального компонента формируются так:

$$RD_{sijklf}^{23-T} = \left\{(FUN_f^{sijkl}, TD_f^{sijkl})\right\},\ RD_{sijklf}^{23-D} = \left\{(FUN_f^{sijkl}, DD_f^{sijkl})\right\},\ RD_{sijklf}^{23-F} = \left\{(FUN_f^{sijkl}, FD_f^{sijkl})\right\},$$

$$RP_{sijklf}^{23-T} = \left\{(FUN_f^{sijkl}, TP_f^{sijkl})\right\},\ RP_{sijklf}^{23-D} = \left\{(FUN_f^{sijkl}, DP_f^{sijkl})\right\},\ RP_{sijklf}^{23-F} = \left\{(FUN_f^{sijkl}, FP_f^{sijkl})\right\}.$$

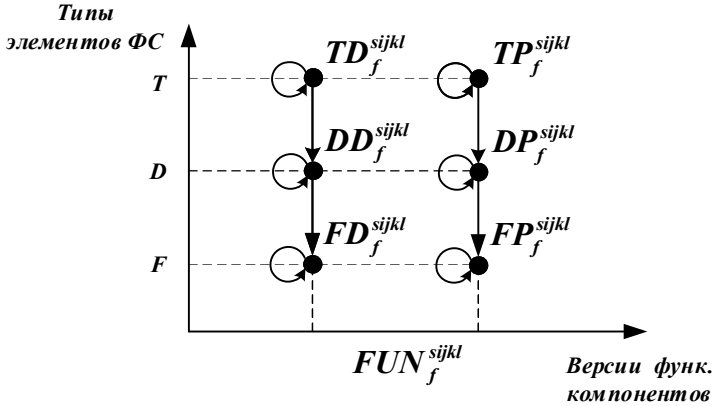


Рис. 2.17 Структура функционального компонента на физическом уровне

Для функционального компонента множества версий функциональных модулей M^{sijkl}, реализующих действие $\overline{p}_{4l(sijk)}$:

$$FUN^{sijkl} \rightarrow\!\!< TD^{sijkl}, DD^{sijkl}, FD^{sijkl}, TP^{sijkl}, DP^{sijkl}, FP^{sijkl}, \quad RDP3^{sijkl} >;$$

$$RD_{sijkl}^{23-T} = \bigcup_{f} RD_{sijklf}^{23-T}, RD_{sijkl}^{23-D} = \bigcup_{f} RD_{sijklf}^{23-D}, RD_{sijkl}^{23-F} = \bigcup_{f} RD_{sijklf}^{23-F};$$

$$RP_{sijkl}^{23-T} = \bigcup_{f} RP_{sijklf}^{23-T}, RP_{sijkl}^{23-D} = \bigcup_{f} RP_{sijklf}^{23-D}, RP_{sijkl}^{23-F} = \bigcup_{f} RP_{sijklf}^{23-F};$$

для функционального компонента множества функциональных модулей M^{sijk}, реализующих k-ую операцию $\overline{p}_{3k(sij)}$:

$$FUN^{sijk} \rightarrow\!\!< TD^{sijk}, DD^{sijk}, FD^{sijk}, TP^{sijk}, DP^{sijk}, FP^{sijk}, \quad RDP3^{sijk} >;$$

$$RD_{sijk}^{23-T} = \bigcup_{l} RD_{sijkl}^{23-T}, RD_{sijk}^{23-D} = \bigcup_{l} RD_{sijkl}^{23-D}, RD_{sijk}^{23-F} = \bigcup_{l} RD_{sijkl}^{23-F};$$

$$RP_{sijk}^{23-T} = \bigcup_{l} RP_{sijkl}^{23-T}, RP_{sijk}^{23-D} = \bigcup_{l} RP_{sijkl}^{23-D}, RP_{sijk}^{23-F} = \bigcup_{l} RP_{sijkl}^{23-F};$$

для функционального компонента множества функциональных модулей M^{sij}, реализующих j-ую процедуру $\overline{p}_{2j(si)}$:

$$FUN^{sij} \rightarrow\!\!< TD^{sij}, DD^{sij}, FD^{sij}, TP^{sij}, DP^{sij}, FP^{sij}, \quad RDP3^{sij} >$$

$$RD_{sij}^{23-T} = \bigcup_{k} RD_{sijk}^{23-T}, RD_{sij}^{23-D} = \bigcup_{k} RD_{sijk}^{23-D}, RD_{sij}^{23-F} = \bigcup_{k} RD_{sijk}^{23-F};$$

$$RP_{sij}^{23-T} = \bigcup_{k} RP_{sijk}^{23-T}, RP_{sij}^{23-D} = \bigcup_{k} RP_{sijk}^{23-D}, RP_{sij}^{23-F} = \bigcup_{k} RP_{sijk}^{23-F};$$

для функционального компонента множества функциональных модулей M^{si}, реализующих i-ый этап $\overline{p}_{1i(s)}$:

$$FUN^{si} \rightarrow\!\!< TD^{si}, DD^{si}, FD^{si}, TP^{si}, DP^{si}, FP^{si}, \quad RDP3^{si} >;$$

$$RD_{s}^{23-T} = \bigcup_{j} RD_{si}^{23-T}, RD_{si}^{23-D} = \bigcup_{j} RD_{sij}^{23-D}, RD_{si}^{23-F} = \bigcup_{j} RD_{sij}^{23-F};$$

$$RP_{si}^{23-T} = \bigcup_{j} RP_{sij}^{23-T}, RP_{si}^{23-D} = \bigcup_{j} RP_{sij}^{23-D}, RP_{si}^{23-F} = \bigcup_{j} RP_{sij}^{23-F};$$

для функционального компонента множества функциональных модулей M^{s}, реализующих s-ый процесс p_{0s}:

$$FUN^{s} \rightarrow\!\!< TD^{s}, DD^{s}, FD^{s}, TP^{s}, DP^{s}, FP^{s}, \quad RDP3^{s} >.$$

$$RD_{s}^{23-T} = \bigcup_{i} RD_{si}^{23-T}, RD_{s}^{23-D} = \bigcup_{i} RD_{si}^{23-D}, RD_{s}^{23-F} = \bigcup_{i} RD_{si}^{23-F};$$

$$RP_s^{23-T} = \bigcup_i RP_{si}^{23-T},\ RP_s^{23-D} = \bigcup_i RP_{si}^{23-D},\ RP_s^{23-F} = \bigcup_i RP_{si}^{23-F};$$

Следовательно, функциональный компонент ВИС определяется следующим образом:

$$FUN \rightarrow\!\!\prec TD, DD, FD, TP, DP, FP, RDP3 >.$$

$$RD^{23-T} = \bigcup_s RD_s^{23-T},\ RD^{23-D} = \bigcup_s RD_s^{23-D},\ RD^{23-F} = \bigcup_s RD_s^{23-F};$$

$$RP^{23-T} = \bigcup_s RP_s^{23-T},\ RP^{23-D} = \bigcup_s RP_s^{23-D},\ RP^{23-F} = \bigcup_s RP_s^{23-F};$$

$$RF^{23} = \{RD^{23-T}, RD^{23-D}, RD^{23-F}, RP^{23-T}, RP^{23-D}, RP^{23-F}\}$$

Интерфейсный компонент INT_f^{sijkl} f-ой версии модуля m_f^{sijkl}, реализующий действие $\overline{p}_{4l(sijk)}$, на уровне файловой системы представляется следующим образом (рис. 2.18):

$$INT_f^{sijkl} \rightarrow\!\!\prec IT_f^{sijkl}, ID_f^{sijkl}, IF_f^{sijkl}, IR3_f^{sijkl} >,$$

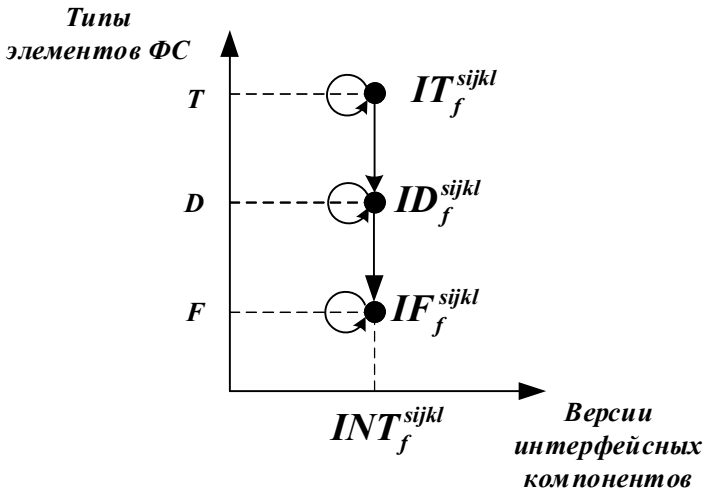


Рис. 2.18 Структура интерфейсного компонента на физическом уровне

где IT_f^{sijkl} – множество томов интерфейсного компонента версии модуля или интерфейсных томов, ID_f^{sijkl} – множество интерфейсных директорий, IF_f^{sijkl} – множество интерфейсных файлов, $IR3_f^{sijkl}$ – взаимосвязи между интерфейсными томами, директориями и файлами.

При этом взаимосвязи между уровнями логических компонентов и элементов файловой системы для интерфейсного компонента формируются так:

$$IR_{sijkl}^{23-T} = \bigcup_f IR_{sijklf}^{23-T},\ IR_{sijkl}^{23-D} = \bigcup_f IR_{sijklf}^{23-D},\ IR_{sijkl}^{23-F} = \bigcup_f IR_{sijklf}^{23-F};$$

Для интерфейсного компонента множества версий функциональных модулей M^{sijkl}, реализующих действие $\overline{p}_{4l(sijk)}$:

$$INT^{sijkl} \rightarrow\!\!\prec IT^{sijkl}, ID^{sijkl}, IF^{sijkl}, IR3^{sijkl} >;$$

$$IR_{sijkl}^{23-T} = \bigcup_{f} IR_{sijklf}^{23-T}, IR_{sijkl}^{23-D} = \bigcup_{f} IR_{sijklf}^{23-D}, IR_{sijkl}^{23-F} = \bigcup_{f} IR_{sijklf}^{23-F};$$

для интерфейсного компонента множества функциональных модулей M^{sijk}, реализующих ***k***-ую операцию $\overline{p}_{3k(sij)}$:

$$INT^{sijk} \rightarrow\!\!\prec IT^{sijk}, ID^{sijk}, IF^{sijk}, IR3^{sijk} >;$$

$$IR_{sijk}^{23-T} = \bigcup_{l} IR_{sijkl}^{23-T}, IR_{sijk}^{23-D} = \bigcup_{l} IR_{sijkl}^{23-D}, IR_{sijk}^{23-F} = \bigcup_{l} IR_{sijkl}^{23-F};$$

для интерфейсного компонента множества функциональных модулей M^{sij}, реализующих ***j***-ую процедуру $\overline{p}_{2j(si)}$:

$$INT^{sij} \rightarrow\!\!\prec IT^{sij}, ID^{sij}, IF^{sij}, IR3^{sij} >;$$

$$IR_{sij}^{23-T} = \bigcup_{k} IR_{sijk}^{23-T},$$

$$IR_{sij}^{23-D} = \bigcup_{k} IR_{sijk}^{23-D}, IR_{sij}^{23-F} = \bigcup_{k} IR_{sijk}^{23-F};$$

для интерфейсного компонента множества функциональных модулей M^{si}, реализующих ***i***-ый этап $\overline{p}_{1i(s)}$:

$$INT^{si} \rightarrow\!\!\prec IT^{si}, ID^{si}, IF^{si}, IR3^{si} >;$$

$$IR_{si}^{23-T} = \bigcup_{j} IR_{sij}^{23-T},$$

$$IR_{si}^{23-D} = \bigcup_{j} IR_{sij}^{23-D}, IR_{si}^{23-F} = \bigcup_{j} IR_{sij}^{23-F};$$

для интерфейсного компонента множества функциональных модулей M^{s}, реализующих ***s***-ый процесс p_{0s}:

$$INT^{s} \rightarrow\!\!\prec IT^{s}, ID^{s}, IF^{s}, IR3^{s} >;$$

$$IR_{s}^{23-T} = \bigcup_{i} IR_{si}^{23-T},$$

$$IR_{s}^{23-D} = \bigcup_{i} IR_{si}^{23-D}, IR_{s}^{23-F} = \bigcup_{i} IR_{si}^{23-F};$$

Тем самым, интерфейсный компонент ВИС формально можно определить так:

$$INT \rightarrow\!\!\prec IT, ID, IF, IR3 >.$$

$$IR^{23-T} = \bigcup_{s} IR_{s}^{23-T}, \; IR^{23-D} = \bigcup_{s} IR_{s}^{23-D},$$

$$IR^{23-F} = \bigcup_{s} IR_{s}^{23-F}; \; IR^{23} = \{IR^{23-T}, IR^{23-D}, IR^{23-F}\}.$$

Множество всех томов ВИС ***T*** представляет из себя совокупность информационных томов,

томов доступов, томов процедур и томов интерфейсов:

$$\boldsymbol{T = TI \cup TD \cup TP \cup IT}.$$

Множество всех директорий ВИС $\boldsymbol{D}$ представляет из себя совокупность информационных директорий, директорий доступов, директорий процедур и директорий интерфейсов:

$$\boldsymbol{D = DI \cup DD \cup DP \cup ID}.$$

Множество всех файлов ВИС $\boldsymbol{F}$ представляет из себя совокупность информационных файлов, файлов доступов, файлов процедур и файлов интерфейсов:

$$\boldsymbol{F = FI \cup FD \cup FP \cup IF}.$$

Взаимосвязи между всеми элементами файловой системы ВИС $\boldsymbol{R3}$ описывают две различные структуры ВИС (рис 2.19):

$$\boldsymbol{R3 = R3_1 \cup R3_2}.$$

Множество $\boldsymbol{R3_1}$ описывает файловую структуру хранения ВИС, определяя иерархические связи между томами, директориями и файлами:

$$\boldsymbol{R3_1 = RI3 \cup RDP3 \cup IR3}.$$

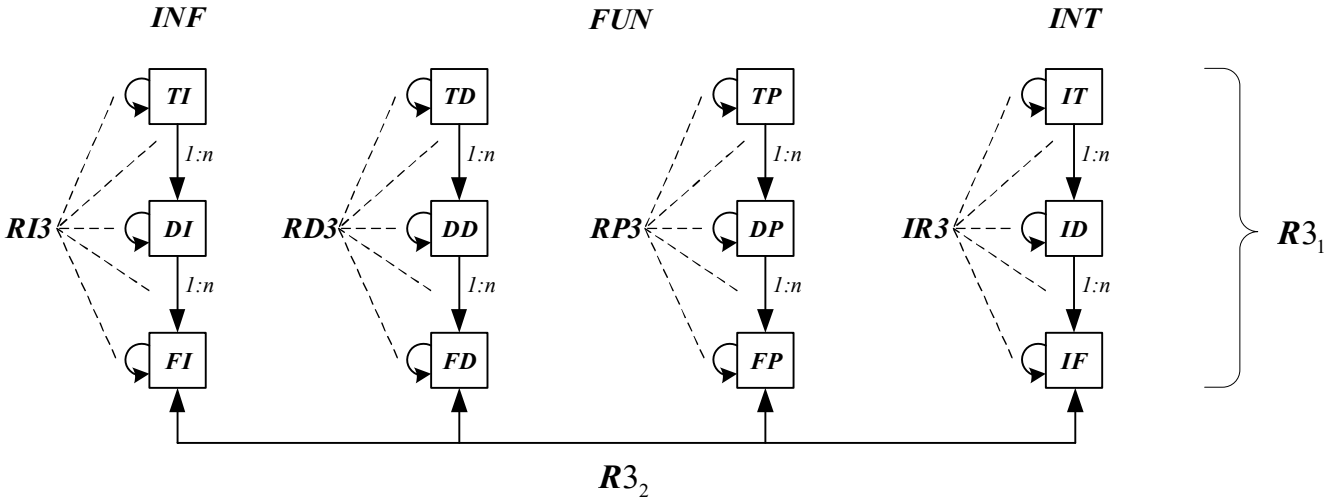


Рис. 2.19 Все виды взаимосвязей между различными элементами файловой системы

Множество $\boldsymbol{R3_2}$ описывает структуру функционирования ВИС, определяя её как набор взаимодействий между файлами различных типов:

$$\boldsymbol{R3_2 \subset F \times F}.$$

Исходя из вышеизложенного, модель ВИС на третьем уровне абстракции файловой системы (физическом), формально определена так:

$$\boldsymbol{VIS3 = < T, D, F, R3 >}.$$

2.2.2 Разработка концептуальной модели процесса функционирования визуальной интегрированной среды

Для определения свойств процессов, объектов и элементов программного инструментария в рамках предлагаемой модели ВИС необходимо её содержательное описание. Поскольку процессы создания АС и функционирования ВИС – совпадающие, то содержательным описанием функционирования и развития ВИС является концептуальная модель этапов создания АС.

Автоматизация формирования и обработки составляющих моделей предметных задач выполняется в соответствии с методологией автоматизации интеллектуального труда. Для каждого программного модуля, входящего состав ВИС, может быть сформирована совокупность моделей на каждом этапе их создания (рис 2.20). При этом, основными моделями будут являться: концептуальная модель, позволяющая выявить и структурировать знания о предметной области; даталогическая, отражающая организацию выбранной программно-технической среды для хранения и обработки выявленной ранее системы знаний.

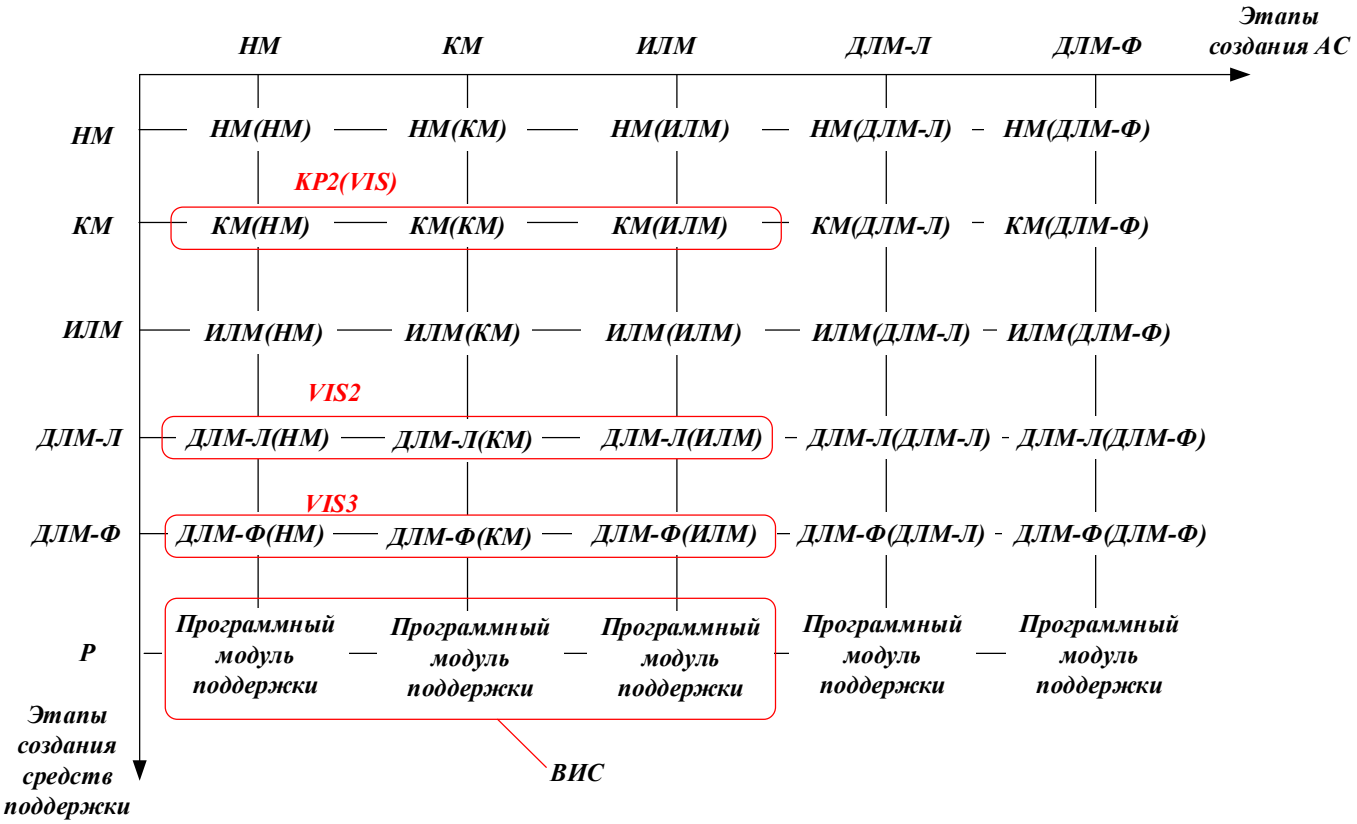


Рис. 2.20 Автоматизация этапов МАИТ в соответствии с самой методологией

Формальное описание объектов увязки основной концептуальной структуры позволит выявить взаимосвязи между объектами, входящими в задачу функционирования визуальной интегрированной среды.

Формально концептуальная модель определяется следующим образом. [46]

Концептуальная модель n-ой предметной задачи состоит из следующих составляющих компонентов:

$$KP2(n) =< A(n), TH2(n), \overline{FU2}(n), R_2^{KP}(n) >,$$

где $A(n)$ – множество предметных категорий, $TH2(n)$ – множество статических взаимосвязях на ПК, $\overline{FU2}(n)$ – множество динамических взаимосвязей на ПК, $R_2^{KP}(n)$ – множество взаимосвязей между составляющими.

При этом множество предметных категорий представляет собой совокупность подмножеств ПК, принадлежащих разным классам (уровням):

$$A(n) = \bigcup_{i=1}^{7} A_i, \quad A_i \cap A_j = \varnothing, \quad A_i = \{a_{il}\},$$

где a_{il} – l-ая ПК на i-ом уровне.

Множество статических взаимосвязей концептуальной модели n-ой предметной задачи имеет вид:

$$TH2(n) = (T2(n), H2(n), \overline{H2}(n), Th2(n)),$$

где $T2(n)$ – множество бинарных связей предметных категорий, $H2(n)$ – множество тернарных связей ПК, $\overline{H2}(n)$ – множество схем предметных категорий, $Th2(n)$ – множество бинарных связей на схемах ПК.

Множество бинарных связей $T2(n)$ содержит следующие подмножества:

$$TA(n) = \{LA_i(n), LA_iA_j(n), LA_j / A_i(n)\},$$

где $LA_i(n)$ –связи типа «упорядочивание», $LA_iA_j(n)$ – связи типа «состав», $LA_j / A_i(n)$ – связи типа «компоновка».

Множество тернарных связей между предметными категориями определяется так:

$$H2(n) = \{HA_i(n)A_j(n)A_k(n)\},$$

Каждая тернарная связь формирует элементарную схему из трех ПК – контекстуальной, структурной, монадической:

$$h^2_{iljmkn} = (a_{il}, a_{jm}, a_{kn})$$

Совокупность элементарных схем с общей контекстуальной категорией образуют схему ПК:

$$\overline{h^2_{il}} = \{h^2_{iljmkn}, ..., h^2_{iljukq}\}, m \neq u$$

Формально множество схем можно представить как:

$$\overline{H2}(n) = \bigcup_i \overline{H2_i}(n), \ \overline{H2_i} \cap \overline{H2_j} = \varnothing, \ \overline{H2_i} = \{\overline{h^2_{il}}\}.$$

Бинарные связи схем задают множество статических ограничений на взаимосвязь схем:

$$Th2(n) \subset \overline{H}2(n) \times \overline{H}2(n),$$

$$Th2(n) = \{L\overline{H}2_i(n), \overline{H}2_i(n)\overline{H}2_k(n), L\overline{H}2_k(n) / \overline{H}2_i(n)\},$$

где $L\overline{H}2_i(n)$ – множество бинарных связей типа «упорядочивание», $\overline{H}2_i(n)\overline{H}2_k(n)$ – множество связей типа «состав», $L\overline{H}2_k(n) / \overline{H}2_i(n)$ – множество связей типа «компоновка»

Множество динамических отношений на элементах концептуальной модели n-ой предметной задачи $\overline{FU2}(n)$ определяется следующим образом:

$$\overline{FU2}(n) = (\overline{F2}(n), \overline{U2}(n)),$$

где $\overline{F2}(n)$ – система предметных зависимостей первого рода, $\overline{U2}(n)$ множество предметных зависимостей второго рода.

Система предметных зависимостей первого рода $\overline{F2}(n)$ – множество зависимостей первого рода и бинарных связей между ними:

$$\overline{F}2(n) = (\overline{W}(n), \overline{FV}2(n)),$$
$$\overline{FV}2(n)) \subset \overline{W}(n) \times \overline{W}(n),$$
$$\overline{W}(n) = \{\overline{w}_{ps}\},$$

где $\overline{W}(n)$ – множество ПЗ первого рода, $\overline{FV}2(n)$ – множество бинарных связей между ПЗ первого рода, $\overline{w}_{ps}$ – s-ая предметная зависимость первого рода p-го уровня сложности.

Система предметных зависимостей второго рода содержит набор ограничений на допустимые сочетания монадических ПК и позволяет формально описать их взаимосвязи. СПЗ-2 состоит из следующих подмножеств:

$$\overline{U}2(n) = (\overline{Q2}(n), \overline{UQ2}(n)),$$

$\overline{Q2}(n)$ – множество предметных зависимостей второго рода, $\overline{UQ2}(n)$ – множество бинарных связей между предметными зависимостями второго рода.

$$\overline{Q2}(n) = \{\overline{q^2_{sp}}\},$$

где $\overline{q^2_{sp}}$ – p-ая ПЗ-2 на s-ом уровне сложности.

Концептуальная структура концептуальной модели процесса функционирования ВИС представлена на рисунках 2.22-2.24.

Для задачи функционирования ВИС множества предметных категорий $A(n)$ состоят из следующих подмножеств: [35]

$A_2(n) \equiv P(n)$ – множество предметных процессов;

$A_3(n) \equiv Z(n)$ – множество предметных задач;

$A_4(n) \equiv K(n)$ – множество предметных компонентов;

$A_5(n) \equiv O(n)$ – множество предметных объектов;

$A_6(n) \equiv R(n)$ – множество предметных признаков.

Так как проектирование ВИС осуществляется в терминах её перспективного функционирования, из множества предметных процессов можно выделить процесс функционирования ВИС:

$P(n) = \{p_a\}$, p_0 – процесс функционирования ВИС.

В рамках процесса функционирования можно выделить задачу функционирования ВИС:

$Z(n) = \{z_a\}$, z_0 – задача функционирования ВИС.

Декомпозиция задачи функционирования ВИС осуществляется в соответствии со структурой абстрактной производственной задачи. (рис. 2.21) [40]

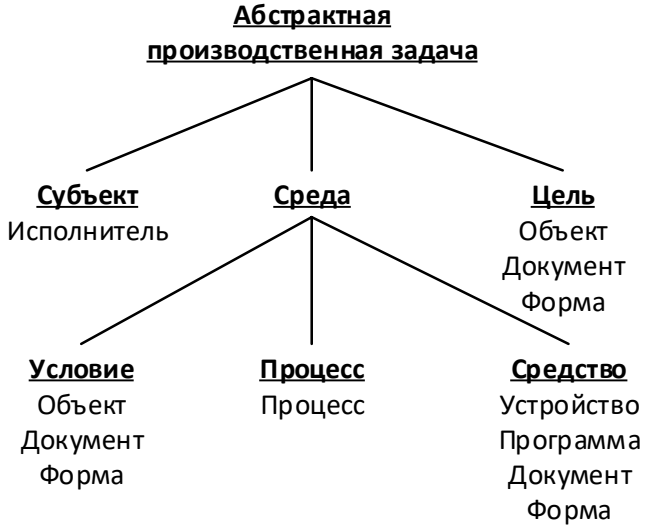


Рис. 2.21 Структура абстрактной производственной задачи

С учетом структуры абстрактной производственной задачи в задаче функционирования ВИС можно выделить следующие компоненты:

$$K(n) = (k_{SR}, k_P, k_{SUB}, K_U, K_C),$$

где k_{SR} – компонент «среда», k_P – «процесс», k_{SUB} – «субъект», K_U – представляет совокупность компонентов «условие», K_C – совокупность компонентов «цель». Множества K_U и K_C имеют аналогичное друг другу наполнение, так как задача функционирования ВИС заключается в изменении состояний одинаковых множеств объектов.

$$\forall t \in ('U','C') \quad K_t = (k_{to}, k_{tof}, k_{tf}, k_{tfd}, k_{td}),$$

где k_{to} – компонент «объект», k_{tof} – «форма объекта», k_{tf} – «форма», k_{tfd} – «форма документа», k_{td} – «документ».

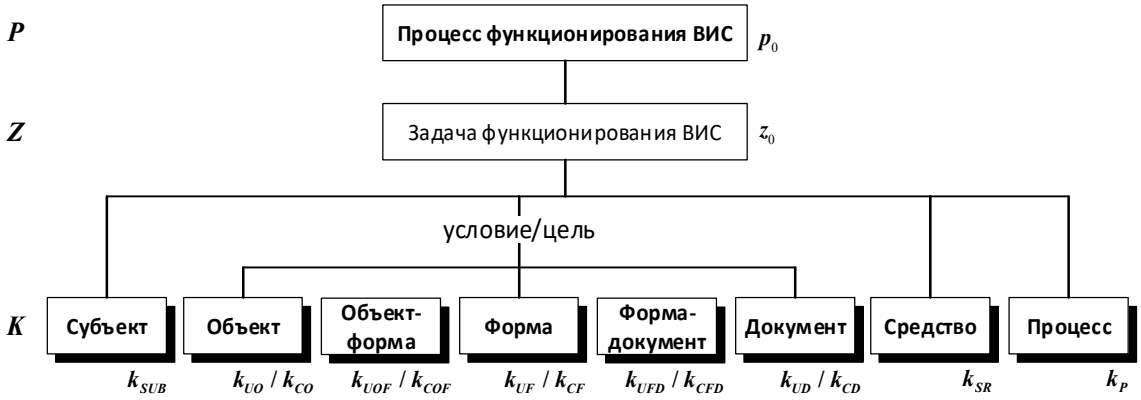


Рис. 2.22 Концептуальная структура процесса функционирования ВИС (начало)

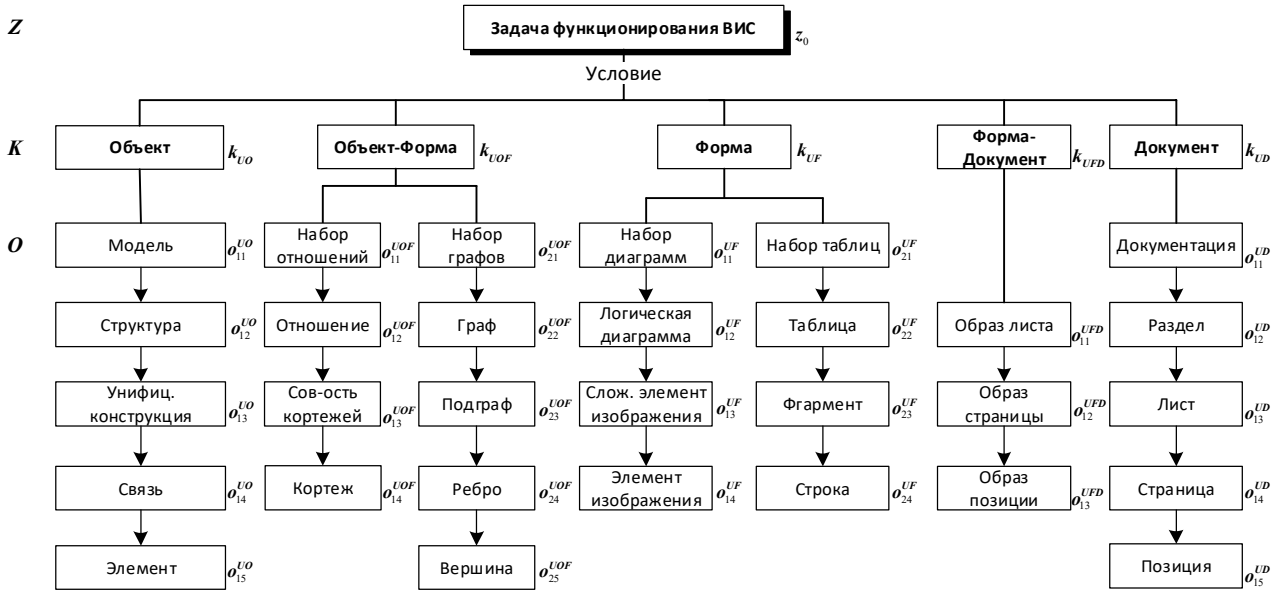


Рис. 2.23 Концептуальная структура процесса функционирования ВИС (продолжение)

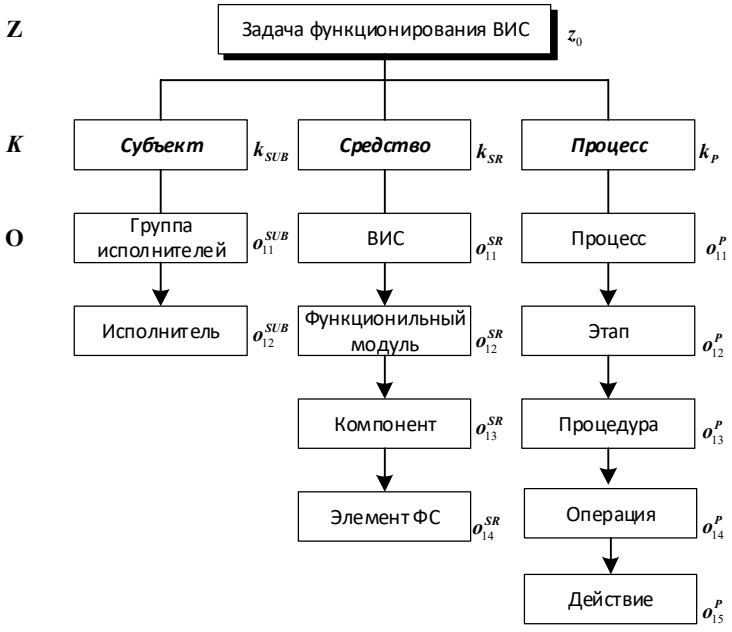


Рис. 2.24 Концептуальная структура процесса функционирования ВИС (завершение)

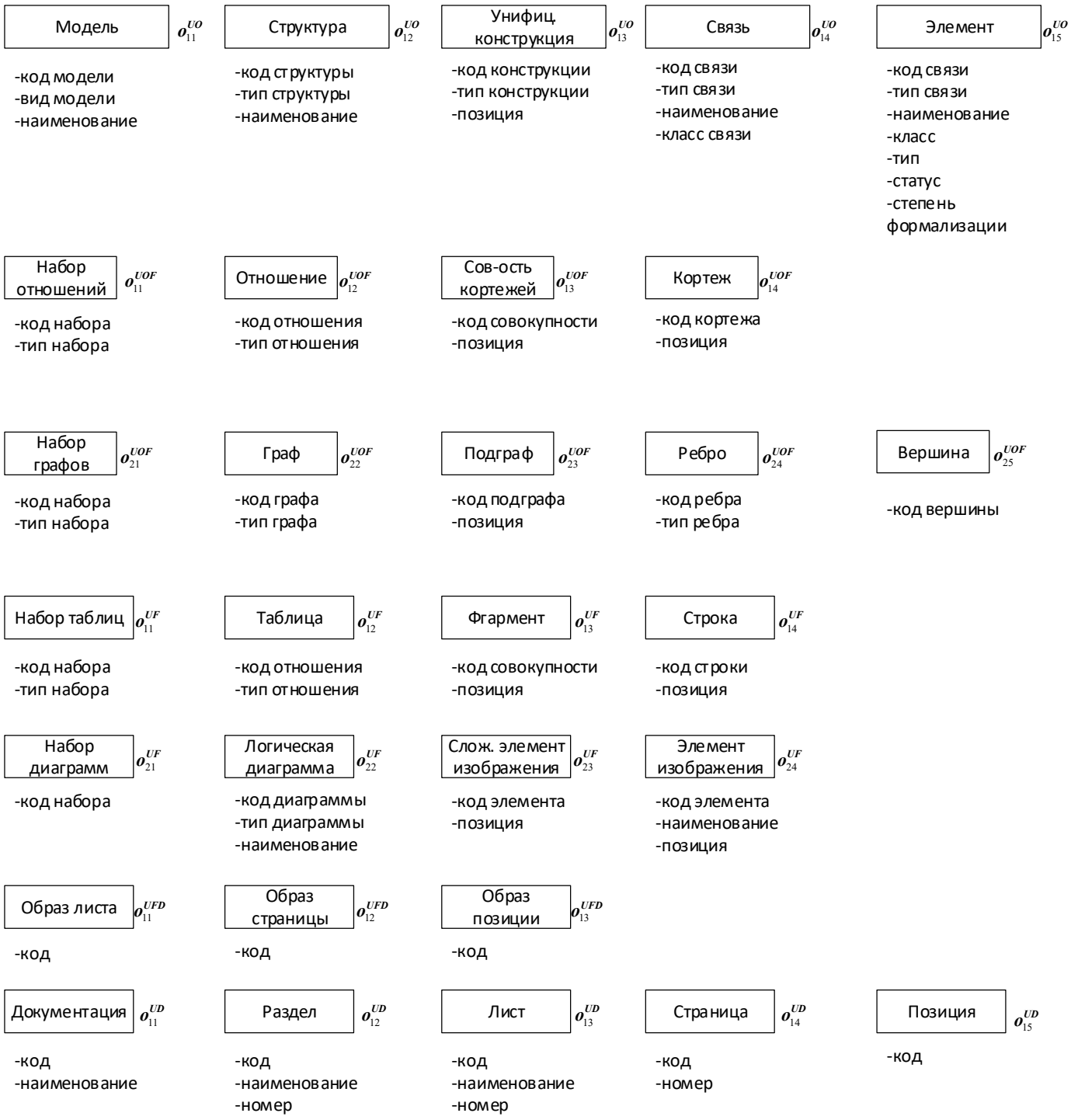


Рис. 2.25 Признаки объектов задачи функционирования ВИС

Множество предметных объектов определяется уточнением вышеперечисленных компонентов:

$$\boldsymbol{O(n) = \{O^f\}}\ ,\ \boldsymbol{O^f = \{o_{lm}^f\}},$$

где $\boldsymbol{o_{lm}^f}$ – объект, уточняющий $\boldsymbol{f}$-ый компонент, индексы $\boldsymbol{l}$ и $\boldsymbol{m}$ определяют номер структуры уточнения компонента и номер объекта в структуре. С учетом множества компонентов совокупность объектов может быть представлена следующим образом:

$$O(n) = (O^U, O^C, O^{SUB}, O^{SR}, O^{PR}),$$

$$\forall t \in ('U','C')\ \ O^t = (O^{tO}, O^{tOF}, O^{tF}, O^{tFD}, O^{tD})\ .$$

где $O^{SUB} = (o_{11}^{SUB}, o_{12}^{SUB})$, o_{11}^{SUB} – объект «группа исполнителей», o_{12}^{SUB} – объект «исполнитель»;

$O^{SR} = (o_{11}^{SR}, o_{12}^{SR}, o_{13}^{SR}, o_{14}^{SR})$, o_{11}^{SR} – объект «вис», o_{12}^{SR} – объект «функциональный модуль», o_{13}^{SR} – объект «компонент», o_{14}^{SR} – объект «элемент файловой системы»;

$O^P = (o_{11}^P, o_{12}^P, o_{13}^P)$, o_{11}^P – объект «процесс», o_{12}^P – объект «этап», o_{13}^P – объект «процедура», o_{14}^P – объект «действие», o_{15}^P – объект «операция»;

$O^{tO} = (o_{11}^{tO}, o_{12}^{tO}, o_{13}^{tO}, o_{14}^{tO}, o_{15}^{tO})$, $\forall t \in ('U','C')$, o_{11}^{tO} – объект «модель», o_{12}^{tO} – объект «структура», o_{13}^{tO} – объект «унифицированная конструкция», o_{14}^{tO} – объект «связь», o_{15}^{tO} – объект «элемент»;

$O^{tOF} = (o_{11}^{tOF}, o_{12}^{tOF}, o_{13}^{tOF}, o_{14}^{tOF}, o_{21}^{tOF}, o_{22}^{tOF}, o_{23}^{tOF}, o_{24}^{tOF}, o_{25}^{tOF})$, $\forall t \in ('U','C')$, o_{11}^{tOF} – объект «набор отношений», o_{12}^{tOF} – объект «отношение», o_{13}^{tOF} – объект «совокупность кортежей», o_{14}^{tOF} – объект «кортеж», o_{21}^{tOF} – объект «набор графов», o_{22}^{tOF} – объект «граф», o_{23}^{tOF} – объект «подграф», o_{24}^{tOF} – объект «ребро», o_{25}^{tOF} – объект «вершина»;

$O^{tF} = (o_{11}^{tF}, o_{12}^{tF}, o_{13}^{tF}, o_{14}^{tF}, o_{21}^{tF}, o_{22}^{tF}, o_{23}^{tF}, o_{24}^{tF})$, $\forall t \in ('U','C')$, o_{11}^{tF} – объект «набор диаграмм», o_{12}^{tF} – объект «логическая диаграмма», o_{13}^{tF} – объект «сложный элемент изображения», o_{14}^{tF} – объект «элемент изображения», o_{21}^{tF} – объект «набор таблиц», o_{22}^{tF} – объект «таблица», o_{23}^{tF} – объект «фрагмент», o_{24}^{tF} – объект «строка»;

$O^{tFD} = (o_{11}^{tFD}, o_{12}^{tFD}, o_{13}^{tFD})$, $\forall t \in ('U','C')$, o_{11}^{tFD} – объект «образ листа», o_{12}^{tFD} – объект «образ страницы», o_{13}^{tFD} – объект «образ позиции»;

$O^{tD} = (o_{11}^{tD}, o_{12}^{tD}, o_{13}^{tD}, o_{14}^{tD}, o_{15}^{tD},)$, $\forall t \in ('U','C')$ o_{11}^{tD} – объект «документация», o_{12}^{tD} – объект «раздел», o_{13}^{tD} – объект «лист», o_{14}^{tD} – объект «страница», o_{15}^{tD} – объект «позиция».

$R(n) = \{r_k^{flm}\}$ – множество признаков объекта o_{lm}^f.

$C(n) = \{c_e^{flmk}\}$ – множество значений признака r_k^{flm}

Для задачи функционирования визуальной интегрированной среды множество бинарных связей типа «состав» определяется следующим образом:

$$LA_iA_j(n) = \{LA_2(n)A_3(n), LA_3(n)A_4(n), LA_4(n)A_5(n), LA_5(n)A_6(n), LA_6(n)A_7(n)\},$$

где с учетом переобозначения:

$$LA_2(n)A_3(n) \equiv LP(n)Z(n) = \{(p_a, z_b) \mid p_a \in P(n), z_a \in Z(n)\}$$

определяет связи между ПК классов «процесс» и «задача»;

$$LA_3(n)A_4(n) \equiv LZ(n)K(n) = \{(z_a, k_b) \mid z_a \in Z(n), k_b \in K(n)\}$$

определяет связи между ПК классов «задача» и «компонент»;

$$LA_4(n)A_5(n) \equiv LK(n)O(n) = \{(k_f, o_{lm}^f) \mid k_f \in K(n), o_{lm}^f \in O(n)\}$$

определяет связи между ПК классов «компонент» и «объект»;

$$LA_5(n)A_6(n) \equiv LO(n)R(n) = \{(o_{lm}^f, r_k^{flm}) \mid o_{lm}^f \in O(n), r_k^{flm} \in R(n)\}$$

определяет связи между ПК классов «объект» и «признак»;

$$LA_6(n)A_7(n) = LR(n)C(n) = \{(r_k^{flm}, o_e^{flmk}) \mid r_k^{flm} \in R(n), o_e^{flmk} \in C(n)\}$$

определяет связи между ПК классов «признак» и «значение»

Бинарные связи типа «компоновка» в основной концептуальной структуре задачи функционирования ВИС определены следующим образом:

$$LA_j \,/\, A_i(n) = \{LA_3(n) \,/\, A_2(n), LA_4(n) \,/\, A_3(n), LA_4(n)A_5(n), LA_5(n)A_6(n)\},$$

где с учетом переобозначения:

$$LA_3(n) \,/\, A_2(n) \equiv LZ(n) \,/\, P(n) = \{(z_a, z_d) \mid$$
$$\mid \exists p_b : (p_b, z_a) \in LP(n)Z(n) \,\&\, (p_b, z_d) \in LP(n)Z(n)\},$$

определяет компоновку ПК классов «задача» в рамках вышестоящей ПК класса «процесс»;

$$LA_4(n) \,/\, A_3(n) \equiv LK(n) \,/\, Z(n) = \{(k_f, k_g) \mid$$
$$\mid \exists z_a : (z_a, k_f) \in LZ(n)K(n) \,\&\, (z_a, k_g) \in LZ(n)K(n)\}$$

определяет компоновку ПК классов «компонент» в рамках ПК класса «задача»;

$$LA_5(n) \,/\, A_4(n) \equiv LO(n) \,/\, K(n) = \{(o_{ij}^f, o_{ik}^f) \mid$$
$$\mid \exists k_f : (k_f, o_{ij}^f) \in LK(n)O(n) \,\&\, (k_f, o_{ik}^f) \in LK(n)O(n)\}$$

определяет компоновку ПК классов «объект» в рамках вышестоящей ПК класса «компонент»;

$$LA_6(n) \,/\, A_5(n) \equiv LR(n) \,/\, O(n) = \{(r_k^{flm}, r_t^{flm}) \mid$$
$$\mid \exists o_{lm}^f : (o_{lm}^f, r_k^{flm}) \in LO(n)R(n) \,\&\, (o_{lm}^f, r_t^{flm}) \in LO(n)R(n)\}$$

определяет компоновку ПК классов «признак» в рамках вышестоящей ПК класса «объект»;

$$LA_7(n) \,/\, A_6(n) \equiv LC(n) \,/\, R(n) = \{(c_e^{flmk}, r_v^{flmk}) \mid$$
$$\mid \exists r_k^{flm} : (r_k^{flm}, c_e^{flmk}) \in LR(n)C(n) \,\&\, (r_k^{flm}, r_v^{flmk}) \in LR(n)C(n)\}$$

определяет компоновку ПК классов «значение» в рамках вышестоящей ПК класса «признак»;

Множество тернарных связей ПК задачи функционирования ВИС выглядит следующим образом:

$$H2(n) = \{HZ(n)K(n)O(n),\ HO(n)R(n)C(n)\}.$$

При этом каждая тернарная связь определяет элементарную схему ПК:

$$HZ(n)K(n)O(n) = \{h^2_{z_a k_f o^f_{ij}}\} = \{(z_a, k_f, o^f_{il}) \mid$$
$$|(z_a, k_f) \in LZ(n)K(n),\ (k_f, o^f_{il}) \in LK(n)LO(n)\},$$

где $h^2_{z_a k_f o^f_{ij}}$ – элементарная схема, включающая ПК z_a – контекстуальную, k_f – структурную и o^f_{il} – монадическую;

$$HO(n)R(n)C(n) = \{h^2_{o^f_{lm} r^{flm}_k c^{flmk}_e}\} = \{(o^f_{lm}, r^{flm}_k, c^{flmk}_e) \mid$$
$$|(o^f_{lm}, r^{flm}_k) \in LO(n)R(n),\ (r^{flm}_k, c^{flmk}_e) \in LR(n)LC(n)\},$$

где $h^2_{o^f_{lm} r^{flm}_k c^{flmk}_e}$ элементарная схема, включающая следующие ПК: o^f_{lm}. – контекстуальную, r^{flm}_k – структурную и c^{flmk}_e – монадическую.

С учетом выделенных элементарных схем, в концептуальной структуре задачи функционирования ВИС можно сформировать следующие множества схем ПК:

$$\overline{H}2(n) = \{\overline{H}2_{z_a}(n), \overline{H}2_{o^f_{lm}}(n)\},\ \overline{H}2_{z_a}(n) = \{\overline{h^2_{z_a}}\},\ \overline{H}2_{o^f_{lm}}(n) = \{\overline{h^2_{o^f_{lm}}}\}$$

где $\overline{h^2_{z_a}}$ – схема задачи z_a, $\overline{h^2_{o^f_{lm}}}$ – схема объекта o^f_{lm}.

С учетом множества выделенных схем были сформированы бинарные связи:

бинарные связи типа «состав»:

$$\overline{H}2_i(n)\overline{H}2_k(n) \equiv \overline{H}2_{z_a}(n)\overline{H}2_{o^f_{lm}}(n) = \{(h^2_{z_a}, h^2_{o^f_{lm}}) \mid h^2_{z_a} \in \overline{H}2_{z_a}, h^2_{o^f_{lm}} \in \overline{H}2_{o^f_{lm}}\};$$

бинарные связи типа «компоновка»:

$$L\overline{H}2_k(n) / \overline{H}2_i(n) \equiv L\overline{H}2_{o^f_{lm}} / \overline{H}2_{z_a}(n) = \{(\overline{h^2_{o^f_{lm}}}, \overline{h^2_{o^j_{pq}}}) \mid$$
$$|\exists \overline{h^2_{z_a}} : (\overline{h^2_{z_a}}, \overline{h^2_{o^f_{lm}}}) \in L\overline{H}2_{z_a}\,\overline{H}2_{o^f_{lm}}\ \&\ (\overline{h^2_{z_a}}, \overline{h^2_{o^j_{pq}}}) \in L\overline{H}2_{z_a}\,\overline{H}2_{o^f_{lm}}\},\ f \neq j.$$

Построение схем ПК для задачи функционирования ВИС связано с комбинаторным перебором всех возможных вариаций схем, т.е. перебором всех возможных сочетаний монадических ПК для каждой структурной ПК. Для предметной задачи функционирования ВИС количество возможных сочетаний равно 349 920 000.

Для задачи функционирования ВИС в общем виде ПЗ-2 можно представить как сочетания объектов, принадлежащих различным компонентам:

$$\overline{q^2_{sp}}(i, t, u_{sp}) = (o^f_{kl}, ..., o^g_{mn})_{u_{sp}}\ \&\ F^{sp-2},$$

где F^{sp-2} – условия, описывающие взаимосвязи объектов и компонентов, u_{sp} – множество объектов, входящих в данную ПЗ-2.

Ограничения на взаимосвязи объектов в задачи функционирования ВИС представлены на рисунках 2.26, 2.27.

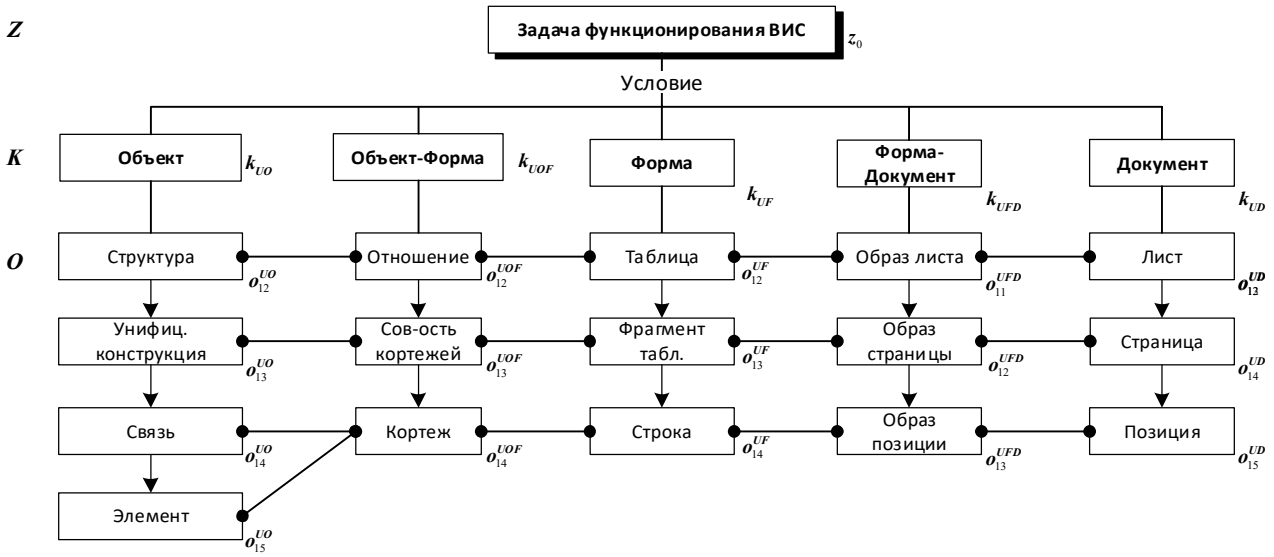


Рис. 2.26 Ограничения на взаимодействия объектов при табличном представлении моделей

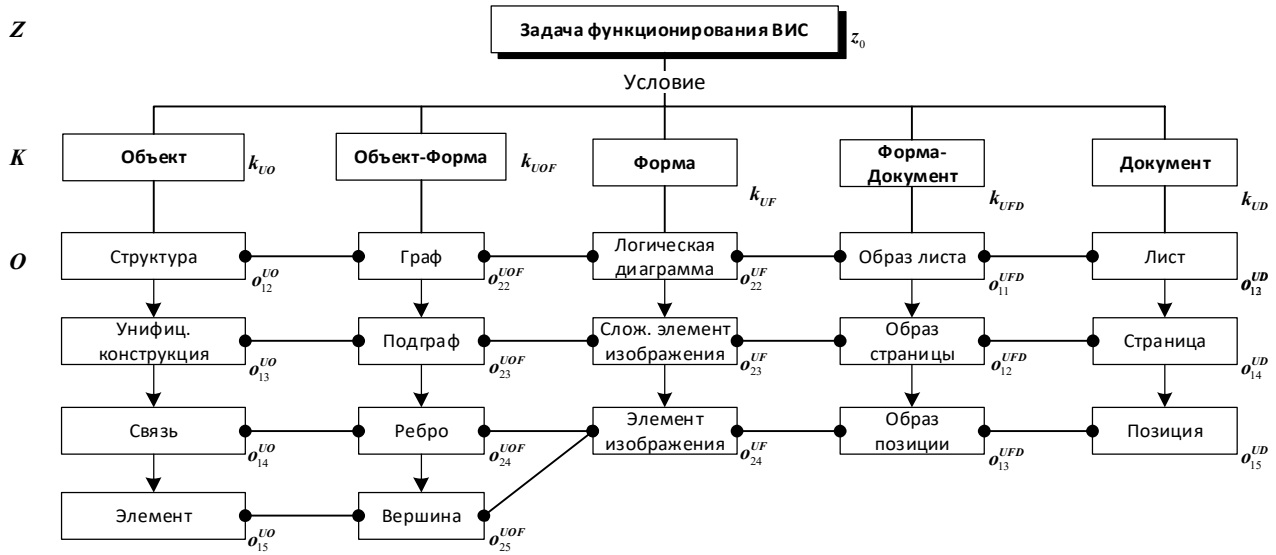


Рис. 2.27 Ограничения на взаимодействия объектов при графическом представлении моделей

Исходя из того, что динамические и/или функциональные составляющие модели формируются на основе базовых и типовых конструкций путем их суперпозиции, то это означает, что соответствующие им графы формируются из унифицированных подграфов, свойства которых будут включать как унифицированные характеристики, так специфические, зависящие от вида модели и индекса структуры.

Аналогично, статическая составляющая модели строится на основе чередования унифицированных подструктур (ранжирования и иерархии), свойства которых также отражают универсальные и специфические характеристики.

Для содержательных составляющих модели, обеспечивающих увязку статических, динамических и/или функциональных структур, граф представляется в матричной форме, где горизонтальный заголовок отражает статическую составляющую, а вертикальный заголовок – динамическую и/или функциональную, а пересечение строки со столбцами отражает элементарную

увязку динамического или функционального элемента со статической структурой.

Такая организация составляющих модельных представлений позволяет выделить в них как специфические, так и инвариантные свойства.

2.2.3 Уточнение формального описания визуальной интегрированной среды

Для выявления универсальных составляющих в модулях ВИС требуется произвести уточнения формального описания даталогической модели среды, с учетом выявленных особенностей структуры моделей МАИТ и процедур их обработки.

Часть версии функционального модуля $\overline{m_f^{sijkl}} \subset m_f^{sijkl}$ может обрабатывать общие для различных моделей свойства элементов, связей элементов или их представлений. Таким образом, частично информационная, алгоритмическая или интерфейсная структуры различных модулей ВИС могут совпадать. Совпадающие части определяются общим для всех моделей перечнем графовых структур, инвариантным набором процедур формирования и обработки моделей, общим способом представления всех моделей в виде диаграмм и спецификаций.

Таким образом, универсальная составляющая u_f^{ij} любого функционального модуля m_f^{sijkl} будет определена ***j***-ой процедурой и ***k***-ой операций:

$$\overline{m_f^{sijkl}} \equiv u_f^{ij} \subset m_w^{ijkl}, \forall s,k,l\,.$$

Такие инвариантные части модулей u_f^{ij} реализуются универсальными информационными, функциональными и интерфейсными компонентами:

$$u_f^{jk} \to < UINF_f^{jk}, UFUN_f^{jk}, UINT_f^{jk}, UR2_f^{jk} >.$$

Как было показано ранее, каждый модуль ВИС есть совокупность версий компонентов трех типов и их связей:

$$m_f^{sijkl} \to < INF_f^{sijkl}, FUN_f^{sijkl}, INT_f^{sijkl}, R2_f^{sijkl} >.$$

Для информационного компонента любого модуля можно выделить две составляющие: инвариантную (универсальную) и специфическую (специализированную):

$$INF_f^{sijkl} =< MINF_f^{sijkl}, UINF_f^{jk}, R2_{sijkf}^{MU-INF} >,$$

где $MINF_f^{sijkl}$ – версия специфической составляющей информационного компонента, $UINF_f^{jk}$ – версия универсальной составляющей информационного компонента, $R2_{sijkf}^{MU-INF}$ – взаимосвязи между универсальной и специфической составляющей информационного компонента версии

модуля. Индекс универсального компонента определяет к какой процедуре и операции он относится.

Аналогично, функциональные и интерфейсные компоненты могут быть декомпозированы на универсальные и специализированные части:

$$FUNK_f^{sijkl} =< MFUNK_f^{sijkl}, UFUN_f^{ijk}, R2_{sijkf}^{MU-FUNK} >,$$

$$INT_f^{sijkl} =< MINT_f^{sijkl}, UINT_f^{ijk}, R2_{sijkf}^{MU-INT} >.$$

Специфическая составляющая информационного компонента $MINF_f^{sijkl}$ может быть представлена множеством специализированных элементов файловой системы:

$$MINF_f^{sijkl} \to < MTI_f^{sijkl}, MDI_f^{sijkl}, MFI_f^{sijkl}, MRI2_f^{sijkl} >,$$

где MTI_f^{sijkl} – множество специализированных информационных томов средства поддержки m_f^{sijkl}, MDI_f^{sijkl} – множество специализированных информационных директорий, MFI_f^{sijkl} – множество специализированных информационных файлов, $MRI2_f^{sijkl}$ – взаимосвязи между специализированными информационными элементами ФС.

Специфическая составляющая функционального компонента $MFUN_f^{sijkl}$ на уровне файловой системы представлена как:

$$MFUN_f^{sijkl} \to < MTD_f^{sijkl}, MDD_f^{sijkl}, MFD_f^{sijkl}, MTP_f^{sijkl}, MDP_f^{sijkl}, MFP_f^{sijkl}, MRPD2_f^{sijkl} >,$$

где MTD_f^{sijkl} – множество специализированных томов доступов средства поддержки m_f^{sijkl}, MDD_f^{sijkl} – множество специализированных директорий доступов, MFD_f^{sijkl} – множество специализированных файлов доступов, где MTP_f^{sijkl} – множество специализированных томов процедур, MDP_f^{sijkl} – множество специализированных директорий процедур, MFP_f^{sijkl} – множество специализированных файлов процедур, $MRPD2_f^{sijkl}$ – взаимосвязи между специализированными функциональных элементами ФС.

Специфическая составляющая интерфейсного компонента $MINT_f^{sijkl}$ на уровне файловой системы представлена как:

$$MINT_f^{sijkl} \to < MIT_f^{sijkl}, MID_f^{sijkl}, MIF_f^{sijkl}, MIR2_f^{sijkl} >$$

где MIT_f^{sijkl} – множество специализированных интерфейсных томов средства поддержки m_f^{sijkl}, MID_f^{sijkl} – множество специализированных интерфейсных директорий, MIF_f^{sijkl} – множество специализированных интерфейсных файлов, $MIR2_f^{sijkl}$ – взаимосвязи между специализированными интерфейсными элементами ФС.

Универсальная составляющая информационного компонента INF_f^{sijk} может быть представлена множеством специализированных элементов файловой системы:

$$UINF_f^{jk} \rightarrow < UTI_f^{jk}, UDI_f^{jk}, UFI_f^{jk}, URI2_f^{jk} >,$$

где UTI_f^{jk} – множество универсальных информационных томов средства поддержки m_f^{sijkl}, UDI_f^{jk} – множество универсальных информационных директорий, UFI_f^{jk} – множество универсальных информационных файлов, $URI2_f^{jk}$ – взаимосвязи между универсальными информационными элементами ФС.

Универсальная составляющая функционального компонента FUN_f^{sijkl} на уровне файловой системы представлена как:

$$UFUN_f^{jk} \rightarrow < UTD_f^{jk}, UDD_f^{jk}, UFD_f^{jk}, UTP_f^{jk}, UDP_f^{jk}, UFP_f^{jk}, URPD2_f^{jk} >,$$

где UTD_f^{jk} – множество универсальных томов доступов средства поддержки m_f^{sijkl}, UDD_f^{jk} – множество универсальных директорий доступов, UFD_f^{jk} – множество универсальных файлов доступов, где UTP_f^{jk} – множество универсальных томов процедур, UDP_f^{jk} – множество универсальных директорий процедур, UFP_f^{jk} – множество универсальных файлов процедур, $URPD2_f^{jk}$ – взаимосвязи между универсальными функциональных элементами ФС.

Универсальная составляющая интерфейсного компонента INT_f^{sijkl} на уровне файловой системы представлена как:

$$UINT_f^{jk} \rightarrow < UIT_f^{jk}, UID_f^{jk}, UIF_f^{jk}, UIR2_f^{jk} >$$

где UIT_f^{jk} – множество универсальных интерфейсных томов средства поддержки m_f^{sijkl}, UID_f^{jk} – множество универсальных интерфейсных директорий, UIF_f^{jk} – множество универсальных интерфейсных файлов, $UIR2_f^{jk}$ – взаимосвязи между универсальными интерфейсными элементами ФС.

С учетом уточненного описания каждый функциональный модуль будет определяться множествами файлов двух различных видов (рис. 2.28): специализированными файлами, входящими в состав только одного модуля, и универсальными – принадлежащими нескольким модулям, обрабатывающих один вид модели, структуры или процедуры.

Метод моделирования ВИС позволяет сформировать структуру среды в виде множеств версий средств поддержки процедур этапов МАИТ. Графическая интерпретация структуры приведена на рисунке 2.29.

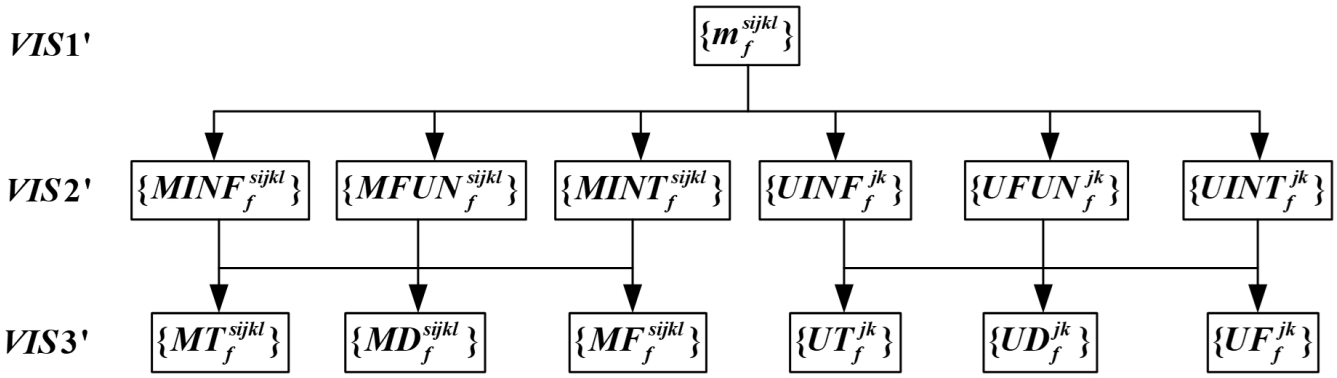


Рис. 2.28 Структура уточненного описания ВИС

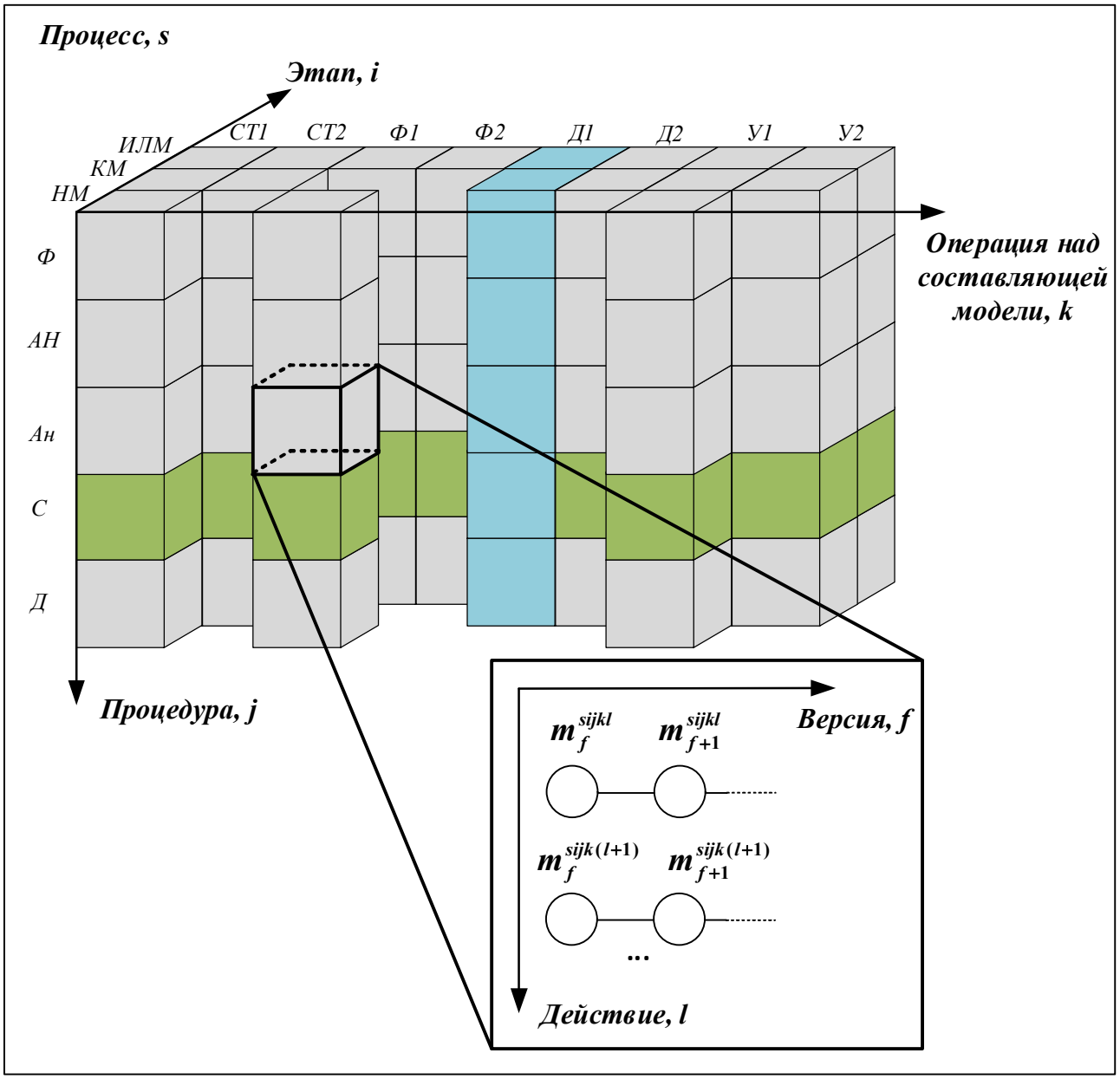


Рис. 2.29 Графическая интерпретация структуры ВИС

2.3 Выводы по главе

Сформулированные требования к визуальной интегрированной среде позволили перейти к разработке метода её моделирования.

При анализе методов и методик, составляющих методологию автоматизации интеллектуального труда, была проведена унификация модельных представлений и процедур их обработки на этапах предпроектного обследования и проектирования АС. На основании особенностей состава и структуры модельных представлений было сформировано объединенное модельное представление, позволяющее описать начальную, концептуальную и инфологическую модели в общем виде. На основании единообразного описания моделей как под отдельные предметные задачи, так и под комплекс задач был унифицирован перечень процедур, применяемых при формировании и обработке моделей.

Унификация моделей и процедур их обработки, а также применение системного подхода позволило выявить взаимосвязи между характеристиками процедур моделирования в рамках МАИТ, функциями функционирования визуальной интегрированной среды, функциями управления её функционированием, и функциями её развития.

На основании выделенных взаимосвязей разработан метод моделирования визуальной интегрированной среды, описывающий среду на различных уровнях детализации: как множество функциональных модулей, на логическом уровне – как множество компонентов, и на физическом – как множество элементов файловой системы. Особенность разработанного метода заключается в разделении элементов ВИС на логическом и физическом уровнях на универсальные и специализированные с учетом общих и различных свойств элементов и связей элементов в моделях, а также процедур их обработки.

Разработанный метод моделирования ВИС позволил сформировать её состав и структуру и перейти к разработке методики управления развитием среды

ГЛАВА 3
РАЗРАБОТКА МЕТОДИКИ УПРАВЛЕНИЯ РАЗВИТИЕМ ВИЗУАЛЬНОЙ ИНТЕГРИРОВАННОЙ СРЕДЫ

3.1 Общие положения и обоснования

Промышленный процесс создания автоматизированных систем в соответствии с МАИТ предполагает последовательное выполнение набора функциональных процедур на всех этапах создания АС. Результатом выполнения функциональных процедур МАИТ является формирование документации, содержащей специальные диаграммы и спецификации. Первоначально применялась «ручная» технология выполнения процедур, т.е. без использования средств автоматизации. Впоследствии, для сокращения времени и количества ошибок, возникающих при ручном выполнении процедур, часть из них была автоматизирована отдельными средствами поддержки. Возникла проблема информационного взаимодействия между различными средствами поддержки, которое осуществлялось, опять же, в ручном режиме. Для устранения ручного переноса данных между средствами поддержки, была разработана интегрированная среда (ИС), объединившая различные средства поддержки процедур в виде функциональных модулей, осуществляя управление функционированием этих модулей и обеспечивая их информационную связность.

Недостатком ИС является ориентация на полуавтоматизированную технологию формирования моделей, ориентированную на частично автоматизированный ввод данных спецификаций в среду и последующее автоматическое формирование контрольных диаграмм по спецификациям. Сравнив контрольные диаграммы с «ручным» вариантом, можно сделать вывод о корректности введённых данных.

Разработка и применение графических редакторов для формирования всех видов диаграмм на начальные этапы создания позволило перейти к формированию новой визуальной интегрированной среды.

Особенностью ***визуальной*** интегрированной среды является формирование её функциональных модулей из универсальных компонентов, и, входящих в их состав, универсальных файлов, реализующих одинаковые информационные, функциональные и интерфейсные структуры, присущие различным модулям. Разделение компонентов на универсальные и специализированные вынуждает пересмотреть как процесс конфигурирования ИС, так и процесс её развития в целом. Исходя из этого, требуется разработка методики управления развитием визуальной интегрированной среды.

Методика управления развитием визуальной интегрированной среды поддержки проектирования автоматизированных систем предназначена для:

а) специалистов в области разработки программного обеспечения, развивающих функциональность системы;

б) системных администраторов и специалистов по конфигурированию программного обеспечения, обеспечивающих настройку версии интегрированной среды для решения требуемых задач.

Разработка методики управления развитием включает следующие аспекты:

-разработка классификации и структуры элементов визуальной интегрированной среды различного уровня детализации (модули, компоненты, файлы) на основании: структур моделей предметных задач объектного уровня в соответствии с МАИТ; методик выполнения функциональных процедур на этапах начального, концептуального и инфологического моделирования; особенностей различных видов представления структур формируемых моделей; формального описания визуальной интегрированной среды.

-разработка формы представления версий интегрированной среды на различных уровнях детализации её даталогической модели;

-разработка форм фиксации состава и структуры визуальной интегрированной среды и правил их заполнения;

-разработка методики формирования новых компонентов среды, их организации и взаимодействия;

-разработка форм и правил их заполнения для фиксации версий ВИС.

3.2 Место методики управления развитием визуальной интегрированной среды

Методика управления развитием визуальной интегрированной среды предполагает манипулирование её файловой структурой для поэтапного наращивания функциональных возможностей.

Поэтому, ввиду особенностей структуры ВИС для её поэтапной модернизации требуется разработка новой методики управления развитием, ориентированной на конфигурирование среды путем манипулирования файлами двух видов: универсальными и специализированными.

Процесс развития ВИС носит циклический характер – каждая следующая версия ВИС разрабатывается средствами текущей. (рис. 3.1)

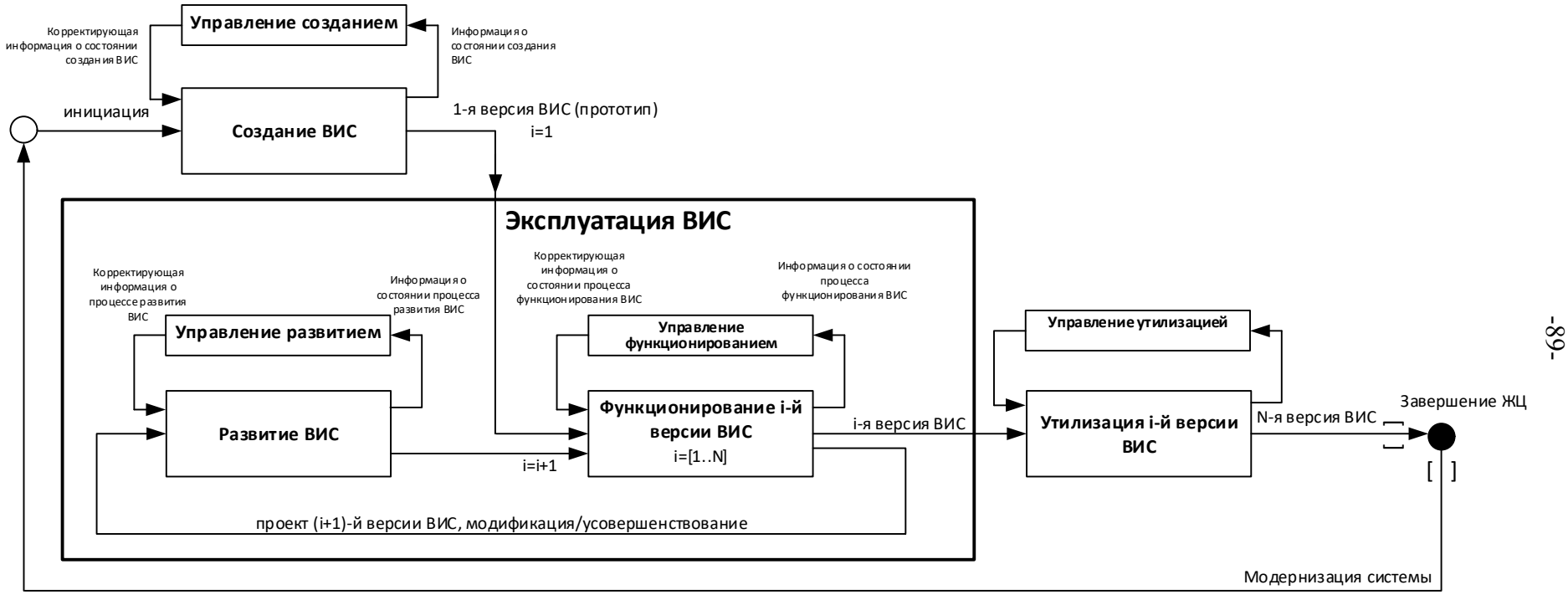


Рис. 3.1 Место процесса управления развитием в жизненном цикле ВИС

3.3 Разработка классификатора модулей визуальной интегрированной среды

Сформированная модель визуальной интегрированной среды позволила разработать классификатор функциональных модулей (табл. 3.1). Для каждого класса модуля был определен примерный перечень выходных данных, в соответствии с методиками формирования и обработки моделей согласно МАИТ. (рис. 3.2) Некоторые процедуры еще не имеют методической проработки, однако место будущих средств их поддержки в составе визуальной среды уже определено.

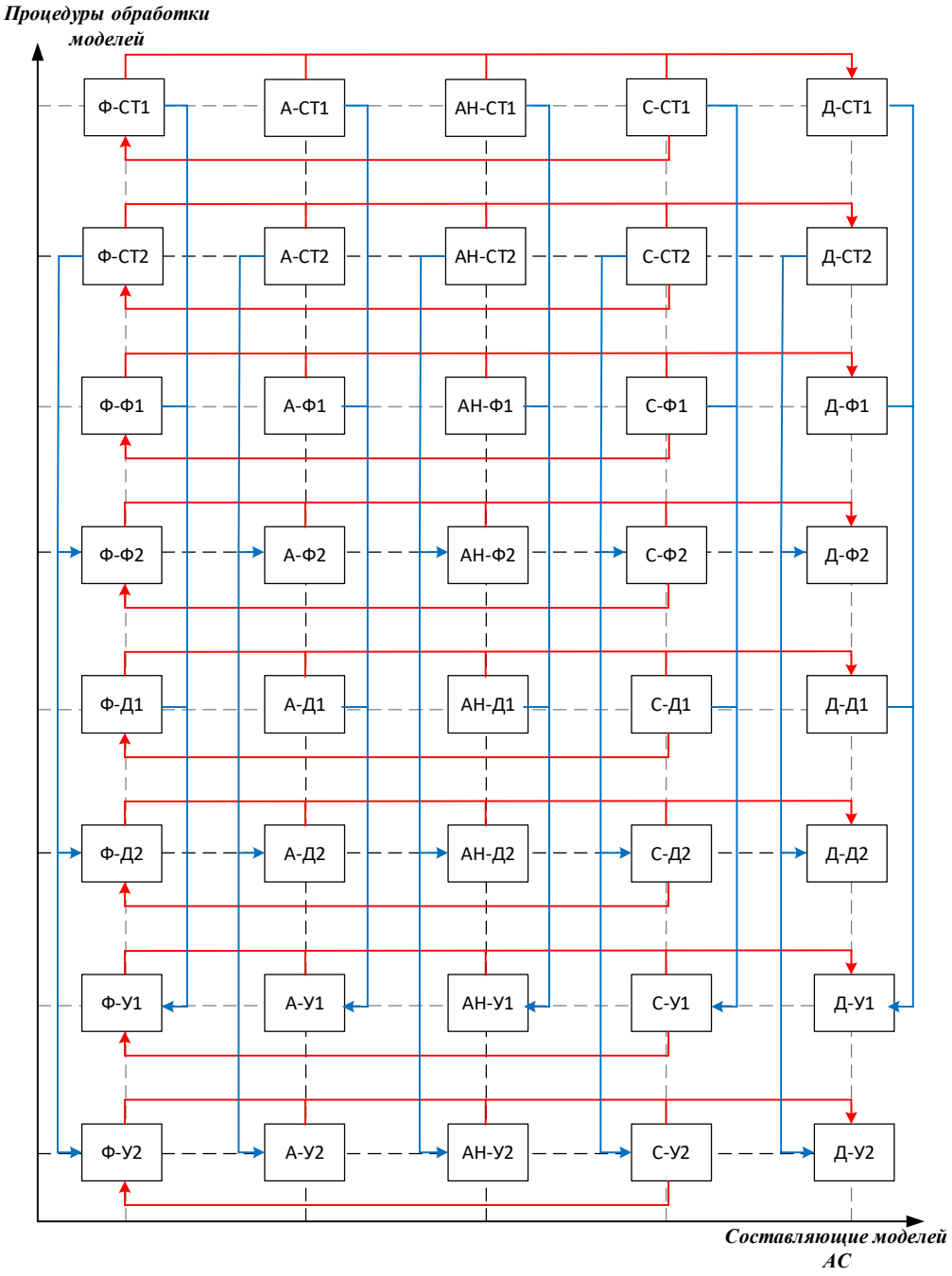


Рис. 3.2 Взаимосвязи между классами модулей ВИС

Таблица 3.1 Классификатор модулей ВИС

Наименование классификатора	**Описание классификатора**	**Выходные данные**
Ф_СТ1	Модули формирования основной структуры статической составляющей	Информация об элементах модели, о бинарных связях элементов модели
Ф_СТ2	Модули формирования производной структуры статической составляющей	Информация о схемах элементов модели, бинарных связях схем
Ф_Д1	Модули формирования динамической составляющей первого рода	Информация о динамических элементах модели первого рода, бинарных связях этих элементов
Ф_Д2	Модули формирования динамической составляющей второго рода	Информация о динамических элементах модели второго рода, бинарных связях этих элементов
Ф_Ф1	Модули формирования функциональной составляющей первого рода	Информация о функциональных элементах модели первого рода, бинарных связях этих элементов
Ф_Ф2	Модули формирования функциональной составляющей второго рода	Информация о функциональных элементах модели второго рода, бинарных связях этих элементов
Ф_У1	Модули формирования модели в целом на основных структурах	Информация о взаимосвязях информационных элементов, функциональных и динамических элементов первого рода
Ф_У2	Модули формирования модели в целом на производных структурах	Информация о взаимосвязях информационных элементов, функциональных и динамических элементов второго рода
А_СТ1	Модули анализа основной структуры статической составляющей	Информация о корректности структуры
А_СТ2	Модули анализа производной структуры статической составляющей	Информация о корректности структуры
А_Д1	Модули анализа динамической составляющей первого рода	Информация о корректности структуры

Наименование классификатора	**Описание классификатора**	**Выходные данные**
А_Д2	Модули анализа динамической составляющей второго рода	Информация о корректности структуры
А_Ф1	Модули анализа функциональной составляющей первого рода	Информация о корректности структуры
А_Ф2	Модули анализа функциональной составляющей второго рода	Информация о корректности структуры
А_У1	Модули анализа модели в целом на основных структурах	Информация о корректности структуры
А_У2	Модули анализа модели в целом на производных структурах	Информация о корректности структуры
АН_СТ1	Модули аналитической обработки основной структуры статической составляющей	Результаты применения операций аналитической обработки на статической структуре, в зависимости от комплекса операций
АН_СТ2	Модули аналитической обработки производной структуры статической составляющей	Результаты применения операций аналитической обработки на производной структуре, в зависимости от комплекса операций
АН_Д1	Модули аналитической обработки динамической составляющей первого рода	Результаты применения операций аналитической обработки на структуре первого рода, в зависимости от комплекса операций
АН_Д2	Модули аналитической обработки динамической составляющей второго рода	Результаты применения операций аналитической обработки на структуре второго рода, в зависимости от комплекса операций
АН_Ф1	Модули аналитической обработки функциональной составляющей первого рода	Результаты применения операций аналитической обработки на структуре первого рода, в зависимости от комплекса операций

Наименование классификатора	Описание классификатора	Выходные данные
АН_Ф2	Модули аналитической обработки функциональной составляющей второго рода	Результаты применения операций аналитической обработки на структуре второго рода, в зависимости от комплекса операций
АН_У1	Модули аналитической обработки модели в целом на основных структурах	Результаты применения операций аналитической обработки на структуре, в зависимости от комплекса операций
АН_У2	Модули аналитической обработки модели в целом на производных структурах	Результаты применения операций аналитической обработки на структуре, в зависимости от комплекса операций
Д_СТ1	Модули документирования основной структуры статической составляющей	Табличное/графическое представление структуры
Д_СТ2	Модули документирования производной структуры статической составляющей	Табличное/графическое представление структуры
Д_Д1	Модули документирования динамической составляющей первого рода	Табличное/графическое представление структуры
Д_Д2	Модули документирования динамической составляющей второго рода	Табличное/графическое представление структуры
Д_Ф1	Модули документирования функциональной составляющей первого рода	Табличное/графическое представление структуры
Д_Ф2	Модули документирования функциональной составляющей второго рода	Табличное/графическое представление структуры
Д_У1	Модули документирования модели в целом на основных структурах	Табличное/графическое представление структуры
Д_У2	Модули документирования модели в целом на производных структурах	Табличное/графическое представление структуры

Наименование классификатора	Описание классификатора	Выходные данные
С_СТ1	Модули синтеза основных структур статической составляющей	Информация об элементах модели, о бинарных связях элементов модели
С_СТ2	Модули синтеза производных структур статической составляющей	Информация о схемах элементов модели, бинарных связях схем
С_Д1	Модули синтеза динамических составляющих первого рода	Информация о динамических элементах модели первого рода, бинарных связях этих элементов
С_Д2	Модули синтеза динамических составляющих второго рода	Информация о динамических элементах модели второго рода, бинарных связях этих элементов
С_Ф1	Модули синтеза функциональных составляющих первого рода	Информация о функциональных элементах модели первого рода, бинарных связях этих элементов
С_Ф2	Модули синтеза функциональных составляющих второго рода	Информация о функциональных элементах модели второго рода, бинарных связях этих элементов
С_У1	Модули синтеза моделей в целом на основных структурах	Информация о взаимосвязях информационных элементов, функциональных и динамических элементов первого рода
С_У2	Модули синтеза моделей в целом на производных структурах	Информация о взаимосвязях информационных элементов, функциональных и динамических элементов второго рода

Анализ методики формирования и обработки моделей автоматизированных систем согласно МАИТ позволил сформировать взаимосвязи между классами модулей (рис. 3.2). Таким образом, визуальная интегрированная среда будет оболочкой для наполнения программными модулями, являющимися представителями вышеперечисленных классов. Классы модулей должны определять те универсальные компоненты и составляющие их файлы, которые обеспечат как оптимизацию процесса разработки модулей, так и возможность их интеграции.

3.4 Разработка структуры элементов визуальной интегрированной среды

3.4.1 Описание основных конфигурационных элементов ВИС

Визуальная интегрированная среда включает в себя разработанные программные модули поддержки функциональных процедур обработки моделей автоматизированных систем на этапах предпроектного обследования и проектирования АС. В то же время одним из важных требований к ВИС является возможность обновления её функциональных модулей и внедрение новых разработанных.

Для эффективного решения этой задачи, на основании метода моделирования ВИС, её структура среды представляется в виде совокупности конфигурационных элементов на трех уровнях детализации. На каждом уровне составляющими элементами ВИС являются (рис. 3.3):

1. На уровне функциональных модулей: модуль – составляющий элемент среды, реализующий действие или совокупность действий процедуры этапа создания АС.
2. На логическом уровне: универсальный или специализированный компонент (информационный, функциональный или интерфейсный) – логическая составляющая модуля, реализующая структуру данных, доступ, обработку или визуальный элемент.
3. На физическом уровне: универсальный или специализированный файл (информационный, доступа, процедуры или интерфейса) – реализация части компонента на уровне файловой системы.

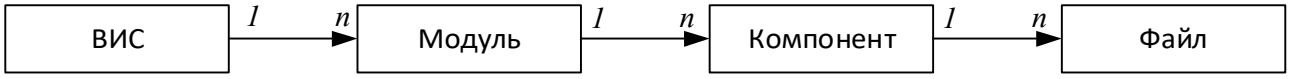


Рис. 3.3 Структура конфигурационных элементов ВИС

Исходя их этого, под развитием визуальной интегрированной среды понимается процесс её функционирования в режиме разработки будущей версии и последующий процесс конфигурирования, позволяющий поэтапно наращивать её функциональные возможности.

С учетом особенностей модели визуальной интегрированной среды, в составе функционального модуля ВИС могут быть компоненты двух типов (рис. 3.4): специализированные, входящие в состав только одного модуля ВИС и реализующие уникальные для модуля информационные, функциональные, интерфейсные конструкции, и универсальные – общие для совокупности модулей и содержащие реализацию унифицированных структур моделей МАИТ. Конфигурирование файлов универсального компонента – универсальных

файлов, – приводит к изменениям в структуре всех модулей, к которым относится компонент.

Каждый конфигурационный элемент ВИС представлен упорядоченным множеством его версий (рис. 3.5). Версия сложного конфигурационного элемента является совокупностью версий элементов, составляющих его.

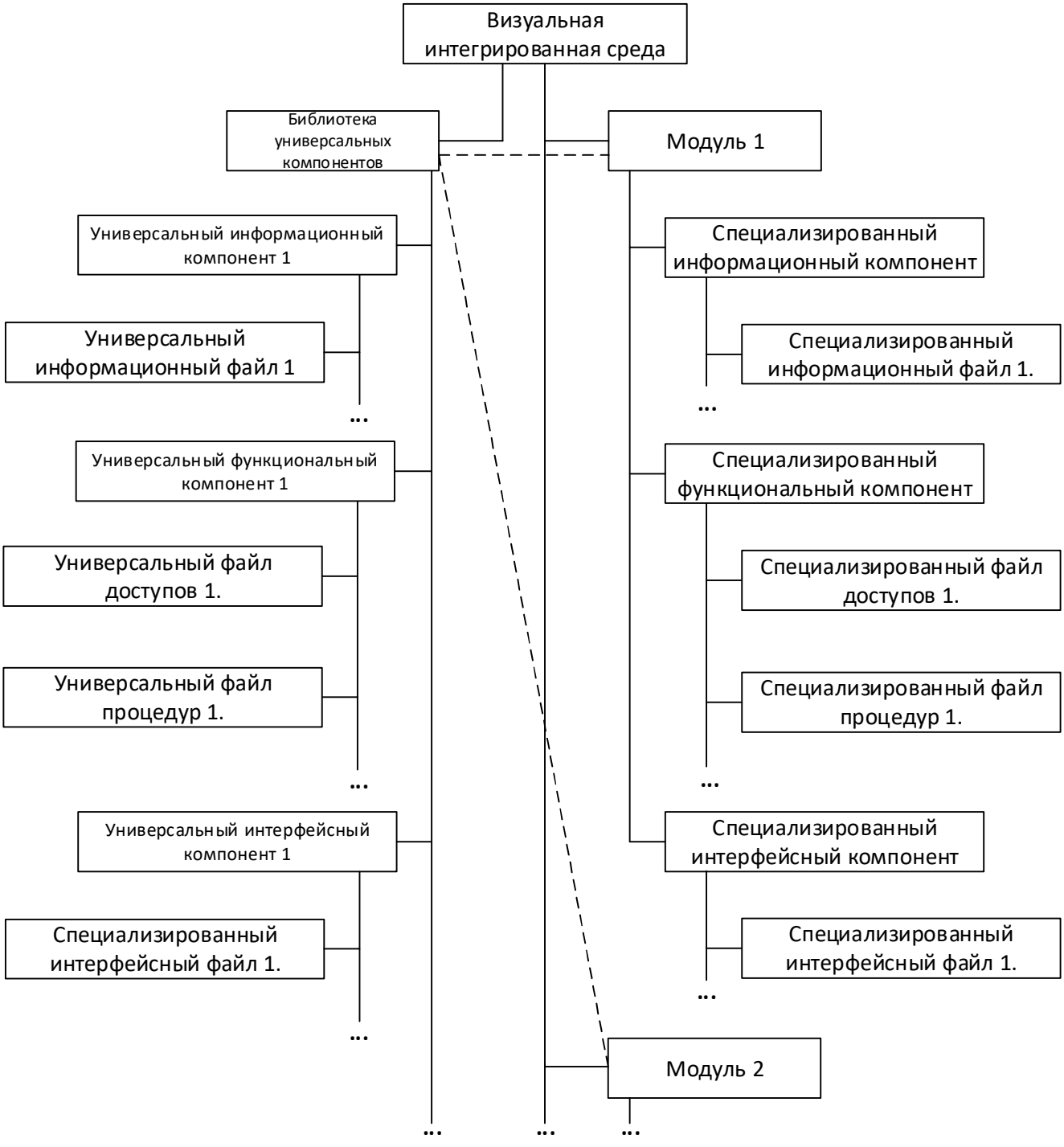


Рис. 3.4 Пример организации конфигурационных элементов ВИС

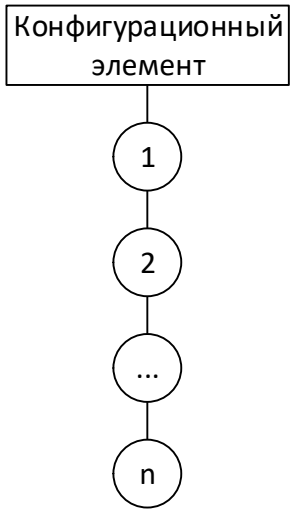


Рис. 3.5 Версии (ветвь) конфигурационного элемента

Особенностью структуры ВИС является наличие библиотеки универсальных компонентов (БУК) – единого репозитория, наполненного множеством универсальных файлов. Управляемая разработчиками ВИС, она является инструментов, позволяющим выполнять эффективное развитие среды, сокращая факты многократной реализации похожих информационных, алгоритмических и интерфейсных конструкций. При разработке собственных модулей разработчики используют уже реализованные элементы из БУК, которые определены классом создаваемого модуля. Таким образом, внесение изменений в БУК в процессе конфигурирования среды вызовет и изменения в модулях среды.

Версия ВИС формируется из совокупности версий конфигурационных элементов ВИС (рис. 3.7, 3.6). Специалист по конфигурации для каждого модуля, представленным совокупностью файлов, формирует перечень актуальных версий. Совокупность версий файлов всех типов определяет версию функционального модуля.

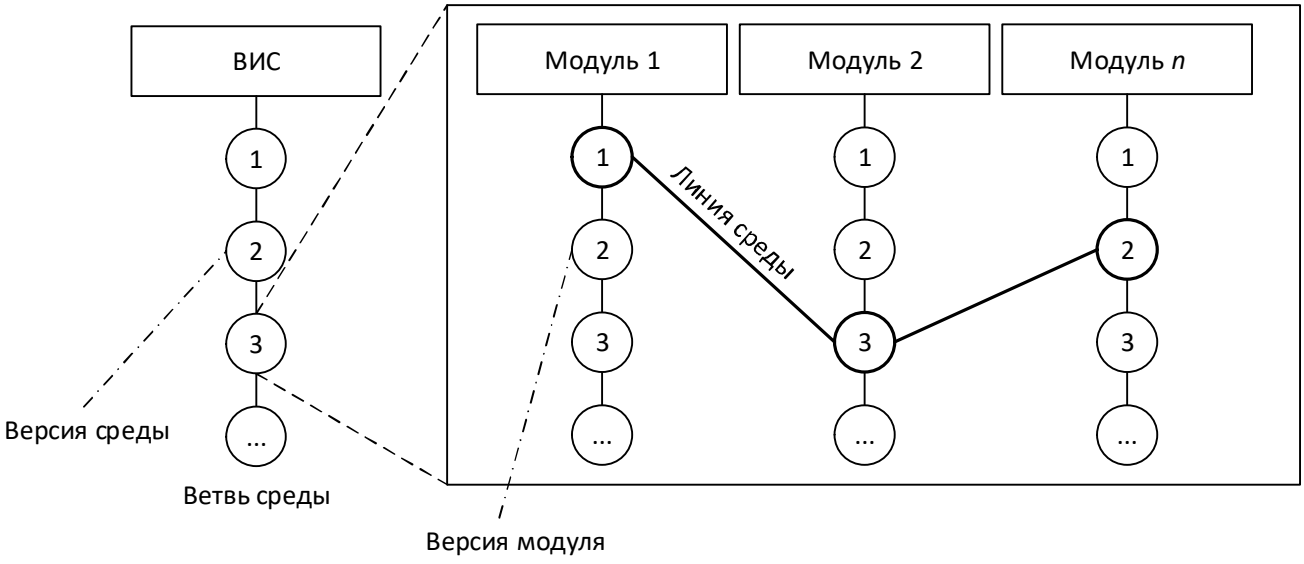


Рис. 3.6 Графическое представление конфигурации ВИС

Версия функционального модуля формируется из двух множеств версий файлов,

образующих линии модуля – линию универсальной и линию специализированной части. Фактически, две линии определяют версию функционального модуля. Изменять универсальную линию необходимо с осторожностью, так как она может входит в состав версий различных функциональных модулей.

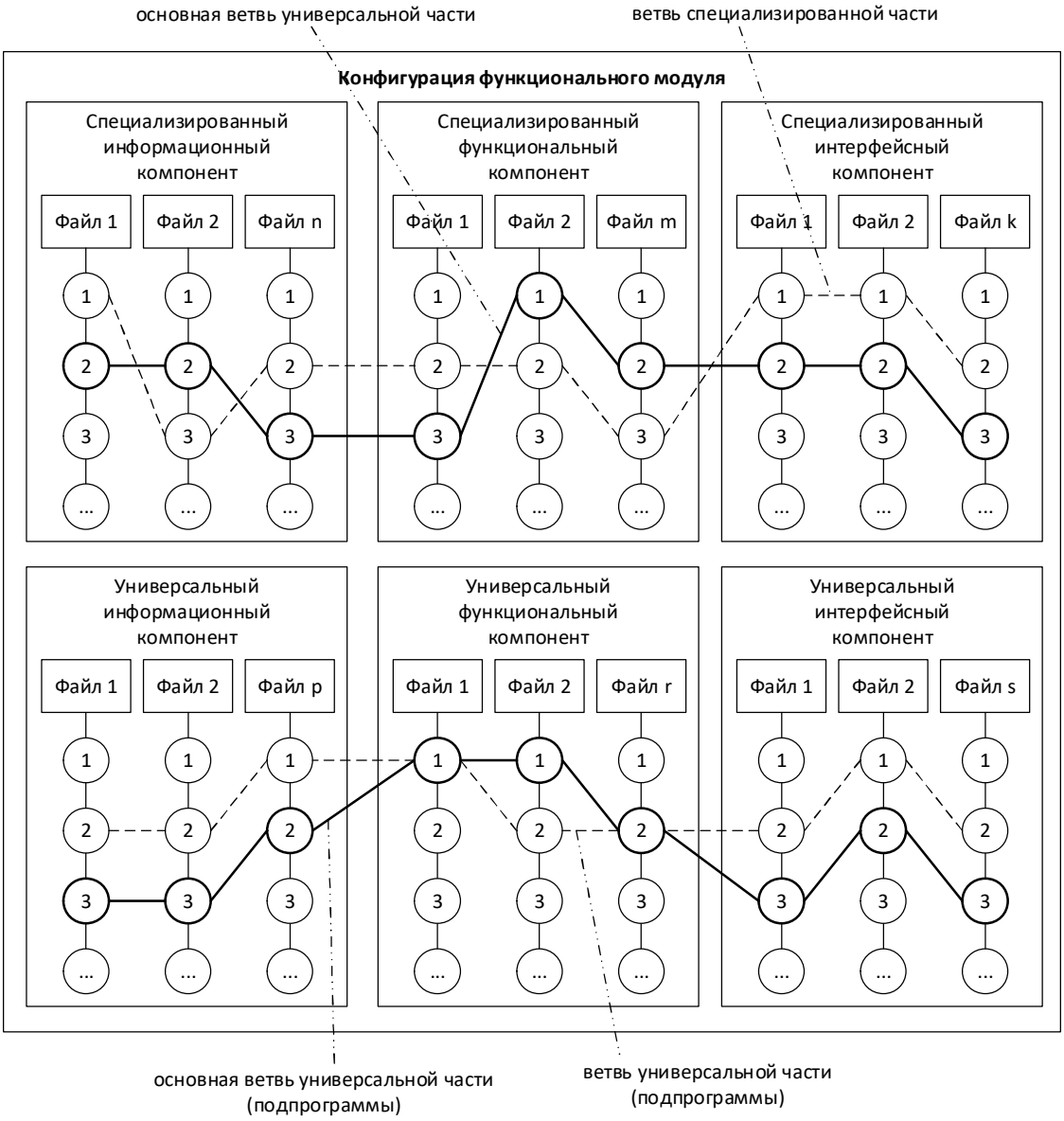


Рис. 3.7 Графическое представление конфигурации модуля ВИС

3.4.2 Структурная диаграмма процесса развития ВИС

Для более подробного описания взаимосвязей объектов процесса развития была разработана ER-модель процесса развития ВИС в нотации UML 2.5.1 (рис 3.8, 3.9), созданная на основе концептуальной модели (приведена в главе 2) .

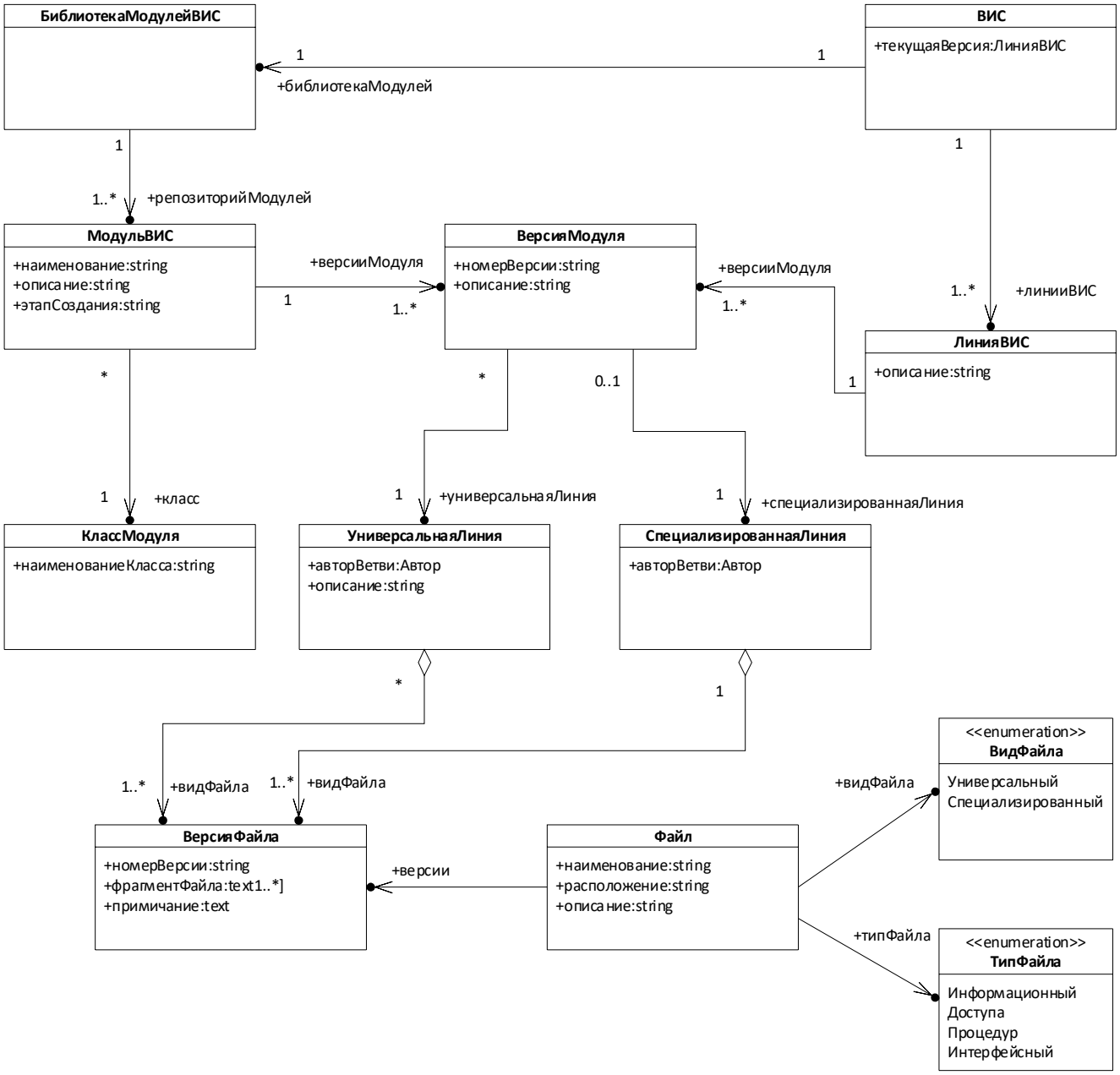


Рис. 3.8 Диаграмма классов процесса развития ВИС (начало)

Описания основных классов разработанной модели:

«ВИС» – совокупность всех версий визуальной интегрированной среды.

«БиблиотекаМодулейВИС» – структурированное множество всех модулей для визуальной интегрированной среды.

«ЛинияВИС» – версия визуальной интегрированной среды, представляющая совокупность версий модулей. Содержит описание версии.

«ВерсияМодуля» – версия функционального модуля ВИС, содержащая универсальную и специализированную линию.

«МодульВИС» – функциональный модуль среды.

«Класс модуля» – класс модуля, описывающий его место в составе среды.

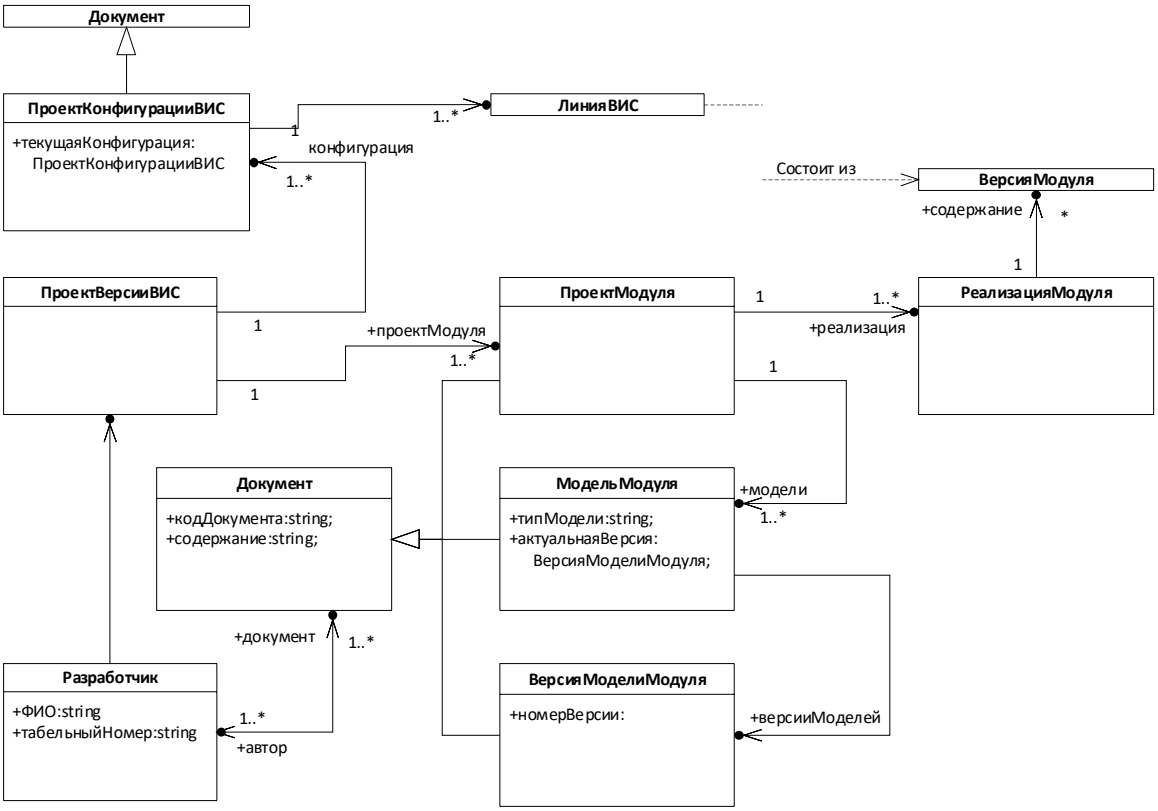


Рис. 3.9 Диаграмма классов процесса развития ВИС (завершение)

«УниверсальнаяЛиния» – множество версий файлов, входящих в состав модуля. Линия может входить в состав всех модулей одного класса.

«СпециализированнаяЛиния» – множество версий файлов, входящих в состав модуля, уникальна для каждого модуля.

«ВерсияФайла» - составляющая модуля, реализующая часть компонента объектно-ориентированного подхода.

«Файл» - совокупность версий файла.

Сформированные информационные модели позволяют перейти к разработке системы действий и формам фиксации их результата.

3.5 Разработка системы действий методики управления конфигурацией ВИС

Система действий представлена в графической нотации функциональных диаграмм в соответствии с МАИТ и сформирована в ПК «ИС-2» (реализация ВИС). Описание нотации приведено в главе 4 (стр. 126). Процесс развития ВИС носит циклический характер. Каждая новая итерация – разработка и ввод в эксплуатацию новой версии ВИС. Корневой цикл процесса

обеспечивает постоянный процесс поэтапного обновления состава среды. Средой выполнения процесса развития ВИС является текущая версия ВИС, за исключением самой первой её версии, которая должна быть сформирована вручную либо существующими средствами поддержки промышленного способа создания в соответствии МАИТ. Каждый программный модуль среды является автоматизированной системой, направленный на автоматизацию части процесса создания АС.

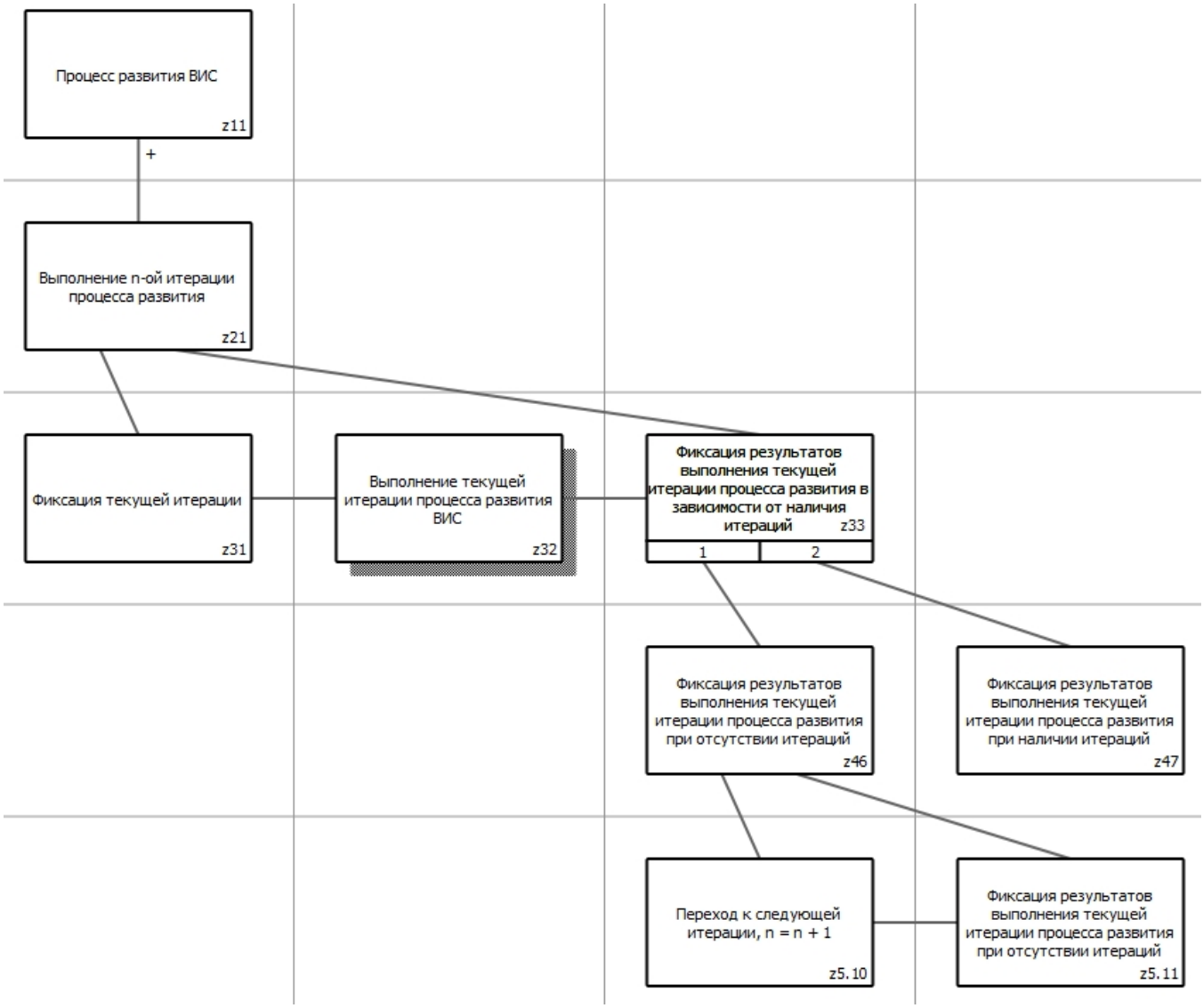


Рис. 3.10 Корневой цикл методики управления развитием ВИС

На каждой итерации процесса развития выполняются следующие действия (рис 3.11):

- Планирование хода текущей итерации – планирование работ исполнителей и информационных ресурсов для формирования новой версии ВИС.
- Реализация новой функциональности ВИС – разработка новых и модернизация текущих модулей ВИС. Важной составляющей действия является унификация текущих специализированных решений, позволяющая оптимизировать текущую структуру ВИС, за счет снижения количества специализированных файлов, и

расширить возможности библиотеки универсальных компонентов.

– Контроль процессов реализации новой функциональности – осуществление управленческих действий над коллективом разработчиков, занимающихся развитием среды.
– Внедрение новой функциональности – переход с текущей версии ВИС на новую, разработанную в рамках текущей итерации процесса развития.

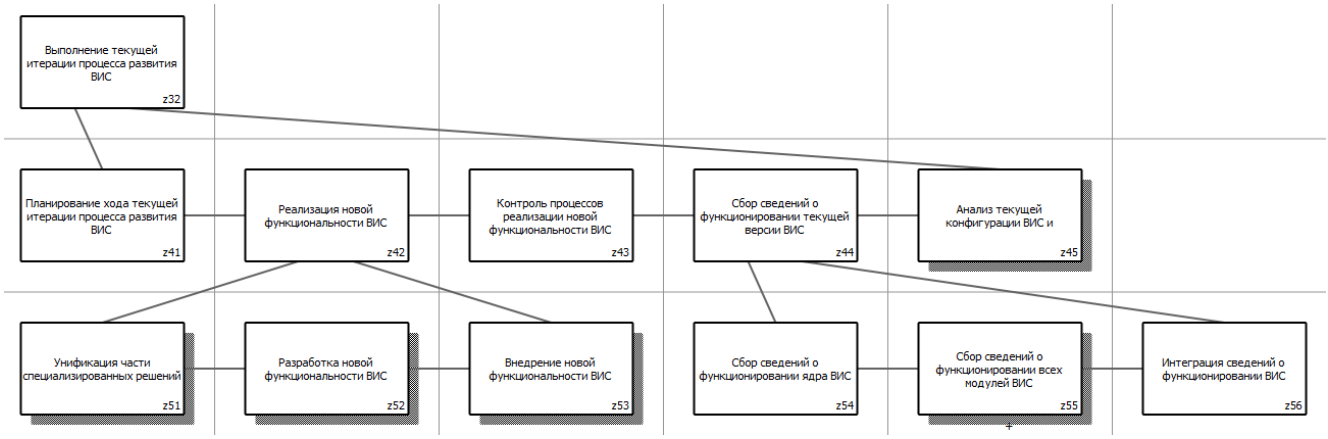


Рис. 3.11 Декомпозиция итерации процесса развития ВИС

Разработка новых модулей, как и их модификация (рис. 3.13, 3.12) выполняется в соответствии с методологией автоматизации интеллектуального труда. Отличие этих двух процедур в исходных данных – разработка новых модулей начинается «с нуля», в то время как модификация подразумевает наличие предыдущих версий проектных решений.

При разработке или модификации функциональных модулей ВИС необходимо выполнить следующие этапы:

– Начальное моделирование – сформировать/модернизировать модель традиционного способа решения подзадачи автоматизации процесса создания АС.
– Концептуальное моделирование – сформировать/модернизировать модель системы знаний подзадачи автоматизации процесса создания АС.
– Инфологическое моделирование – сформировать/модернизировать модель конструкции средства поддержки подзадачи автоматизации процесса создания АС.
– Даталогическое моделирование – сформировать/модернизировать модель технологической среды средства поддержки подзадачи автоматизации процесса создания АС на логическом и физическом уровнях. Производить выбор программно-технических средств реализации в данном случае не требуется, так как средства будут определены техническими требованиями самой ВИС.
– Реализация модуля – программирование и отладка новой версии модуля. Результат – сформированные файлы, представляющие готовый функциональный модуль в

памяти ЭВМ и пригодные для внедрения в состав новой версии среды.

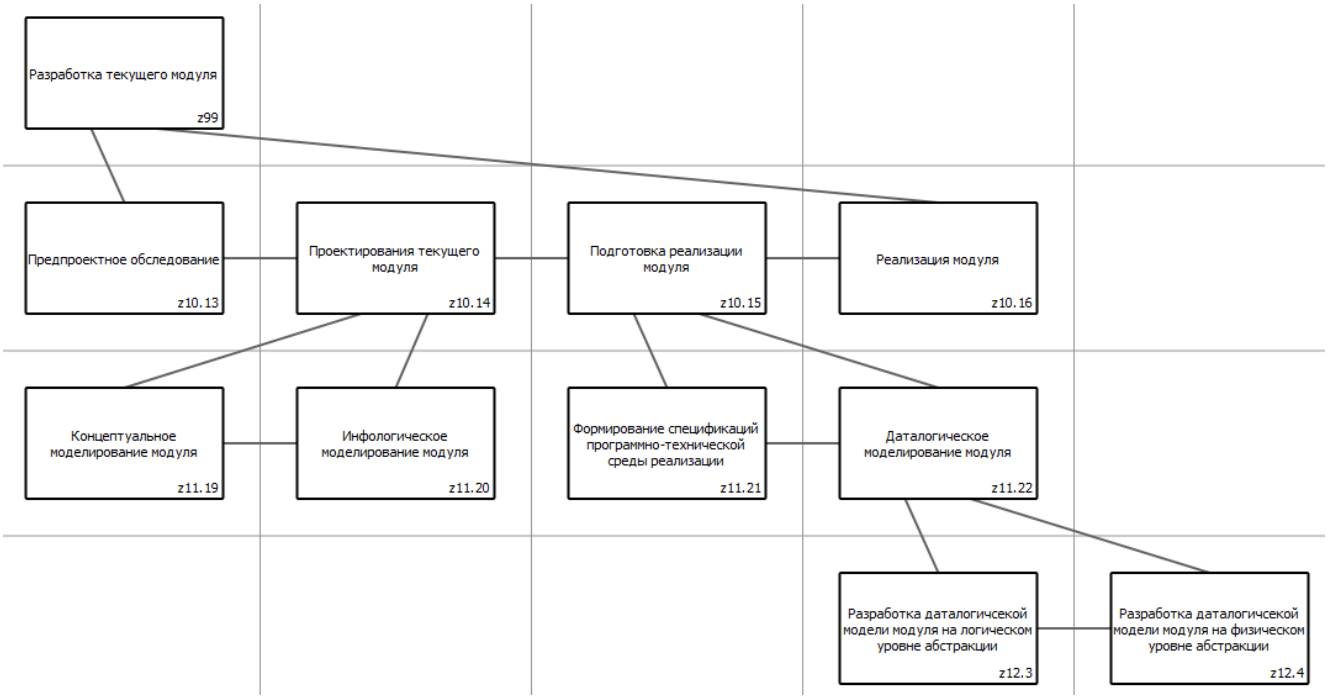


Рис. 3.12 Декомпозиция процесса разработки модуля ВИС

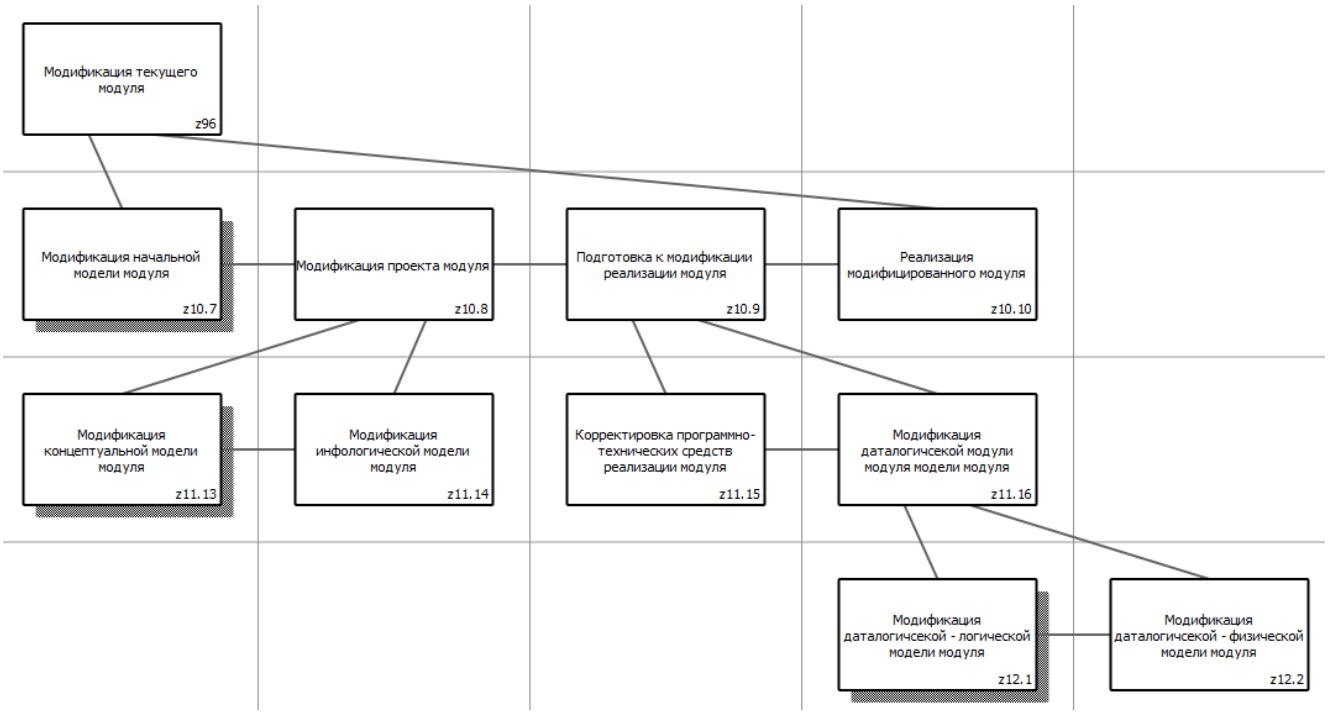


Рис. 3.13 Декомпозиция процесса модификации модуля ВИС

После завершения процессов разработки новых и модификаций существующих модулей их необходимо зафиксировать новую версию ВИС. Все изменения среды происходят на уровне файловой системы. Процесс изменения её файловой структуры называется конфигурированием (рис. 3.14).

В рамках конфигурирования происходит работа с версиями различных конфигурационных элементов – модулей, универсальных и специализированных файлов.

Универсальные и специализированные файлы ВИС могут представлены в файловой системе различными версиями. Необходимость ведения версий файлов обусловлена следующими обстоятельствами:

- для информационных компонентов может возникнуть необходимость модификации структур данных, используемых в модулях ВИС;
- для функциональных компонентов может возникнуть необходимость оптимизации отдельных процедур или запросов;
- для интерфейсных компонентов может возникнуть необходимость изменения визуального отображения (стиля визуализации) или компоновки элементов графического интерфейса пользователя.

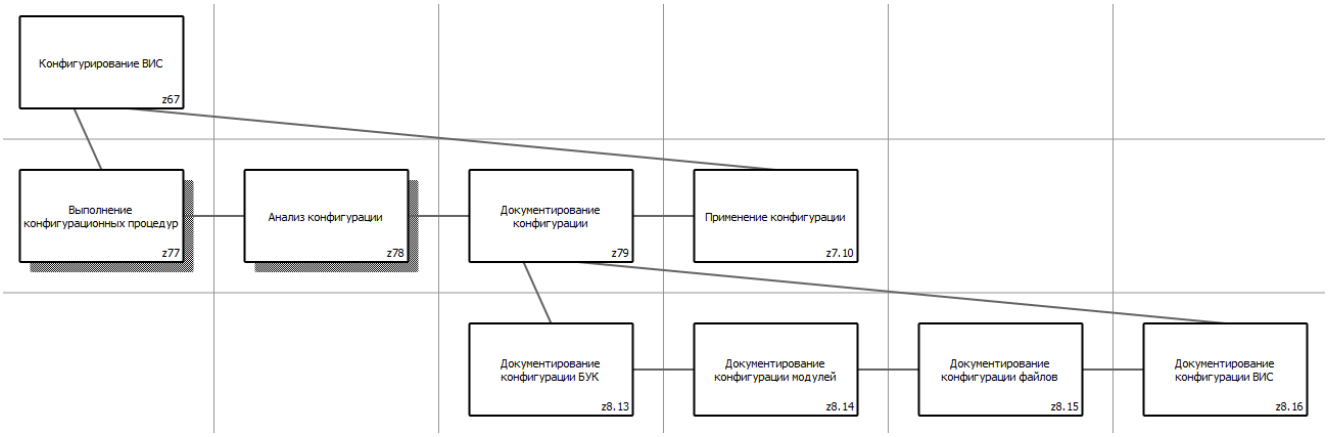


Рис. 3.14 Декомпозиция процесса конфигурирования ВИС

При работе с файлами среды специалист по конфигурации совершает следующие процедуры (рис. 3.15):

- Конфигурация файлов ВИС – работа с файлами ВИС и их версиями. В рамках этой процедуры специалист:
  - Добавляет новые файлы в среду, при этом размещает файл в репозитории ВИС, фиксирует информацию о назначении и типе файла, и описание его первой версии.
  - Работает с версиями существующих файлов.
  - Удаляет файл, при этом удаляются все версии файла из всех конфигураций, в которые они входят. Также удаляется вся справочная информация о файле и его версиях.
- Конфигурация линии библиотеки универсальных компонентов – формирование массива версий универсальных файлов, входящих в среду. Линии могут быть добавлены, изменены или удалены. Все изменения необходимо фиксировать в конфигурационной спецификации.

– Конфигурирует специализированные ветви модулей ВИС – формирует и управляет множеством версий модулей ВИС, определяющими версию среды.

После завершения процесса формирования новой версии среды конфигурация подвергается анализу (рис. 3.16).

Полный вариант методики управления развитием ВИС, сформированный в ПК «ИС-2» (реализация ВИС, описанная в Главе 4) в графическом и табличном представлении приведен в приложении Б. Методика описана в графической нотации системы предметных действий, в соответствии с МАИТ [37] и сформирована модулями документирования функциональной структуры, входящим в первую версию ВИС.

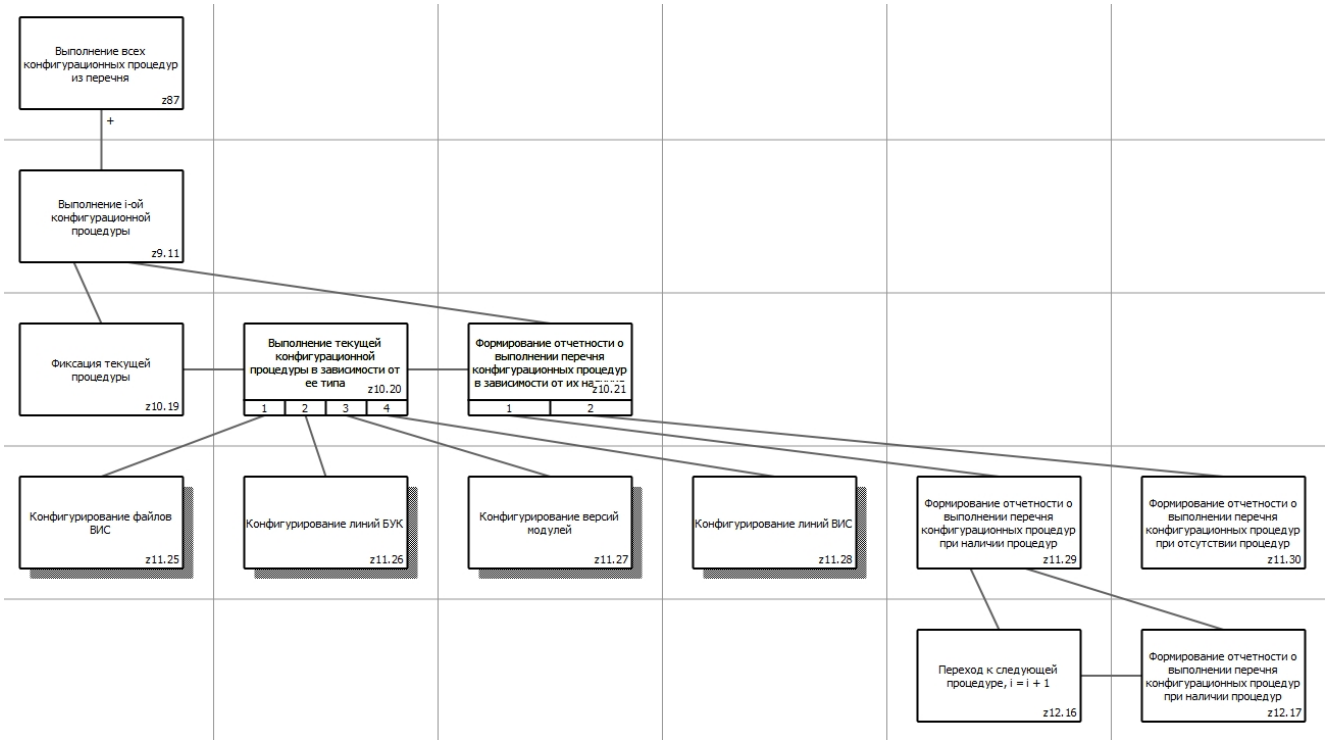


Рис. 3.15 Декомпозиция процесса применения конфигурационных процедур к модулю ВИС

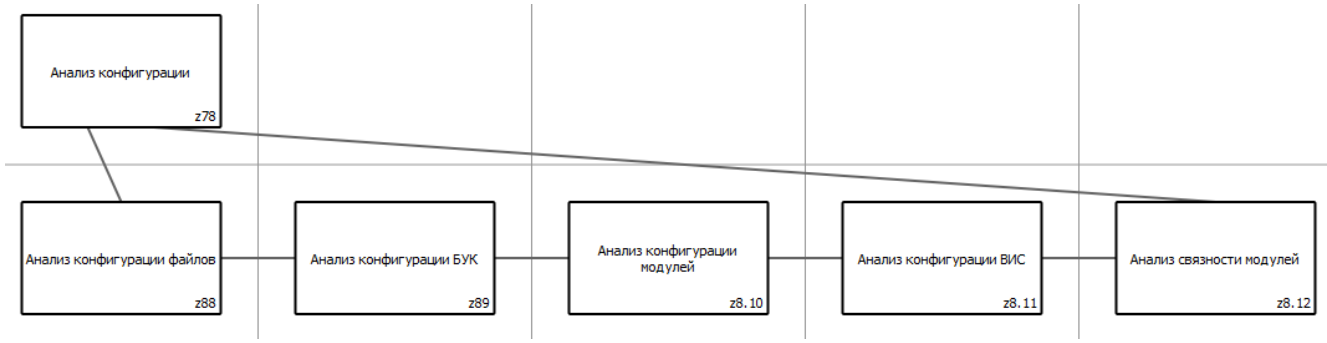


Рис. 3.16 Декомпозиция процесса анализа конфигурации ВИС

3.6 Разработка спецификаций описания конфигурации ВИС

Результаты выполнения процедур этапа развития должны быть зафиксированы в спецификациях, представленные в таблицах 3.2 - 3.12.

Для каждой спецификации сформулированы правила из заполнения. При разработке спецификаций подразумевалось их хранение в реляционных базах данных. Ключевые поля отмечены жирным шрифтом.

Таблица 3.2 Спецификация К1. Описание файлов

Код файла	Имя файла	Тип Файла	Назначение файла	Описание файла	Примечание

Таблица 3.3 Спецификация К2. Описание версий файлов

Код версии файла	Код файла	Расположение версии файла	Описание версии файла	Примечание

Таблица 3.4 Спецификация К3. Описание универсальных линий ВИС

Код линии	Наименование линии	Описание линии	Примечание

Таблица 3.5 Спецификация К3.1. Описание содержания универсальных линий ВИС

Код линии	Код версии файла	Примечание

Таблица 3.6 Спецификация К4. Конфигурация классов ВИС

Код конфигурации класса ВИС	Класс ВИС	Код линии	Код конфигурации ВИС	Примечание

Таблица 3.7 Спецификация К5. Описание специализированных линий ВИС

Код линии	Наименование линии	Описание линии	Примечание

Таблица 3.8 Спецификация К5.1. Описание содержания специализированных линий

Код линии	**Код версии файла**	Примечание

Таблица 3.9 Спецификация К6. Описание функциональных модулей ВИС

Код модуля	Наименование модуля	Класс модуля	Описание модуля	Назначение модуля	Примечание

Таблица 3.10 Спецификация К7. Описание содержания версий модулей ВИС

Код версии модуля	Код специализированной линии	Наименование версии модуля	Статус версии модуля	Описание версии модуля	Примечание

Таблица 3.11 Спецификация К8. Описание линий ВИС

Код линии ВИС	Наименование линии	Описание линии	Примечание

Таблица 3.12 Спецификация К8.1. Описание содержания линий ВИС

Код линии ВИС	**Код версии модуля ВИС**	Примечание

Правила заполнения таблицы спецификации К1. Описание файлов:

Входящий в состав модуля файл, предоставляется разработчиком специалисту по конфигурированию ВИС для размещения в репозитории.

1. Поле «Код файла» содержит уникальный идентификатор файла: <ВидФайла>.<НомерФайла>. Вид файла указывается буквой «У» для универсальных файлов, буквой «С» для специализированных. Номер файла – натуральное число.
2. Имя файла определяется разработчиком файла в виде строки с максимальной длинной в 255 символов.
3. Тип файла указывается обозначением – «И» для информационного файла, «Д» для файла доступов, «П» для файла процедур, «Г» для файла графического интерфейса. Допускается использование файлов комбинированных файлов, сочетающих несколько типов. В этом случае необходимо указывать несколько обозначений слитно.

4. В поле «Описание» заносится описание файла в свободной форме. Длина описания не должна превышать 1024 символа.
5. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения спецификации К2. Описание версий файлов:

В таблицу заносится информация о различных версиях файлов. С точки зрения файловой системы именно версии являются физическими файлами, а сами файлы ВИС из предыдущей спецификации –логические объединения множества версий.

1. Код версии файла указывается натуральном числом.
2. В поле «Код файла» заносится код из спецификации файлов, версия которого ассоциирована с текущей строкой.
3. В поле «Расположение» заносится путь до файла, который представляет версию в формате используемой операционной системы.
4. В поле «Описание» следует указывать ключевые особенности версии файла. Максимальная длина 1024 символа.
6. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения таблицы спецификации К3. Описание универсальных линий ВИС:

1. Код линии должен являться натуральным числом.
2. Наименование линии указывается разработчиком и не должно превышать длины в 255 символов.
3. Описание линии указывается разработчиком в свободной форме. Описание не должно превышать 1024 символа.
4. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения таблицы спецификации К4. Конфигурация классов ВИС:

Классы ВИС увязывают конфигурации универсальных частей модулей определенных классов. Каждый класс должен иметь только одну конфигурацию в рамках версии ВИС.

1. Код линии должен являться натуральным числом.
2. Наименование линии указывается разработчиком и не должно превышать длины в 255 символов.
3. Описание линии указывается разработчиком в свободной форме. Описание не должно превышать 1024 символа.
4. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения таблицы спецификации К5. Описание специализированных линий ВИС:

Специализированные линии являются именными срезами версий файлов и представляют собой основу модуля. Специалист по конфигурации заносит в спецификацию информацию о линиях, согласно документации, предоставленной разработчиками.

1. Код линии должен являться натуральным числом.
2. Наименование линии указывается разработчиком и не должно превышать длины в 255 символов.
3. Описание линии указывается разработчиком в свободной форме. Описание не должно превышать 1024 символа.
4. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения спецификации К6. Описание функциональных модулей ВИС:

Версии модулей предоставляются разработчиками для включения в состав конфигурации ВИС. Специалист по конфигурации указывает в спецификации сведения, согласно документации, полученной от разработчика.

1. Код модуля должен быть обозначен натуральным числом.
2. Наименование модуля определяется разработчиком, не должно превышать 255 символов.
3. Класс модуля указывается согласно классификатору. Определяет универсальную линию, входящую в состав модуля.
4. Описание модуля определяется разработчиком, указывается в свободной форме, может содержать рекомендации по конфигурированию и ссылки на сопроводительную документацию. Размер поля не должен повышать 65535 символов.
5. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения спецификации К7. Описание содержания версий модулей ВИС

Спецификация описывает версии модулей, их соотношения со специализированными линиями. Каждый модуль должен содержать хотя бы одну версию. «Пустой» модуль не может быть добавлен в конфигурацию ВИС.

1. Код версии модуля должен быть представлен натуральным числом.
2. Код специализированной линии указывается согласно спецификации К5.
3. Наименование версии модуля указывается разработчиком и не должно превышать 255 символов.
4. Статус версии указывается буквенным обозначением: «Р» – для ранней версии модуля,

находящейся в разработке, «Б» – для бета-версии модуля, «Ф» для версии модуля, пригодной для эксплуатации.

5. Описание версии модуля формируется разработчиком. Рекомендуется указывать технические особенности и ключевые отличительные черты концертной версии.
6. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

Правила заполнения спецификации К8. Описание линий ВИС:

Линии ВИС описывает конфигурацию версии ВИС. Линии формируются специалистами по конфигурации из версий модулей.

1. Код линии ВИС должен быть указан натуральным числом.
2. Наименование линии ВИС не должно превышать 255 символов.
3. Описание линии должно содержать основные технические особенности версии ВИС и описания её назначения.
4. «Примечание» заполняется специалистом по конфигурации в свободной форме при необходимости.

3.7 Выводы по главе

На основании разработанного метода моделирования ВИС был сформирован классификатор модулей ВИС, позволяющий систематизировать и выявить взаимосвязи между средствами поддержки этапов предпроектного обследования и проектирования АС.

Разработанный классификатор модулей позволил осуществить разработку методики управления развитием ВИС, обеспечивающей поэтапную модернизацию её функциональных модулей.

Методика обеспечивает регламентацию процессов управления разработкой АС в области автоматизации интеллектуального труда и позволяет осуществлять создание новых и модернизацию существующих средств поддержки этой деятельности в составе ВИС.

Разработанная методика управления развитием визуальной интегрированной среды позволили перейти к процессу создания прототипа первой версии ВИС.

ГЛАВА 4
ОПИСАНИЕ РЕАЛИЗАЦИИ ВИЗУАЛЬНОЙ ИНТЕГИРОВАННОЙ СРЕДЫ

4.1 Описание средств реализации ВИС

4.1.1 Описание Microsoft Visual Studio 2015

В качестве среды реализации ВИС была выбрана интегрированная среда разработки Microsoft Visual Studio 2015. Среда содержит комплекс средств, позволяющих производить реализацию и отладку программного обеспечения. Visual Studio поддерживает множество языков программирования, среди которых C++, C#, JavaScript, F#, Visual Basic. Основные возможности Visual Studio, используемые при реализации визуальной интегрированной среды: [49]

- Поддержка C++, высокоуровневый объектно-ориентированный язык программирования, но позволяющий проводить оптимизацию на более низком уровне, в отличии от остальных, представленных в Visual Studio. Объектно-ориентированная парадигма, включающая инкапсуляцию, полиморфизм и наследование, позволит реализовать модульное приложение и систему универсальных компонентов.
- Система организации проектного решения и проектов внутри него позволит организовать разработку модульного приложения как совокупности проектов и автоматизировать сборку и отладку визуальной интегрированной среды в целом.
- Поддержка Team Foundation Server и сервиса Visual Studio Online позволит организовать коллективный процесс реализации ВИС и осуществлять управление им.
- Имеется возможность миграци на более поздние версии Visual Studio (2017+), поддерживающие реализацию приложений под операционную систему Linux.

4.1.2 Описание Qt 5.7.1

Qt 5.7.1 – кроссплатформенный инструментарий, поддерживающий реализацию программных продуктов на языке C++. Основными достоинствами Qt являются: разработка под различные платформы (Windows, Linux) и ориентированность на разработку графических

приложений. [50]

Модули Qt, использованные в реализации ПК «ИС-2»:

1. QtCore – содержит базовую функциональность Qt, не связанную с графическим интерфейсом пользователя (ГИП);

2. QtWidgets – базовый модуль Qt для работы с ГИП; модуль содержит как набор стандартных элементов графического интерфейса, так и позволяет создавать новые – как на основе базовых, так и уникальные. Наличие и удобная реализация данной возможности было основополагающим фактором выбора библиотеки Qt для создания ПК «ИС-2».

3. QtXML – содержит средства для работы с форматом хранения иерархических структур XML; формат был выбран для хранения результатов моделирования в ПК «ИС-2».

4. QtNetwork – предоставляет возможность реализации программ, использующих сетевое взаимодействие через протоколы TCP/UDP и более высокоуровневые.

5. QWebEngine - является модулем для Qt основанным на Chromium, бесплатном веб-браузере, разрабатываемом сообществом (в том числе компаниями Яндекс, Google, Nvidia и др.); необходимость использования веб-браузера обусловлена тем, что формируемые в процессе функционирования программы документы, содержащие преимущественно табличные данные удобно формировать и визуализировать средствами языка разметки HTML; преимуществом также является возможность экспорта формируемых документов в формате веб-страниц в различные текстовые редакторы.

4.2 Описание программного комплекса «ИС-2»

Визуальная интегрированная среда, включающая средства поддержки функциональных процедур МАИТ и средства управления её развитием, реализована в виде программного комплекса (ПК) «ИС-2». На текущий момент времени ПК «ИС-2» предоставляет поддержку процессов начального, концептуального и инфологического моделирования, а также содержит ряд средств наращивания собственной функциональности.

Средства поддержки процедур реализованы в виде функциональных модулей, которые, в свою очередь, базируются на универсальных компонентах. Структура ПК «ИС-2» представлена на рисунке 4.1. Функциональные модули, входящие в ПК «ИС-2», формируют различные версии – «ИС-2. Начальное моделирование», «ИС-2. Концепт» и «ИС-2. Инфология», направленные на поддержу двух различных этапов создания автоматизированных систем. Для повышения эффективности реализации программного комплекса в его рамках была разработана библиотека

универсальных компонентов, содержащая реализации общих для различных этапов моделирования структур, процедур и интерфейсов. На рисунке 4.1 отображены зависимости между функциональными модулями и универсальными компонентами в виде стрелок.

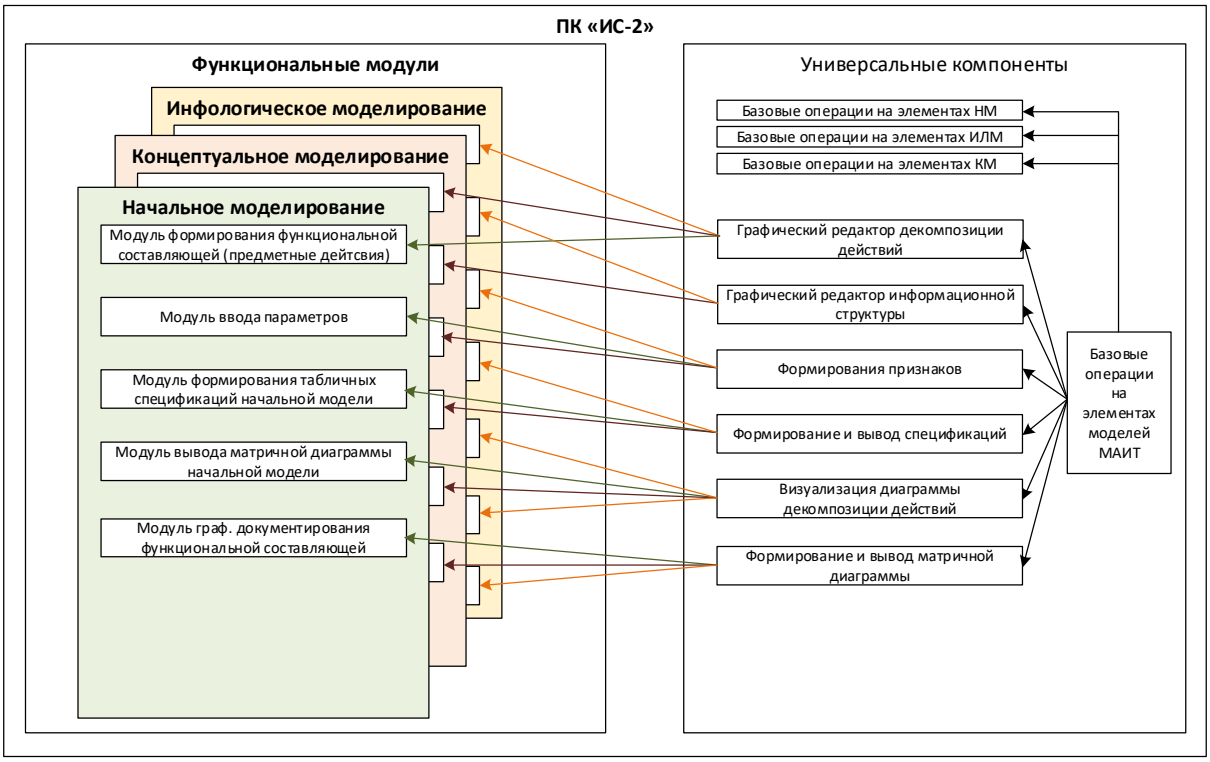


Рис. 4.1 Структура ПК «ИС-2»

Данные версии ИС-2 построены на следующих универсальных компонентах.

Базовые операции на элементах НМ, КМ, ИЛМ – библиотека основных структур данных для описания, хранения и манипулирования элементами. Библиотека используется другими модулями, обеспечивая унифицированный доступ к модельным данным для разработчиков расширений «ИС-2».

Графический редактор декомпозиции действий – базовый редактор для разработки модулей визуального редактирования функциональных или динамических составляющих моделей АС. Содержит базовую функциональность по визуализации элементов диаграмм, позволяет осуществлять управление визуализацией и может быть настроен на все модели МАИТ.

Графический редактор информационной структуры – базовый редактор для разработки модулей визуального редактирования статических составляющих моделей АС. Содержит базовую функциональность по визуализации элементов моделей МАИТ и их статических взаимосвязей всех типов, позволяет осуществлять управление визуализацией и может быть настроен на концептуальную или инфологическую модели.

Формирование признаков – базовый табличный редактор признаков. Позволяет разрабатывать модули для манипулирования множеством параметров, предметных категорий класса «признак» или именованных структурах единиц типа «информационный атрибут» пятого

уровня.

Формирование и вывод спецификаций – библиотека для разработки модулей табличного документирования моделей.

Визуализация диаграммы декомпозиции действий – библиотека для разработки модулей графического документирования функциональных и динамических составляющих модели.

Формирование и вывод матричной диаграммы – библиотека для разработки модулей документирования модели в целом.

4.3 Проектирование структур данных ПК «ИС-2»

В качестве основной структуры данных в объектно-ориентированном программном продукте применяется структура на классах. В разделе представлены разработанные структуры классов для хранения и обработки моделей в ПК «ИС-2» в нотации UML 2.5.1. Представленные классы являются основами для универсальных компонентов, на базе которых разрабатываются функциональные модули для среды, и предоставляют разработчикам средства для создания собственной логики.

Общий принцип, на котором построено дерево классов, – наследование, т.е. вынесение общих атрибутов и методов объектов моделей МАИТ в более общие классы - родители. На рисунке 4.2 представлена структура наследований для классов моделей. Абстрактные классы выделены курсивным шрифтом. Класс «Модель» представляет самые общие структуры без их дифференциации по компонентам (составляющим) моделей. Класс «ОбщаяМодель» уже вводит составляющие, присущие конкретным моделям. Классы «НачальнаяМодель», «КонцептуальнаяМодель» и «ИнфологическаяМодель» уже содержат конкретные структуры данных для каждой модели.

Рассмотрим более детально структуру классов, описывающих модель верхнего уровня абстракции (рис 4.3).

Класс «Модель» – описывает модель самого верхнего уровня абстракции. Он и его составляющие необходимы для единообразного представления структур данных нижних уровней абстракции. Содержит две части: структуру увязки и сложную структуру.

Класс «СложнаяСтруктура» – «структура структур» представляет абстрактную составляющую модели, в общем виде включающую множество структур одного компонента модели.

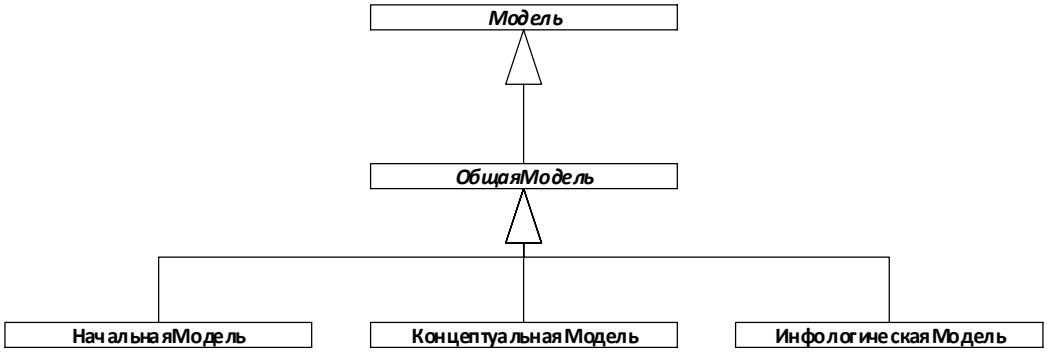


Рис. 4.2 Общая диаграмма классов структур данных ПК «ИС-2»

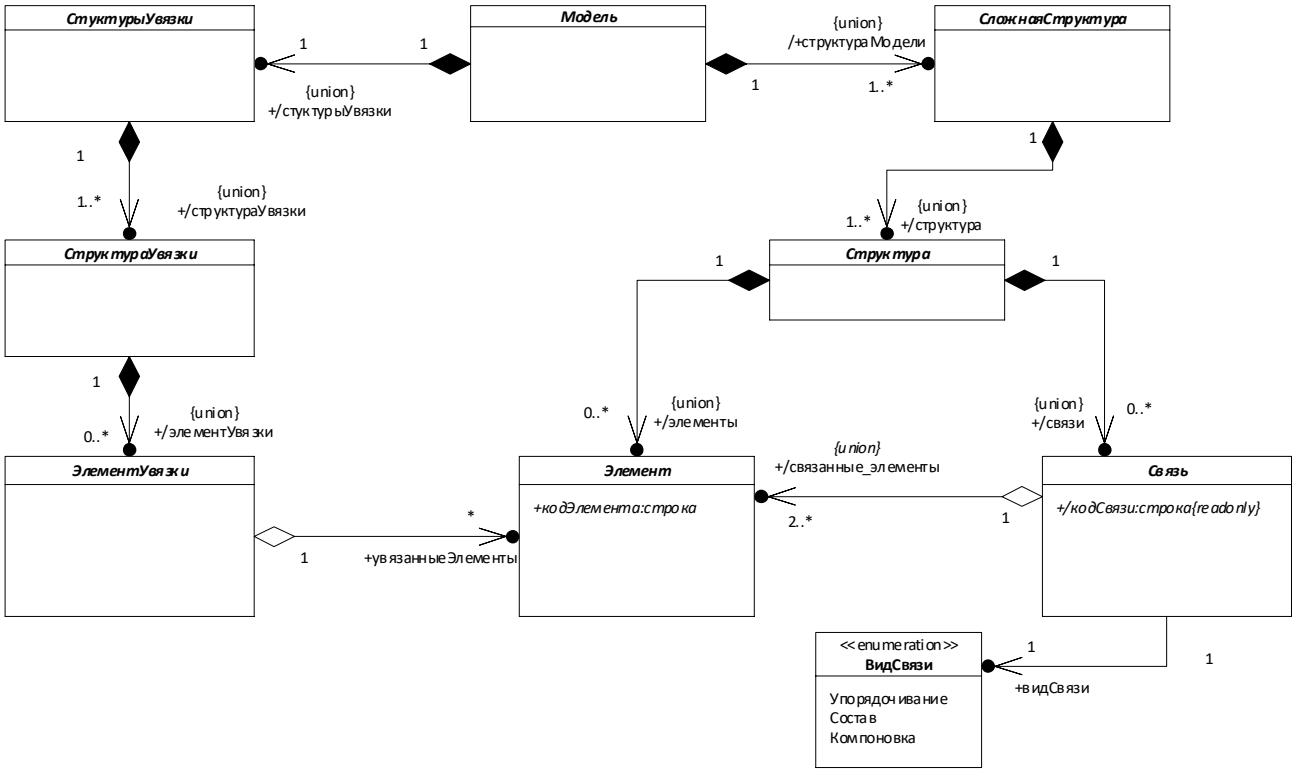


Рис. 4.3 Диаграмма классов модели

Класс «Структура» – основная абстрактная структура модели, составленная в общем виде из множества элементов и их связей.

Класс «Элемент» – абстрактный элемент любой структуры любого компонента модели (например, предметная зависимость первого рода из концептуальной модели или параметр из начальной).

Класс «Связь» – бинарная связь элементов моделей. Множество связей образует иерархическую структуру компонента модели. Связь может быть трех видов:

– упорядочивание – связь между элементами в рамках одного уровня, связь глобального порядка;

- состав – связь «часть - целое» или «общее - конкретное», описывающая иерархию элементов;
- компоновка – связь локального порядка, связывающая однородные элементы в рамках уровня.

Класс «СтруктурыУвязки» – абстрактный класс для организации множества структур увязки.

Класс «СтруктураУвязки» – абстрактный класс, описывающий взаимосвязи между различными компонентами модели.

Класс «ЭлементУвязки» – взаимосвязь между элементами различных компонентов модели.

Формирование общей модели позволило унифицировать дальнейшие структуры данных, сформировать единый подход для описания более конкретных классов.

Следующим уровнем абстракции является уровень общей модели (рис 4.4). Он вводит понятия конкретных компонентов модели и является базовым для реализации структур МАИТ.

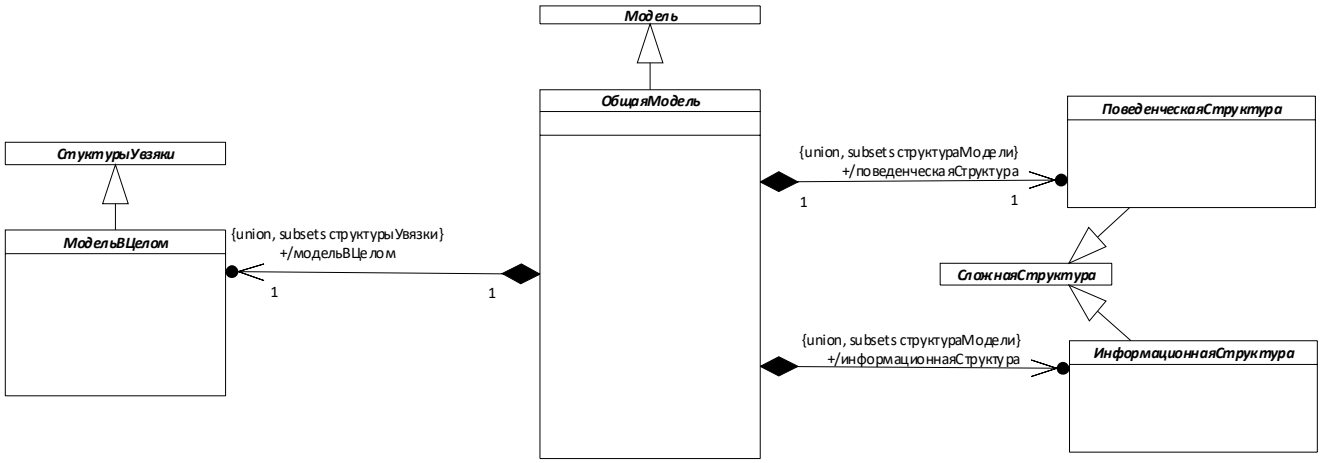


Рис. 4.4 Диаграмма классов. Основные составляющие общей модели

Класс «ОбщаяМодель» – корневой класс для реализации общей модели, содержит сложные структуры под каждый компонент. Наследуется от класса «Модель».

Класс «ПоведенческаяСтруктура» – базовая реализация для динамической и функциональной составляющих моделей МАИТ. Наследуется от класса «СложнаяСтруктура».

Класс «ИнформационнаяСтруктура» – базовая реализация для статической составляющей моделей МАИТ. Наследуется от класса «СложнаяСтруктура».

Класс «МодельВЦелом» – базовая реализация любого компонента увязки моделей. Наследуется от класса «СтруктурыУвязки».

Статический компонент общей модели (рис 4.5) содержит основные структуры, реализации которых присутствуют в МАИТ.

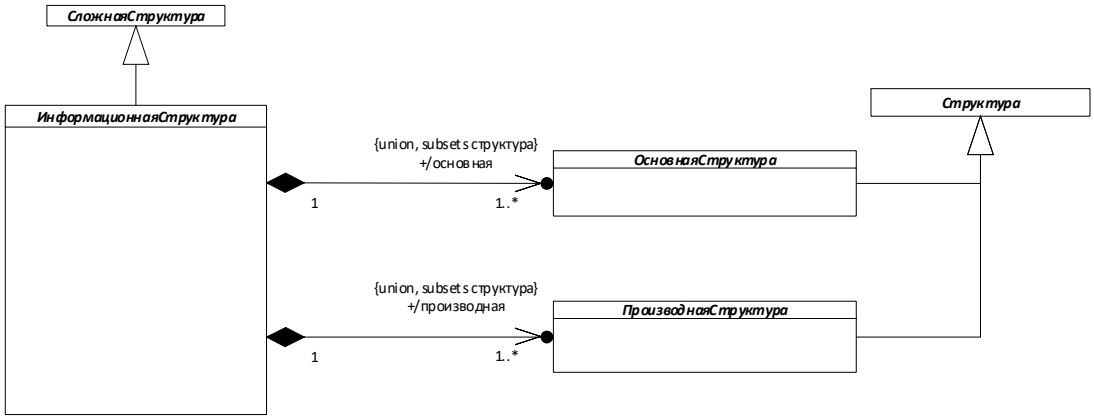


Рис. 4.5 Диаграмма классов. Составляющие информационной структуры

Класс «ОсновнаяСтруктура» (рис. 4.6) – описывает основную статическую структуру любой модели МАИТ как множество элементов и их связей. Является наследником класса «Структура». Реализована в классах «ИнформационнаяСоставляющая» для начальной, «СтатическаяСоставляющаяКМ» для концептуальной и «СтатическаяСоставляющаяИЛМ» для инфологической модели.

Класс «ПроизводнаяСтруктура» (рис. 4.7) – описывает производную структуру на схемах статических элементов модели. Является наследником класса «Структура». Реализована в классах «СтатическаяСоставляющаяКМ» для концептуальной и «СтатическаяСоставляющаяИЛМ» для инфологической модели.

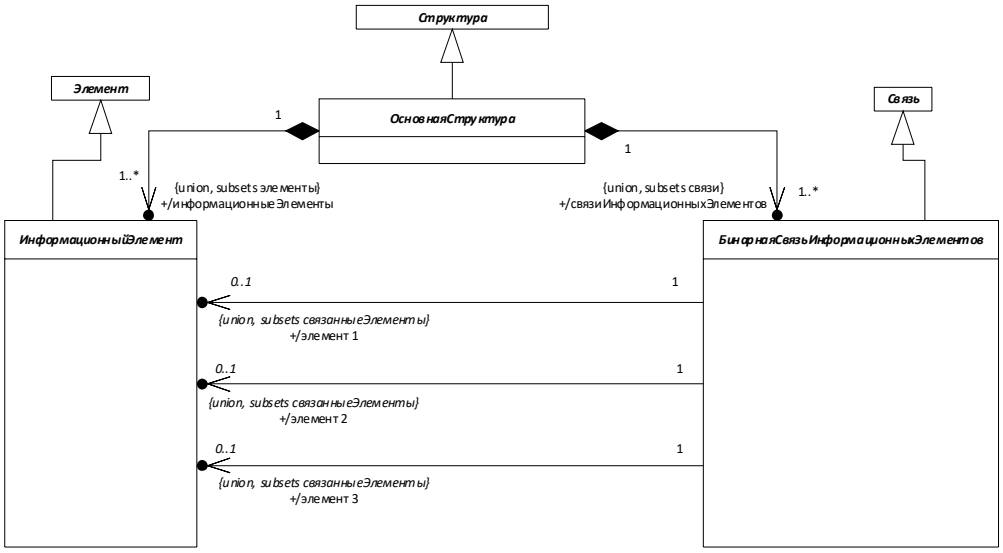


Рис. 4.6 Диаграмма классов. Основная информационная структура

Класс «ИнформационныйЭлемент» – базовый класс для элемента любой основной информационной структуры модели. Является наследником класса «Элемент». Является базовым классом для классов «Параметр» и «Объект» в начальной модели, «ПредметнаяКатегория» в концептуальной и «ИменованнаяСтруктурнаяЕдиница» в инфологической модели.

Класс «БинарнаяСвязьИнформационныхЭлементов» – базовый класс для организации взаимосвязей трех видов (см. описание класса «Связь») между информационными элементами в моделях. Является наследником класса «Связь» и имеет реализации в классах «БинарнаяСвязьПК» для концептуальной модели и «БинарнаяСвязьИСЕ» для инфологической.

Производная структура на схемах (рис 4.7) позволяет описать базовый вариант производной структуры для концептуальной и инфологической модели. В начальной модели подобная структура не предусмотрена. Так как класс является наследником «Структуры», он позволяет описать систему схем как множество связных элементов.

Класс «Схема» – базовый класс для обеспечения реализации производного элемента статической структуры. Является наследником класса «Элемент». Реализован в классах «СхемаПК» для концептуальной модели и «СхемаИСЕ» для инфологической.

Класс «БинарнаяСвязьИСЕ» – базовый класс для реализации взаимосвязей схем. Является наследником класса «СвязьЭлементов» и реализован в классах «БинарнаяСвязьСхемПК» для концептуальной и «БинарнаяСвязьСхемИСЕ» для инфологической.

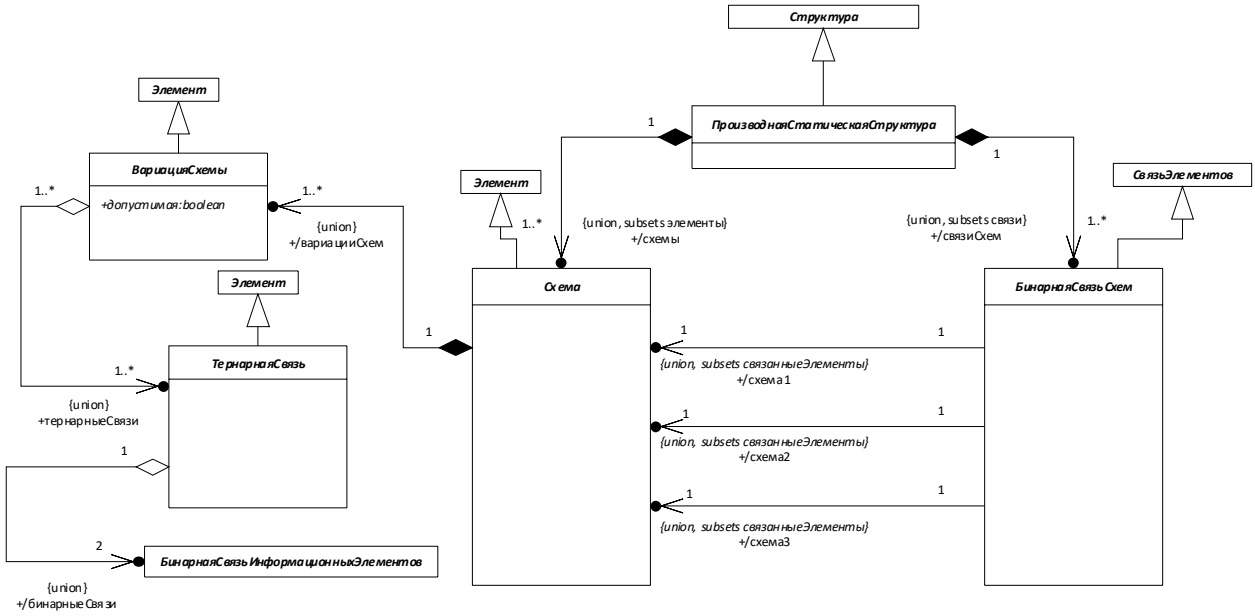


Рис. 4.7 Диаграмма классов. Производная статическая структура

Поведенческая структура (рис. 4.8) описывает абстрактную функциональную и динамическую составляющую моделей МАИТ. Сложная структура содержит две простые – для первого и для второго родов.

Класс «Структура1гоРода» (рис. 4.9) – базовый класс для обеспечения реализации основной функциональной и/или динамической структуры первого рода любой модели, которая отражает процесс решения предметной задачи. Является наследником класса «Структура», поэтому описывает иерархическую графовую структуру. Реализована в классе «СистемаПредметныхДоступов» для начальной модели, в классе

«СистемаПредметныхЗависимостей1гоРода» для концептуальной модели и классах «СистемаПредметныхМанипуляций1гоРода», «СистемаПредметныхДоступов1гоРода» для инфологической модели.

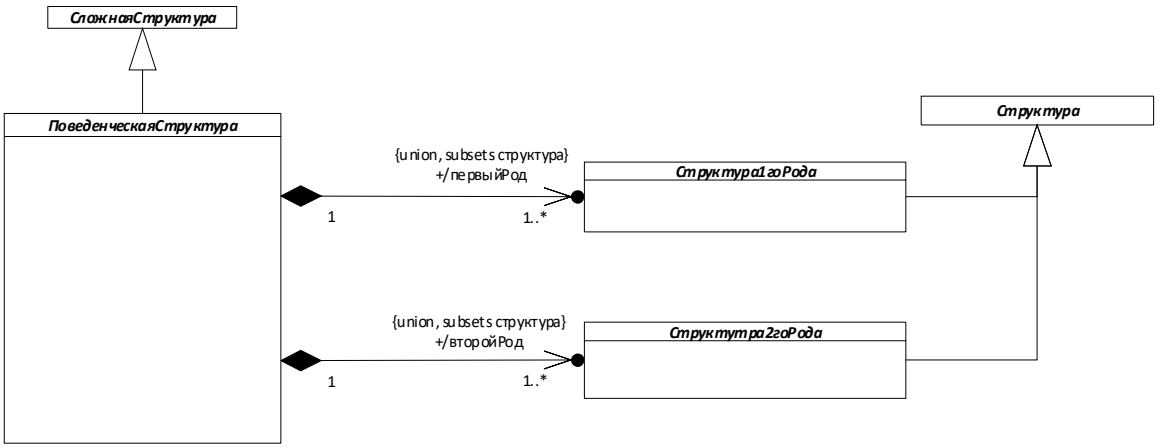


Рис. 4.8 Диаграмма классов. Составляющие поведенческой структуры

Класс «Структура2гоРода» (рис. 4.10) – базовый класс для реализации основной функциональной и/или динамической структуры второго рода модели. Является наследником класса «Структура», поэтому формируется из наследников «Элемент» и «БинарнаяСвязьЭлементов». Реализован в классе «СистемаПредметныхЗависимостей2гоРода» для концептуальной модели и классах «СистемаПредметныхМанипуляций2гоРода», «СистемаПредметныхДоступов2гоРода» для инфологической модели. В начальной модели структура второго рода не предусмотрена.

Класс «Операция» – базовый функциональный/динамический элемент любой модели. Является наследником класса «Элемент». Реализован в классах «ПредметноеДействие» в рамках начальной модели, «ПерметнаяЗависимость1гоРода» в рамках концептуальной, и в классах «ПредметнаяМанипуляция1гоРода» и «ПредметныйДоступ1гоРода» в рамках инфологической.

Класс «БинарнаяСвязьОпераций» – базовый класс для организации взаимосвязей в функциональный/динамический структуре первого рода любой модели. Является наследником класса «Связь». Реализован в классах «БинарнаяСвязьПД» в рамках начальной модели, «БинарнаяСвязьПЗ1» в рамках концептуальной, и в классах «БинарнаяСвязьПМ1» и «БинарнаяСвязьПд1» в рамках инфологической.

Класс «Ограничение» – базовый функциональный/динамический элемент структуры второго рода модели. Является наследником класса «Элемент». Реализован в классе «ПредметнаяЗависимость2гоРода» в рамках концептуальной модели и в классах «ПредметнаяМанипуляция2гоРода» и «ПредметныйДоступ2гоРода» в рамках инфологической

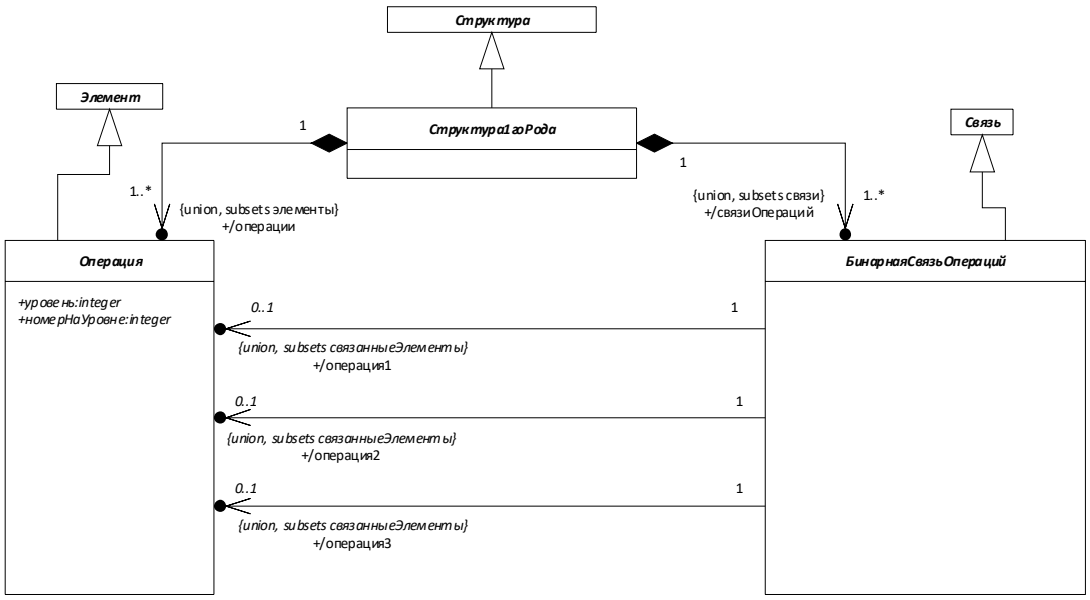


Рис. 4.9 Диаграмма классов. Поведенческая структура первого рода

.

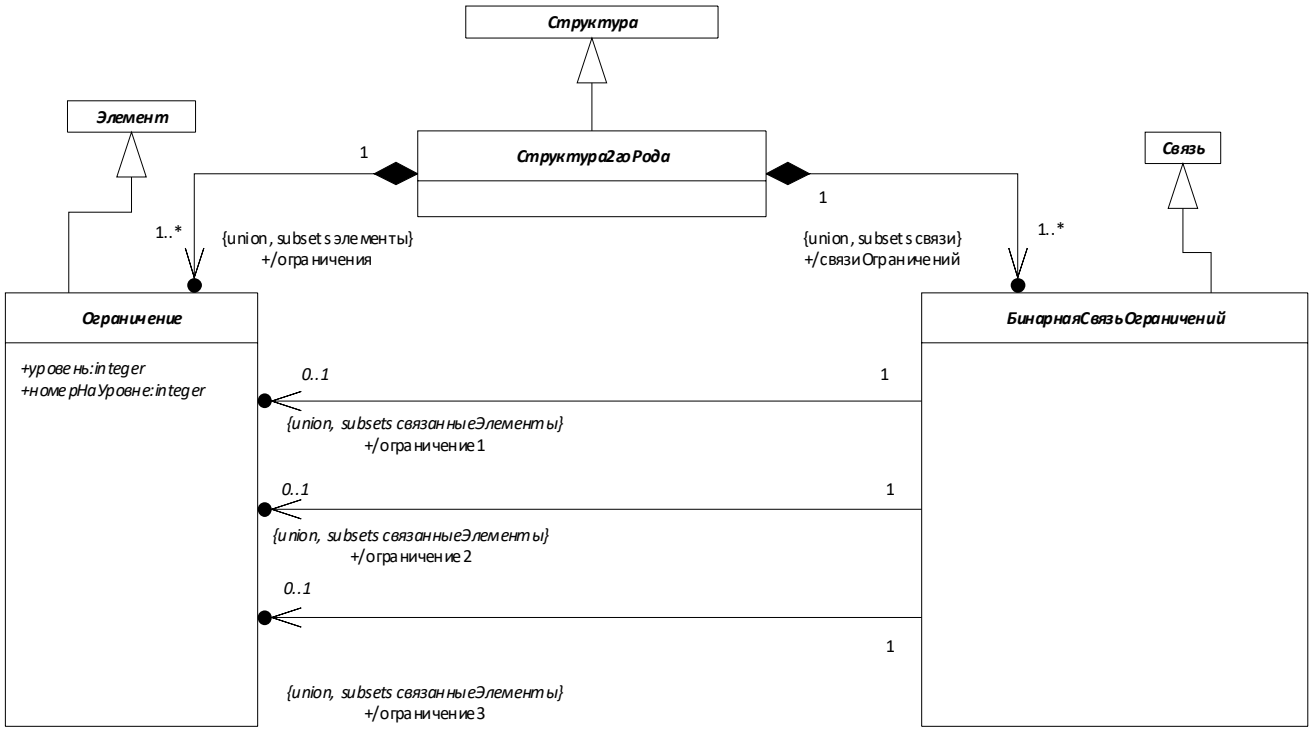


Рис. 4.10 Диаграмма классов. Поведенческая структура второго рода

Класс «БинарнаяСвязьОграничений» – базовый класс для организации взаимосвязей в функциональный/динамический структуре второго рода модели. Является наследником класса «Связь». Реализован в классах «БинарнаяСвязьПЗ2» в рамках концептуальной и в классах «БинарнаяСвязьПМ2» и «БинарнаяСвязьПд2» в рамках инфологической модели.

Класс «МодельВЦелом» (рис 4.11) – сложная структура для множеств содержательных структур в рамках общей модели. Унаследован от класса «СтруктурыУвязки» и реализован в

классах «НачальнаяМодельВЦелом», «КонцептуальнаяМодельВЦелои» и «ИнологическаяМодельВЦелом».

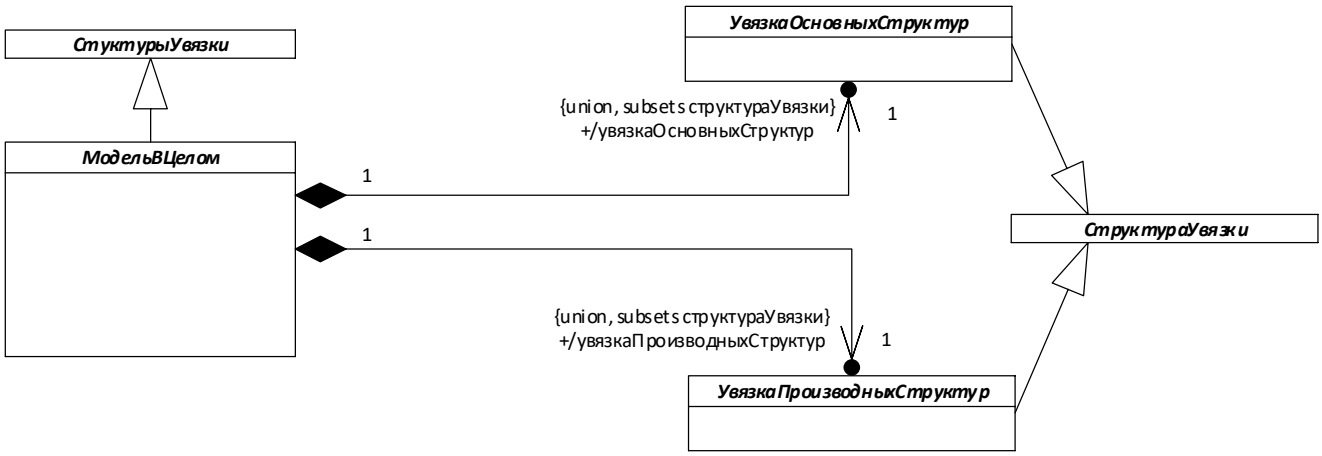


Рис. 4.11 Диаграмма классов. Составляющие модели в целом

Элементами структур увязки (рис. 4.12, 4.13) являются содержательные элементы – взаимосвязи между элементами информационных и функциональных/динамических структур.

Класс «СодержаниеОперации» – базовый класс для организации взаимосвязи между информационными элементами и операциями. Унаследован от класса «СодержаниеОперации». Является родителем для классов «СодержаниеПредметногоДействия» в начальной модели, «СодержаниеПредметнойЗависимости1гоРода» в концептуальной модели, классов «СодержаниеПредметнойМанипуляции1гоРода» и «СодержаниеПредметногодотсупа1гоРода» для инфологической модели.

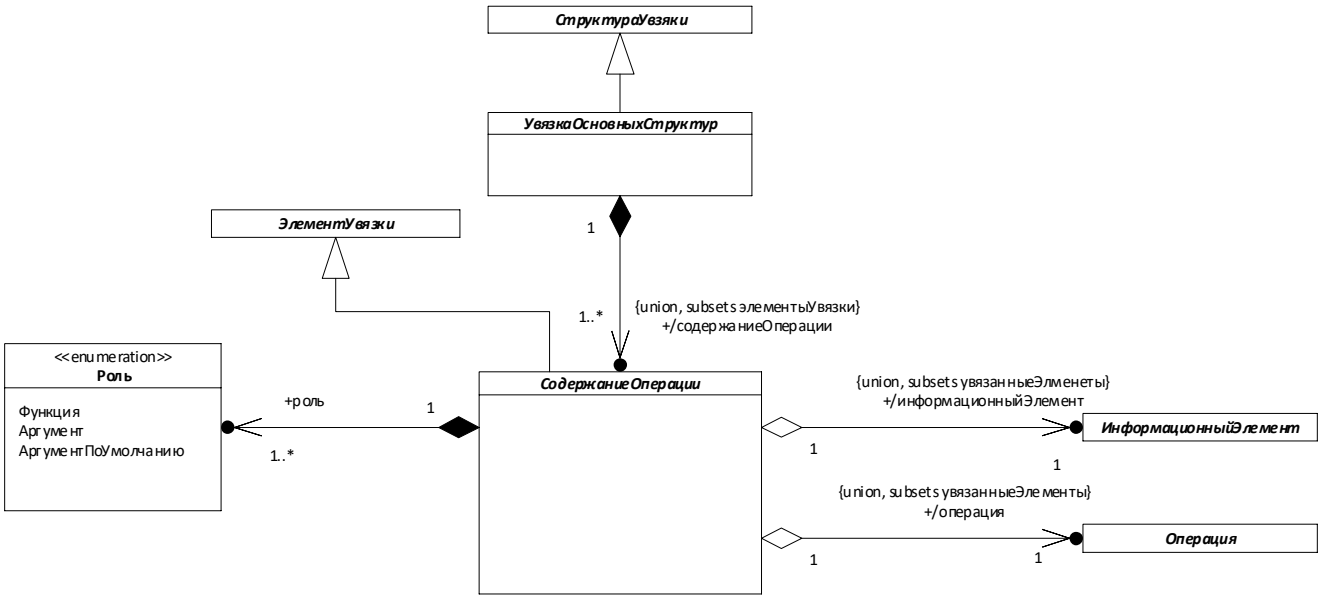


Рис. 4.12 Диаграмма классов. Увязка основных структур

Класс «УвязкаОграничения» – базовый класс для организации взаимосвязи между информационными элементами и ограничениями. Унаследован от класса

«СодержаниеОграничения». Является родителем для класса «СодержаниеПредметнойЗависимости2гоРода» в концептуальной модели, классов «СодержаниеПредметнойМанипуляции2гоРода» и «СодержаниеПредметногодотсупа2гоРода» для инфологической модели.

Полный перечень диаграмм классов, описывающих структуры данных моделей МАИТ приведен в приложении Г.

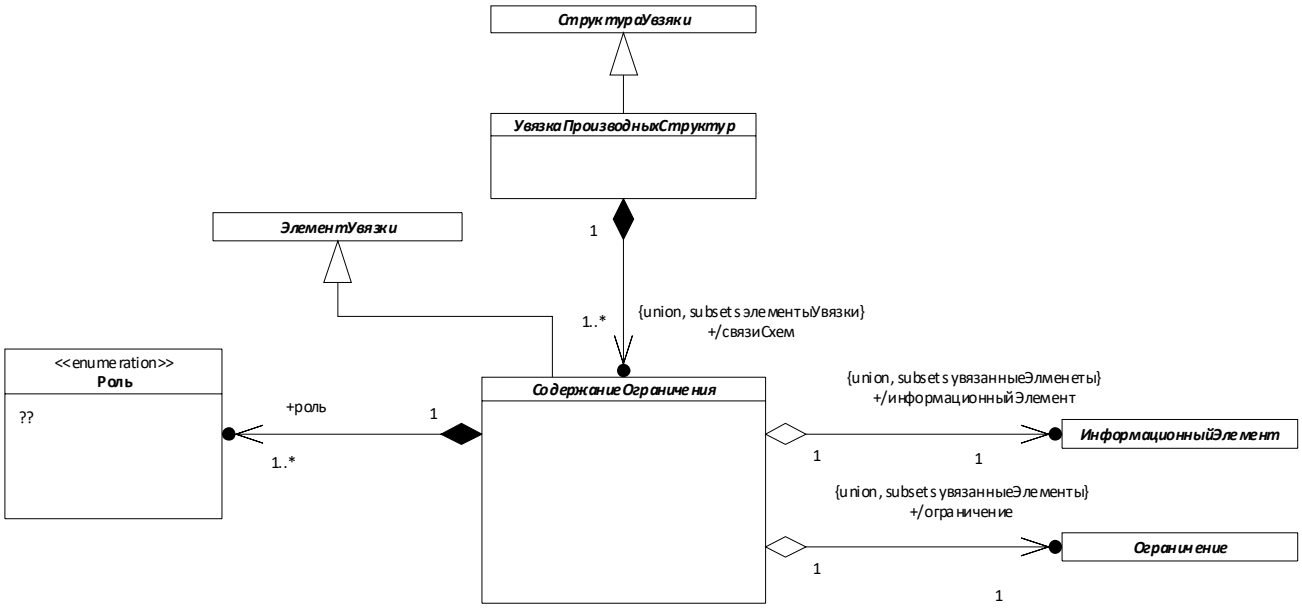


Рис. 4.13 Диаграмма классов. Увязка производных структур

4.4 Описание реализации функциональных модулей ВИС

На данный момент реализованы конфигурации ПК «ИС-2. Начальное моделирование», «ИС-2. Концепт» и прототип «ИС-2. Инфология». В параграфе приведено описание ключевых функциональных элементов разработанных конфигураций. Схожие функциональные модули, применяемые на различных этапах создания АС, не рассматриваются.

4.4.1 Создание проекта

Конфигурация программного комплекса ***«ИС-2. Начальное моделирование»*** предназначена для формирования начальной модели в визуальной форме.

Компоненты модели предметной задачи:

- **функциональная составляющая** (система предметных действий) – отражает алгоритм действий, выполняемых в рамках предметной задачи;
- **информационная составляющая** – множество информационных параметров, используемых при решении задачи;
- **увязка функциональной и информационной составляющих.**

Функциональная составляющая формируется за счет добавления предметных действий и установления на них бинарных связей. Бинарные связи ПД отражают логику процесса решения задачи в виде иерархии уровней декомпозиции и последовательности (упорядоченности) выполнения действий на каждом уровне декомпозиции.

Программа позволяет упростить и автоматизировать процесс создания декомпозиции предметных задач, их табличное представление в виде спецификаций и матричной диаграммы.

При запуске программы открывается ***главное окно*** (рис. 4.14). Оно позволяет создать проект и указать его характеристики. ***Главное окно*** оснащено *основными функциями* для работы с файлом проекта:

- **Новый проект** – создание файла для нового проекта; при этом будет создана пустая версия проекта, которая не содержит элементов информационной составляющей, но содержит одно единственное корневое предметное действие; функциональная структура формируемой модели будет «расти» из этого элемента.
- **Открыть** – открытие уже сохранённого проекта, при этом если в данный момент существует открытый проект, то во *всплывающем окне* нужно подтвердить/отклонить сохранение изменений в текущем проекте;
- **Сохранить** – сохранение текущего проекта;
- **Сохранить как** – сохранение текущего проекта под другим названием, либо в другой папке;
- **Список последних проектов** – открытие одного из последних созданных проектов, при этом, если в данный момент существует открытый проект, то во *всплывающем окне* нужно подтвердить/отклонить сохранение изменений в текущем проекте;
- **Информация о проекте** – внесение изменений об авторе проекта, его статусе, названии и описании.

Для того чтобы начать работу с созданным проектом, следует нажать на кнопку «**К работе**».

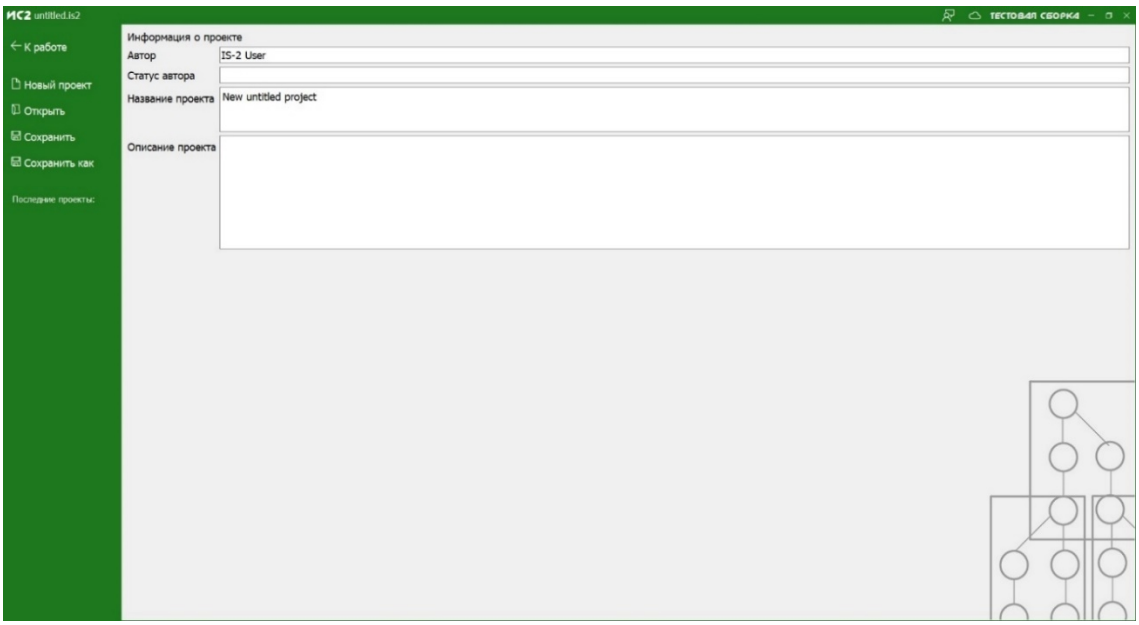


Рис. 4.14 Главное окно программы «ИС2. Начальное моделирование»

4.4.2 Описание редактора системы предметных действий

Редактор СПД (рис 4.15) предназначен для работы с проектом: построение декомпозиции, спецификаций и матричной диаграммы. С помощью кнопки  осуществляется возврат к ***главному окну*** программы.

Панель инструментов обеспечивает быстрый доступ ко всем необходимым функциям при формировании СПД (рис. 4.16). Панель разделена на несколько составляющих:

Навигация, включает в себя функции **«Назад»** и **«Вперёд»** – переход к предыдущему либо возврат к последнему открытому листу с частью декомпозиции;

Создание типовых конструкций – группа кнопок обеспечивает преобразование элементарных блоков в базовые конструкции (последовательность, альтернатива, итерация). Совокупность таких операций позволяет сформировать типовые конструкции (рис. 4.17):

- **последовательность –** определяет последовательность выполнения простых предметных действий на ***p+1*** уровне в рамках сложного предметного действия на уровне ***p***;
- **альтернатива –** определяет выполнение предметного действия на ***p+1*** уровне в соответствии с условием, определённым на уровне ***p***;
- **итерация** (в программе именуется **циклом**) **–** определяет сложное предметное действие на уровне ***p+1***, которое будет выполняться заданное количество раз для задачи, определённой на уровне ***p***;

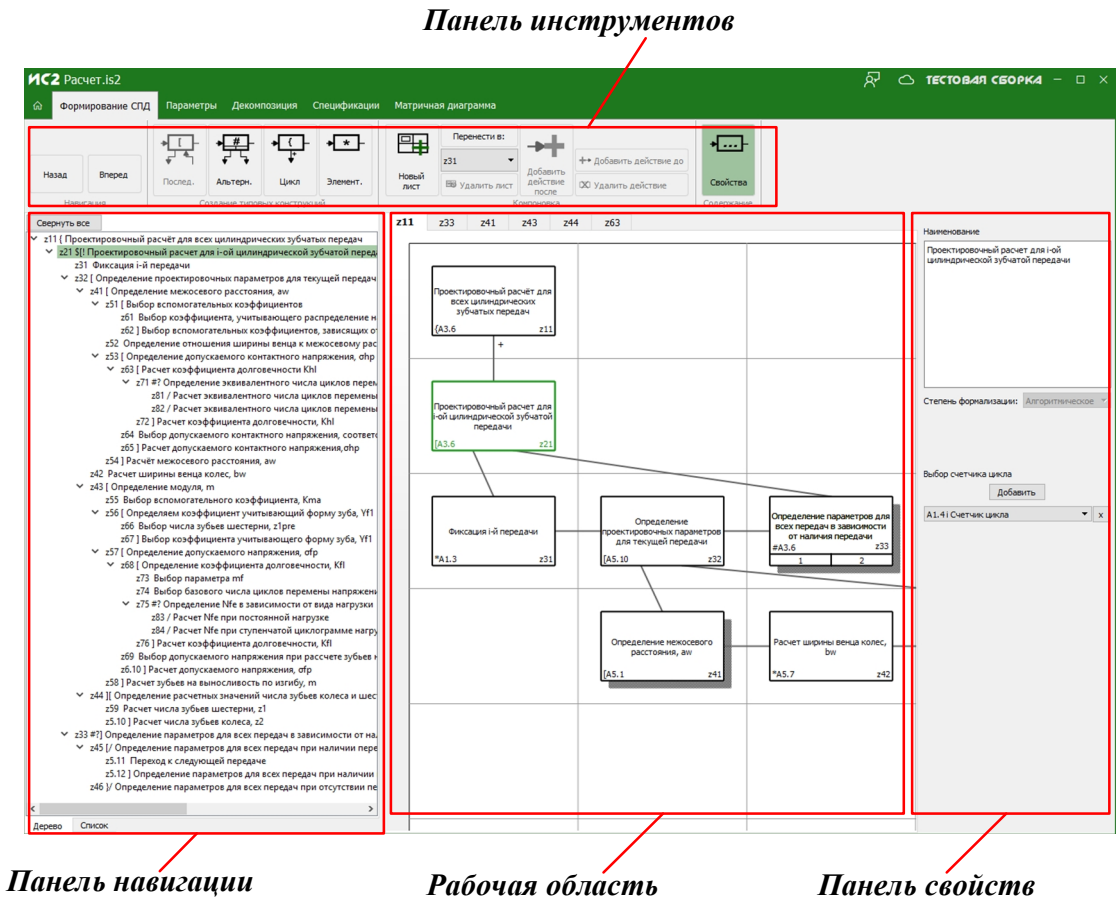


Рис. 4.15 Интерфейс редактора СПД

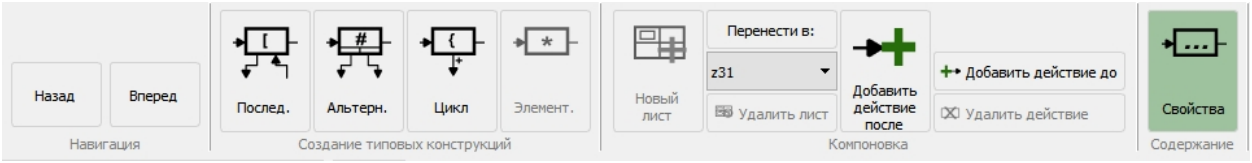


Рис. 4.16 Панель инструментов

- **элементарное ПД** – удаляет все дочерние блоки промежуточного ПД, преобразуя его в элементарное.

Компоновка – группа функций, отвечающих за расположение действия в декомпозиции; состоит из:

– **новый лист** – перенос конструкции на созданный пустой лист при отсутствии места на существующем листе;

– **перенести в –** перенос выделенного ПД на место элементарного ПД, выбранного в выпадающем ниже меню;

– **удалить лист** – перенос конструкции на существующий лист при наличии на нём места с соответствующим удалением листа, на котором она находилась;

– **добавить действие до** – создание действия до выделенного ПД;

– **добавить действие после** – создание действия после выделенного ПД;



Рис. 4.17 Визуальное отображение базовых конструкций

– **удалить действие** – удаление выделенного действия.

Содержание, включает в себя функцию **«Свойства»** – позволяет указать необходимые свойства всех действий декомпозиции (наименование, степень формализации и т.п.).

Рабочая область – разбитая на уровни большая площадь, предназначенная для расположения предметных действий и формирования их в СПД. Возможность перемещения по рабочей области осуществляется при непрерывно нажатой левой кнопки мыши, а масштабирование – при прокрутке колеса мыши.

Панель свойств (рис. 4.18) позволяет изменять свойства как самого предметного действия в зависимости, так и его взаимосвязи с элементами статической составляющей (с параметрами или с предметными категориями). Данная панель имеет следующие общие для всех элементов поля:

-наименование – наименование блока, отображается в его границах на листе графического редактора;

-степень формализации – позволяет задать/изменить степень формализации элемента, закрепленного за блоком диаграммы. Для промежуточных действий выставляется автоматически, в зависимости от вида базовой конструкции, в которую входит элемент.

Для элементарных блоков в панели свойств можно задавать взаимосвязи с элементами статической составляющей модели:

-функция – выпадающий список с параметрами/предметными категориями, позволяющий выбрать функцию действия/предметной зависимости;

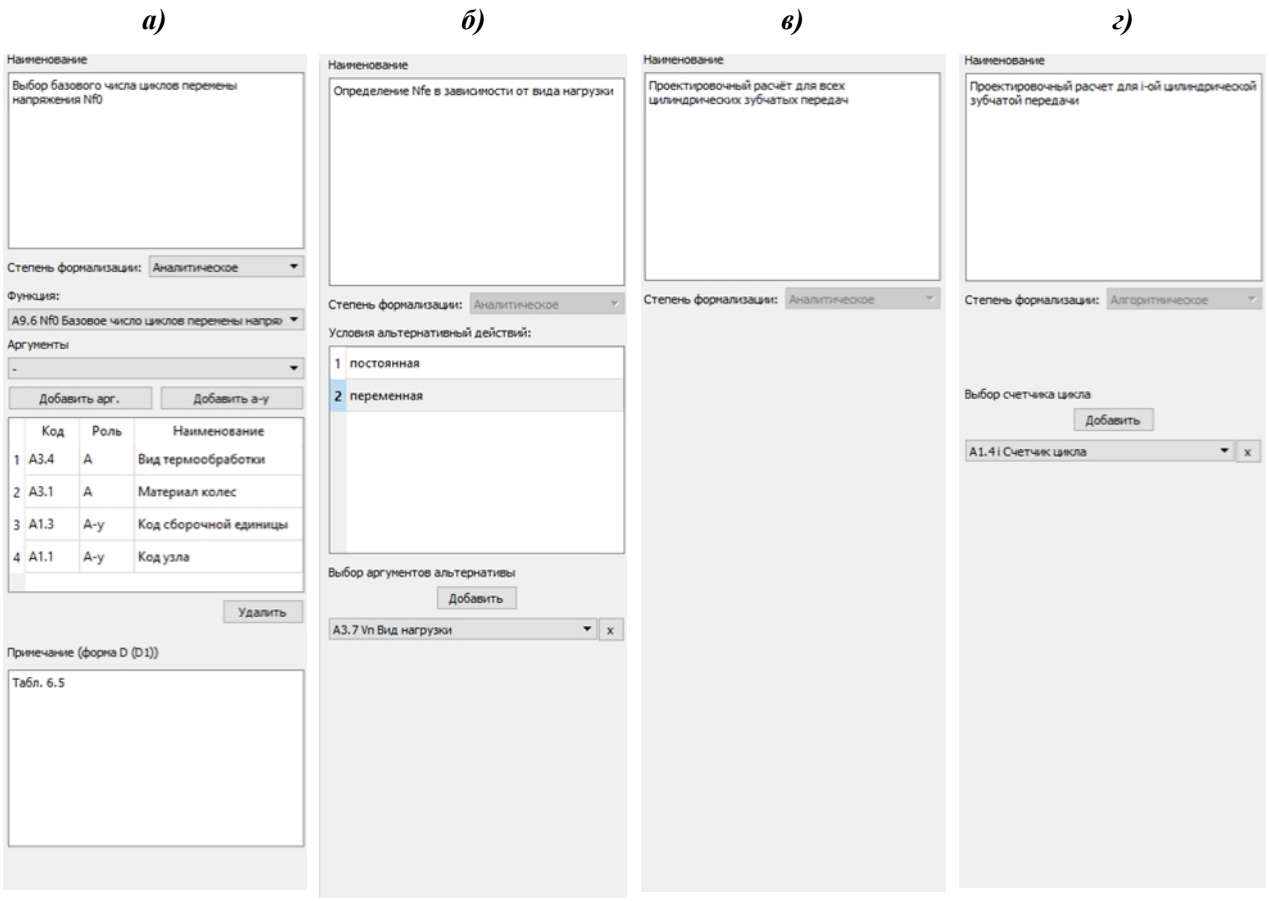


Рис. 4.18 Панели свойств блоков предметных действий различных типов: а) для элементарного ПД, б) для заголовка альтернативы, в) для заголовка итерации, г) для тела итерации (являющегося в примере заголовком последовательности).

-аргументы – совокупность элементов, позволяющий сформировать для выбранного действия/зависимости список аргументов или аргументов по умолчанию.

-примечание – позволяет фиксировать примечание к выделенному элементарному предметному действию (только для начального моделирования).

Для заголовков альтернативы определены следующие поля:

-условия альтернативных действий – позволяет задать текст условий альтернативных действий; количество условий определяется количеством альтернативных действий в подструктуре выделенного заголовка;

-аргументы альтернативы – дает возможность определить, какие параметры/предметные категории являются аргументами выделенной альтернативы;

Для тела итерации в панели свойств можно задать **счетчик** аналогично выбору аргументов альтернативы.

Дерево (список) модели – инструмент, который используется для просмотра структуры модели в виде дерева или списка. При двойном нажатии на то или иное действие оно откроется в декомпозиции (рис. 4.19).

а) *б)*

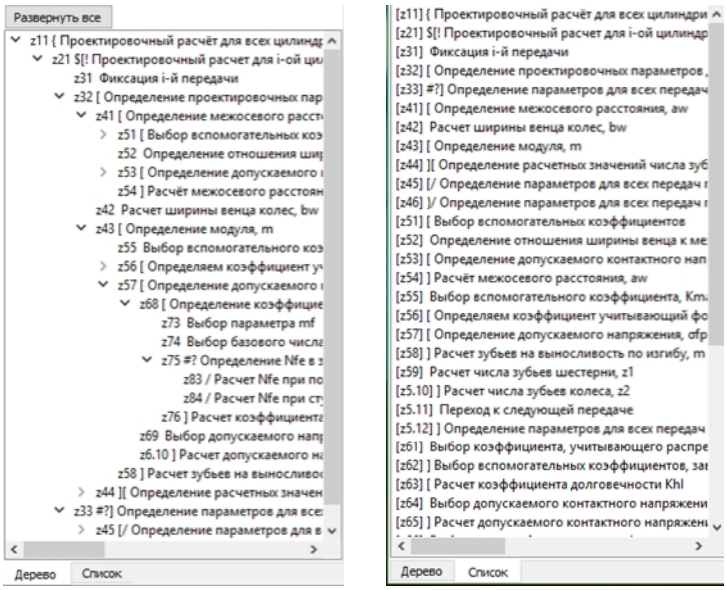


Рис. 4.19 Дерево (а) и список модели (б)

После перехода к работе с проектом на *рабочей области* автоматически создаётся первое ПД. Для того чтобы создать последующие ПД, следует выбрать один из вариантов типовых конструкций, предварительно выделив родительское ПД. Каждое ПД автоматически имеет свою кодировку **Zx.y**, где **x** – номер уровня декомпозиции, а **y –** номер элемента на уровне.

При построении декомпозиции применяются один из двух типов конструкций – базовые и типовые. Типовые конструкции формируются на основе базовых: **«последовательность»**, **«итерация»** и **«альтернатива»**. Они описывают фрагмент алгоритма только на двух смежных уровнях сложности: ***p*** и ***p+1***.

Функции «*Добавить действие после*» и «*Добавить действие до*» на панели «*Компоновка*» создают действие **после или до выбранного предметного действия** (рис. 4.20).

Функция «*Удалить действие*» выполняет роль **удаления ненужного предметного действия**. Действие должно быть выделено.

Так как разложение сложного действия может варьироваться от 2 до 6 объектов (в соответствии с правилами декомпозиции), в программе предусмотрено ограничение, которое не позволяет создать на одном уровне больше 6 предметных действий. Поэтому *при формировании СПД может возникнуть ситуация, при которой не останется места для разложения какого-либо действия.*

Решить данную проблему можно добавлением предметных действий до тех пор, пока на уровне не останется свободного места и тогда программа сама предложит декомпозировать действие на другой лист, либо воспользоваться функцией «*Новый лист*» на панели «*Компоновка*», которая **перенесёт декомпозируемое действие на новый лист** с созданием соответствующей вкладки с кодировкой (рис. 4.21).

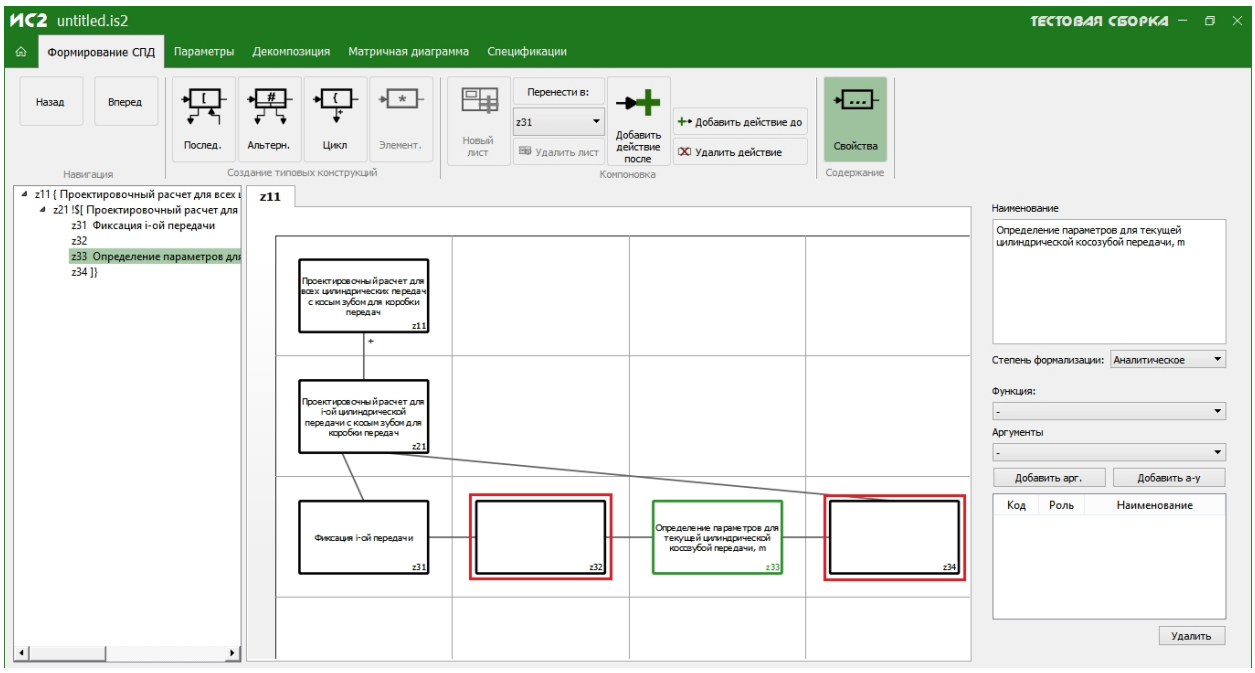


Рис. 4.20 Добавление предметных действий до и после выбранного блока

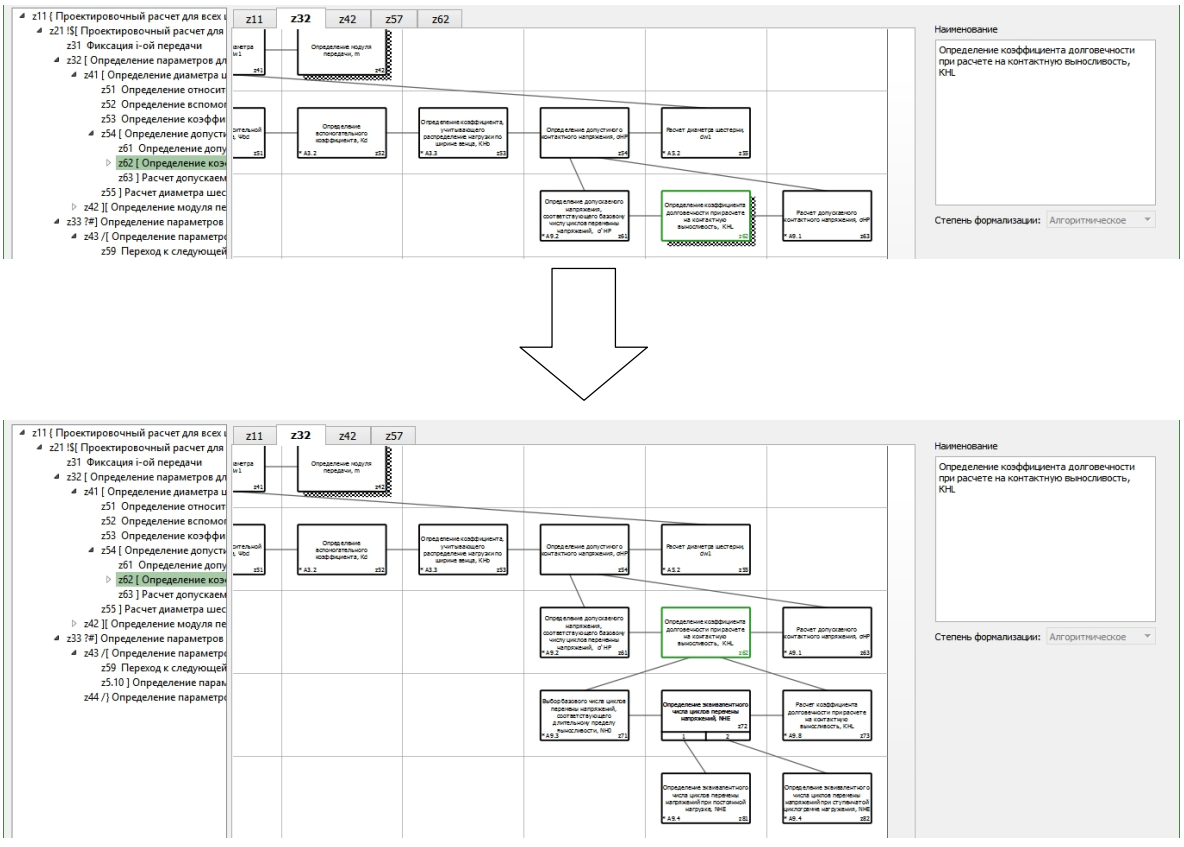


Рис. 4.21 Результат выполнения функции «Удалить лист»

Перенос предметных действий в программе осуществляется выбором блока левой кнопкой мышкой и **переносом** (не отпуская кнопку) **его между любыми блоками в рамках конструкции**, в которой он находится (рис. 4.22).

Функция «*Перенести* в» на панели «*Компоновка*» **переносит** выделенное **предметное действие и всю его подструктуру на место любого элементарного действия** (рис. 4.23, 4.24).

Поскольку после переноса одного из действий в альтернативе Z33 осталось одно действие, то оно было перенесено на уровень выше вместо альтернативы.

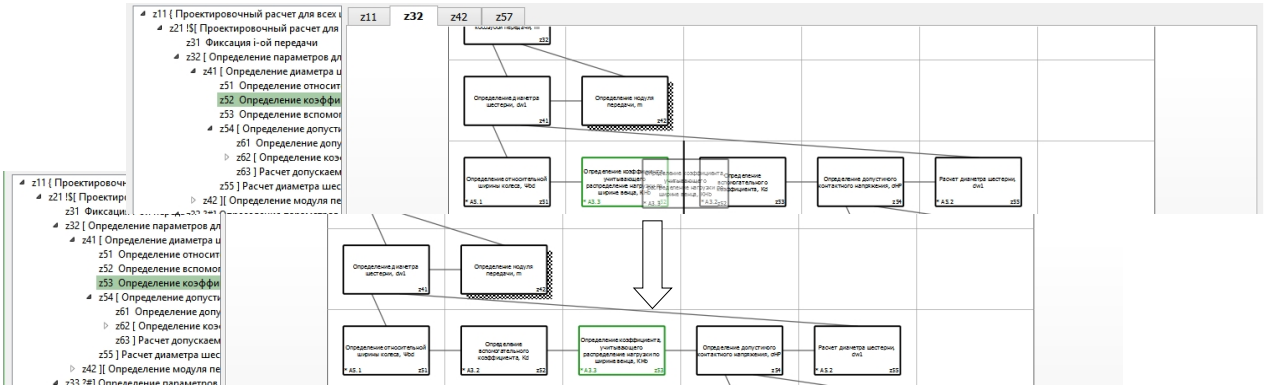


Рис. 4.22 Перенос предметного действия в рамках конструкции, в которой оно находится

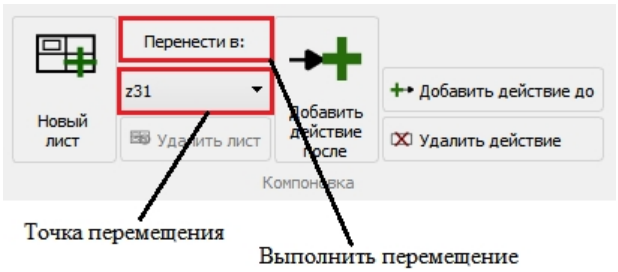


Рис. 4.23 Функция «Перенести в» и точка перемещения выбранного предметного действия

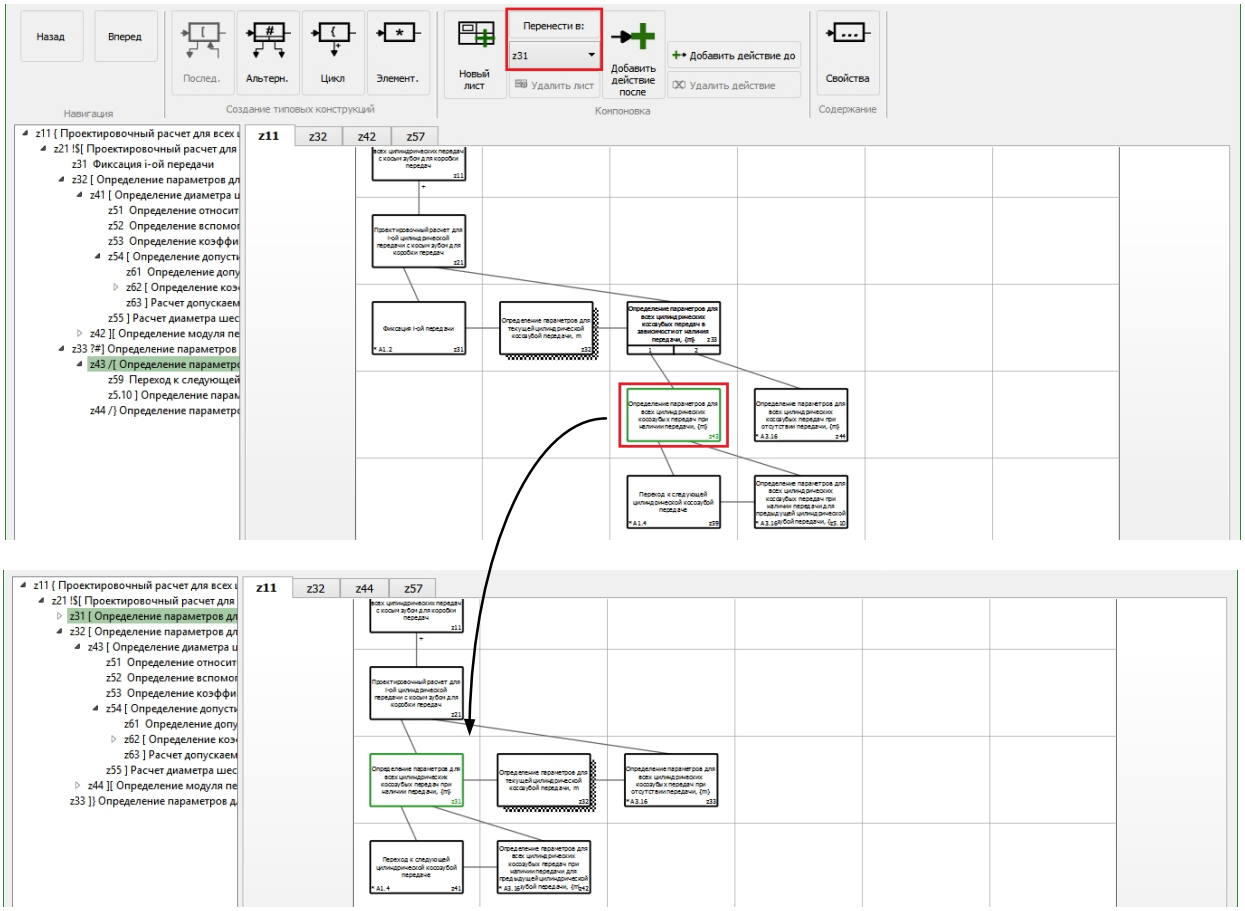


Рис. 4.24 Перенос предметного действия и его подструктуры на место любого элементарного действия

Базовая алгоритмическая конструкция «последовательность» определяет последовательное выполнения предметных действий на ***p+1*** уровне в рамках сложного предметного действия на уровне ***p***.

Базовая алгоритмическая конструкция «итерация» (в программе указана как «цикл») определяет сложное предметное действие на уровне ***p+1***, которое будет выполняться заданное количество раз для задачи, определённой на уровне ***p***.

Базовая алгоритмическая конструкция «итерация» (в программе указана как «цикл») определяет сложное предметное действие на уровне ***p+1***, которое будет выполняться заданное количество раз для задачи, определённой на уровне ***p***.

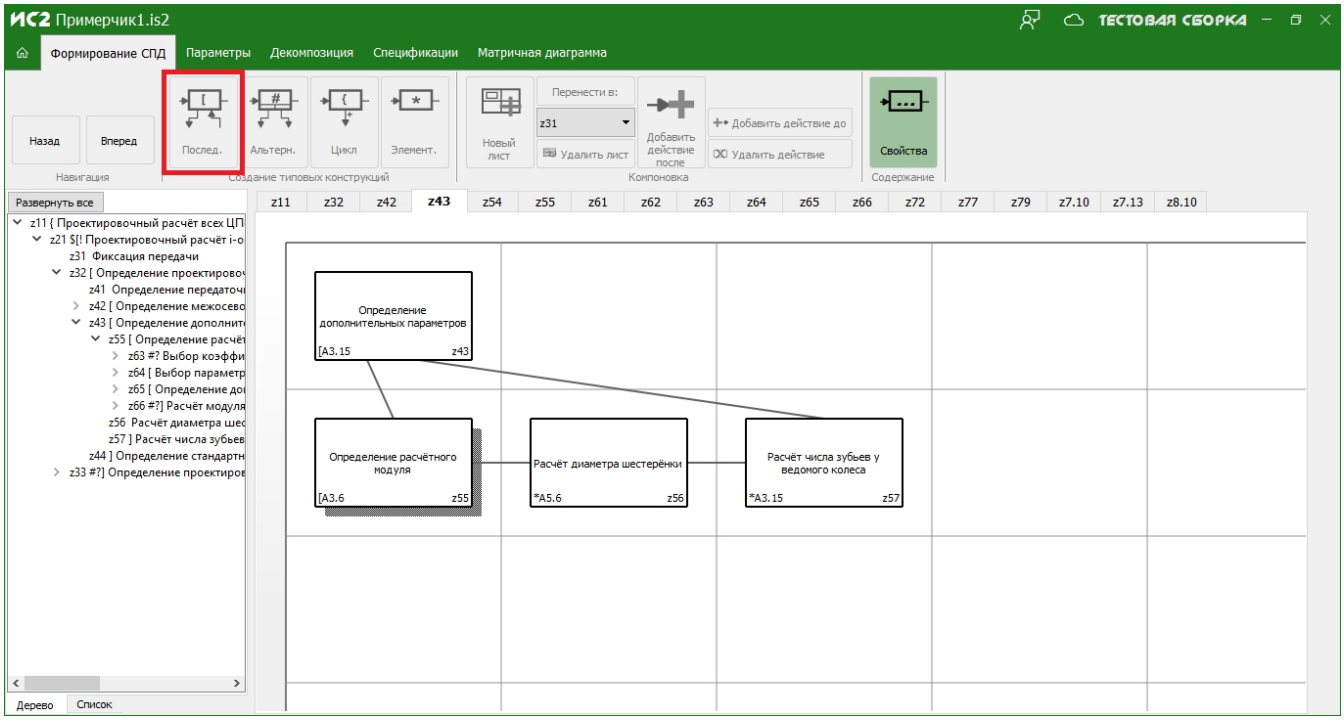


Рис. 4.25 Кнопка «Последовательность» и созданная последовательность в рабочей области

Другой вид конструкций, используемый при построении декомпозиции – типовые: **«переключатель»** и **«цикл»**. Они образуются на основе базовых.

Типовая конструкция «переключатель» формируется на основе базовой конструкции «альтернатива» и «последовательность» и создается путём раскрытия альтернативных действий в виде заголовков последовательностей.

Типовая конструкция «цикл» формируется на основе базовых конструкций «итерация», «последовательность» и «альтернатива». Создается путём представления тела итерации в виде последовательности, состоящей из трех действий. Первое действие фиксирует аргумент итерации, второе – определяет функцию итерации, третье действие раскрывается в виде альтернативы и определяет условие завершения или продолжения цикла (рис. 4.26).

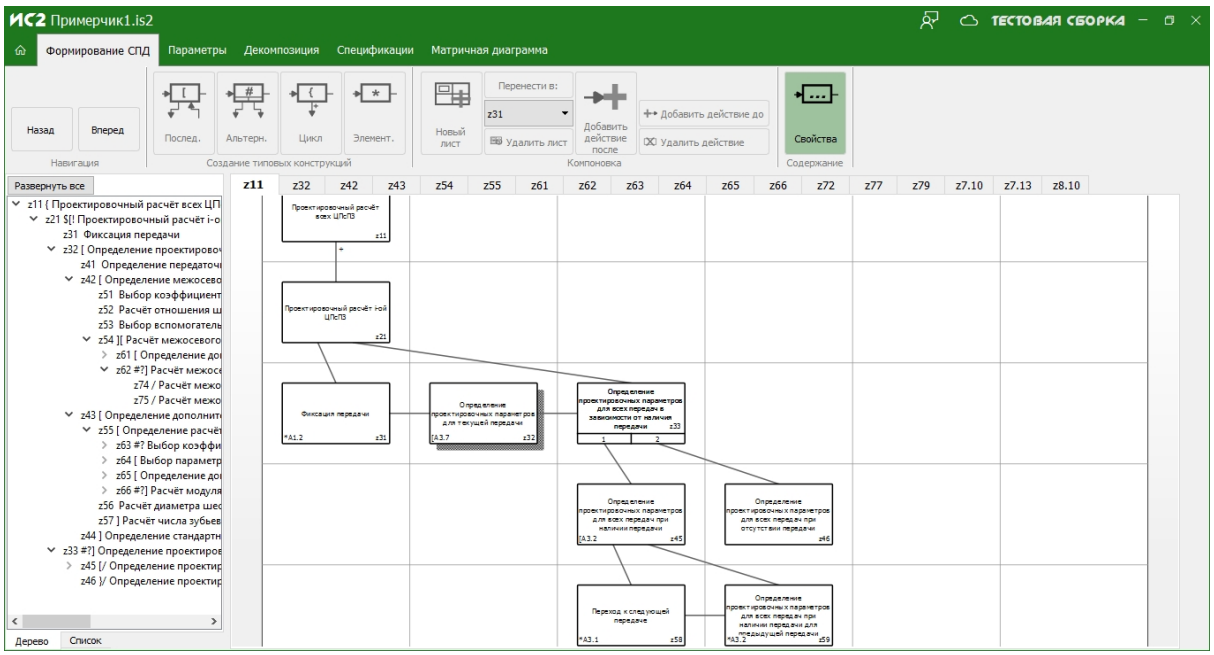


Рис. 4.26 Созданная конструкция «Цикл» в рабочей области

4.4.3 Формирование списка параметров

Параметр – элемент начальной модели объектного уровня, описывающий именованное свойство объекта предметной области в общем виде.

Во вкладке «Параметры» предоставляется возможность присвоить каждому ПД следующие параметры: принадлежность к какой-либо группе и объекту, номер параметра, его обозначение и название.

Кнопка «*Добавить*» **создаёт список параметров** (рис. 4.27).

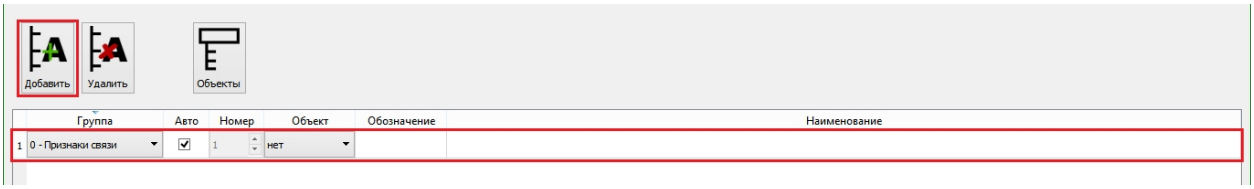


Рис. 4.27 Окно вкладки «Параметры»

После нажатия кнопки «*Добавить*» появляется строка, в которой указывается:

- принадлежность параметра к той или иной группе. Группа выберется из списка, показанного на рисунке 4.28;
- номер параметра в этой группе; поставленная галочка в окошке позволяет автоматически присвоить номер для параметра, снятая – выставляется вручную
- принадлежность к объекту – узел, сборочная единица или деталь;
- обозначение параметра и его название.

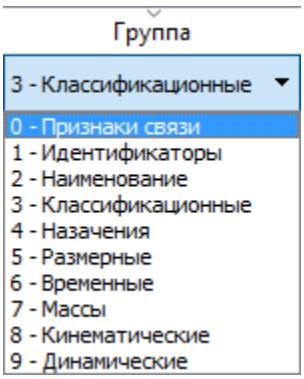


Рис. 4.28 Выпадающее меню со списком групп параметров

Кнопка «*Удалить*» **удаляет параметры**, при этом объект должен быть активен.

Кнопка «*Объекты*» используется **для добавления объектов**. После её нажатия в открывшемся окне можно указать необходимые для проекта объекты (рис. 4.29).

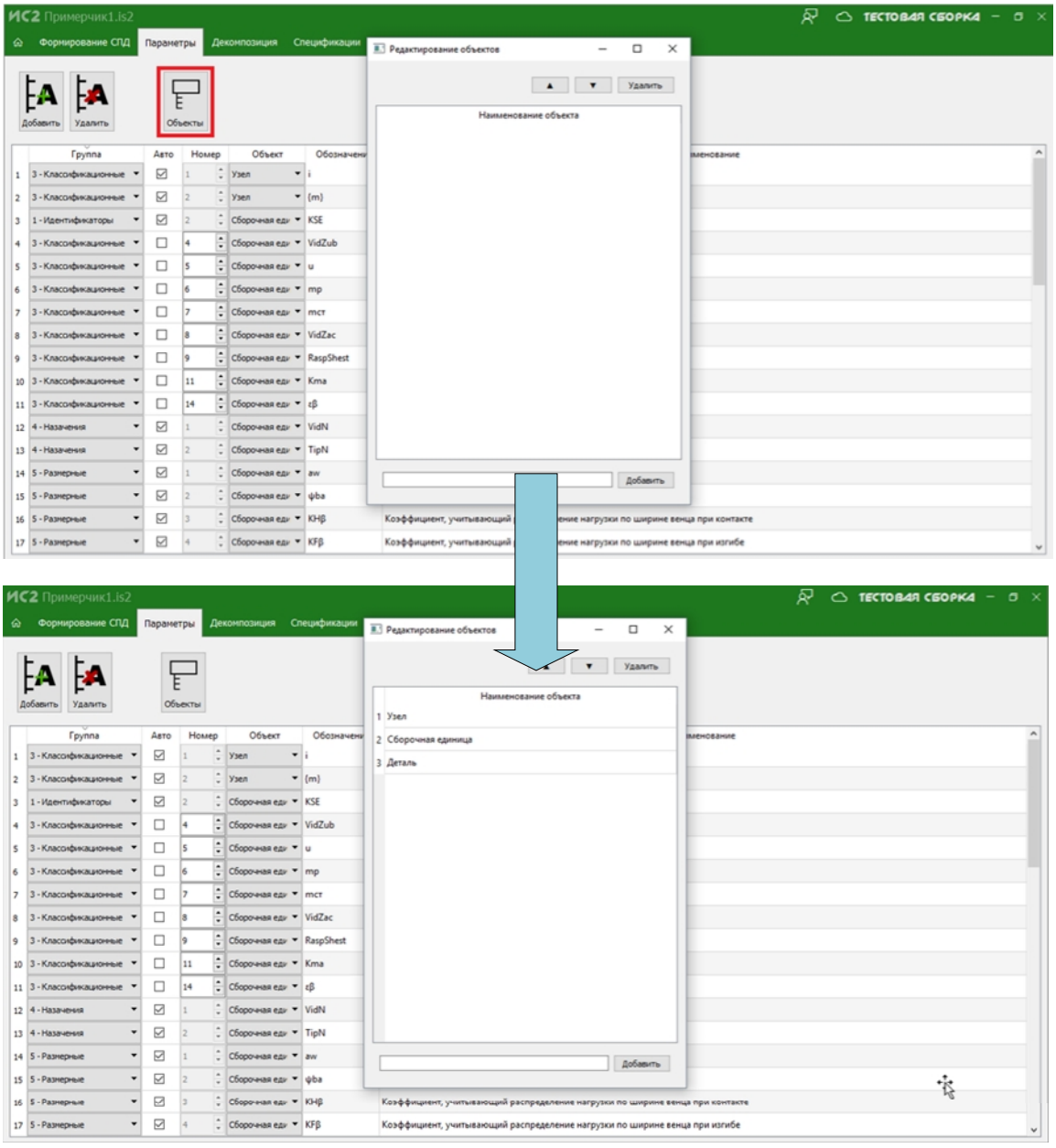


Рис. 4.29 Добавление объектов

После добавления объектов в выпадающем списке можно выбрать принадлежность того или иного параметра к объекту (рис. 4.30).

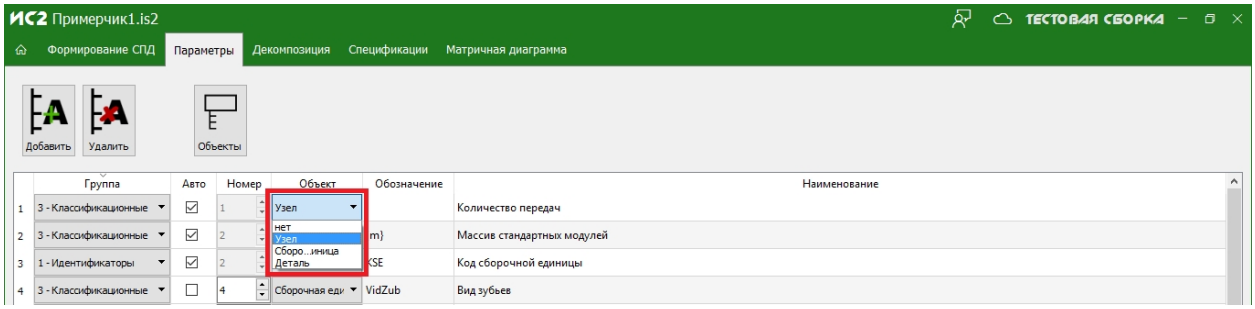


Рис. 4.30 Выбор принадлежности к объекту

4.4.4 Формирование взаимосвязей параметров и предметных действий

Функция «*Свойства*» на панели «*Содержание*» **добавляет параметры к конкретному предметному действию**. Чтобы добавить параметры, предметное действие должно быть активно.

Когда кнопка «Свойства» активна, справа от рабочей области появляются свойства выбранного предметного действия (рис. 4.31).

Для элементарных ПД можно назначить следующие свойства (рис. 4.32 и 4.33):

- функция предметного действия (выбирается в выпадающем списке);
- аргументы предметного действия (также выбираются из выпадающего списка, а затем нажимается кнопка «Добавить аргумент», либо «Добавить аргумент по умолчанию», в зависимости от роли параметра в предметном действии).

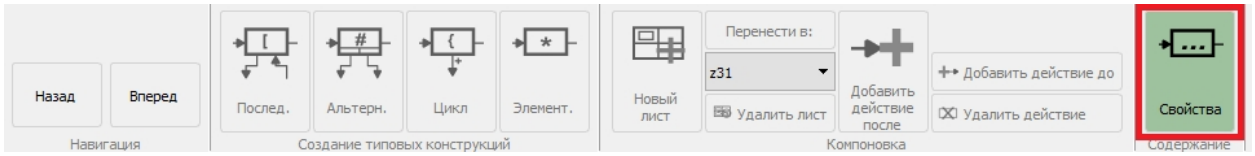


Рис. 4.31 Кнопка «Свойства»

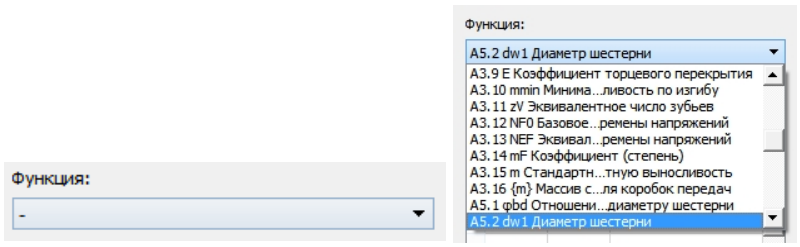


Рис. 4.32 Выпадающее меню «Функция» и список функций

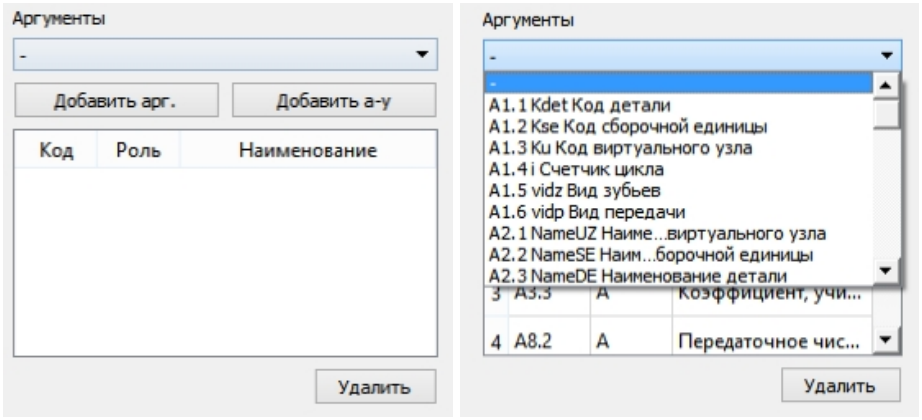


Рис. 4.33 Выпадающее меню с окном «Аргументы» и список аргументов

Для заголовков альтернативы можно назначить свойство (рис. 4.34) аргумента альтернативы (выбирается в выпадающем списке).

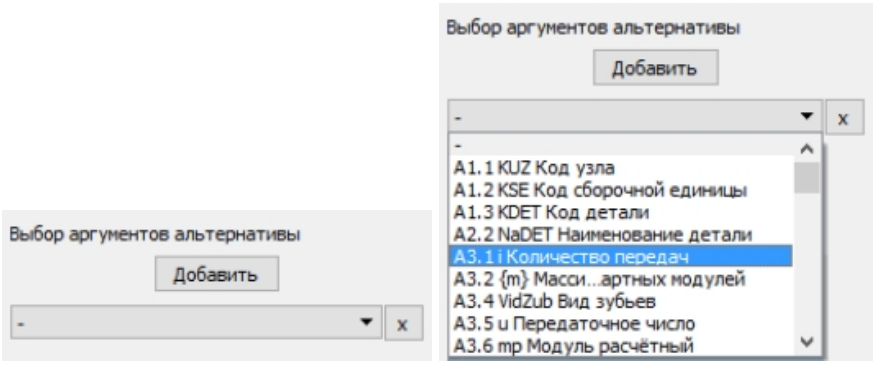


Рис. 4.34 Выпадающее меню «Аргумент альтернативы» и список аргументов

Для тела итерации (блок с обозначением !$) можно назначить свойство (рис. 4.35) счётчика цикла (выбирается в выпадающем списке).

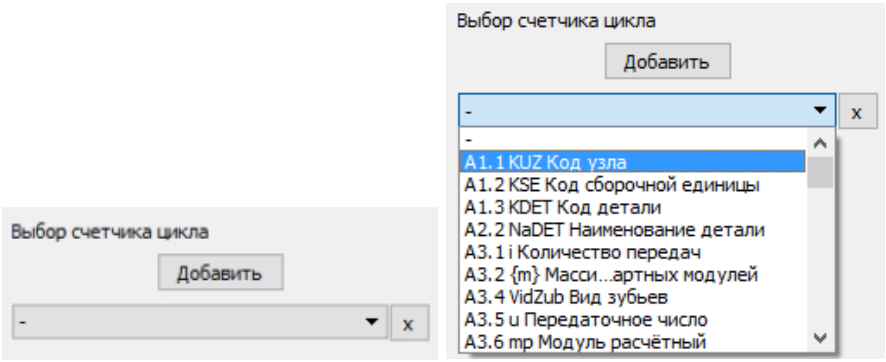


Рис. 4.35 Выпадающее меню «Счётчик цикла» и список счетчиков

4.4.5 Формирование основной концептуальной структуры

Основная концептуальная структура (ОКС) – это схема, формирующая **предметные**

категории (ПК) и связи между ними.

Панель инструментов обеспечивает быстрый доступ ко всем необходимым функциям при формировании ОКС (рис. 4.36).

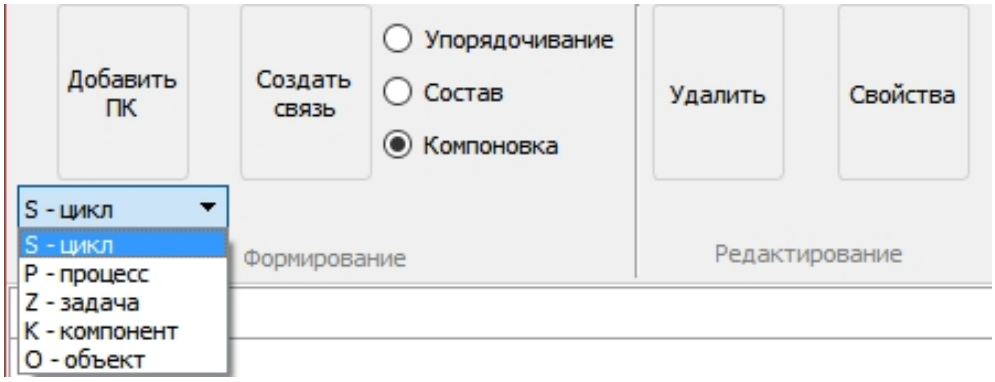


Рис. 4.36 Панель инструментов редактирования ОКС

Панель разделена на несколько составляющих:

- *Формирование*, включает в себя следующие функции:

  – **выпадающее меню с классами понятий** – выбор класса понятий (цикл, процесс, задача, компонент, объект), в котором будет создаваться предметная категория;

  – **добавить ПК –** добавление ПК в выбранном классе понятий;

  – **создать связь –** создание связей между двумя ПК;

  –**переключатели вида связей –** выбор вида создаваемой связи (упорядочивание, состав, компоновка) между ПК. При этом связь «упорядочивание» может быть установлена только между ПК одного класса, «состав» – между ПК смежных классов, а «компоновка» – при условии наличия у соединяемых ПК одного уровня связи «состав» с общей ПК вышестоящего уровня.

- *Редактирование*, включает в себя следующие функции:

  – **удалить –** удаление ПК, при этом он должен быть активен;

  – **свойства –** отображение информации о ПК (рис. 4.37) или связях. Для ПК классов «цикл», «задача» и «объект» задаётся номер группы и номер в группе (это делается как автоматически, так и вручную), а для ПК классов «процесс» и «компонент» выбирается группа (1 – функционирование, 2 – изготовление, 3 – проектирование, 4 – подготовка производства, 5 – исследование, 6 – эксплуатация, 7 – утилизация) и номер (1 – субъект, 2 – объект, 3 – процесс, 4 – документ, 5 – форма, 6 – звёздный разрушитель *(компонента не существует в реальности)* , 7 – средство-устройство, 8 – средство-программа, 9 – средство-документ, 10 – средство-форма/модель) соответственно.

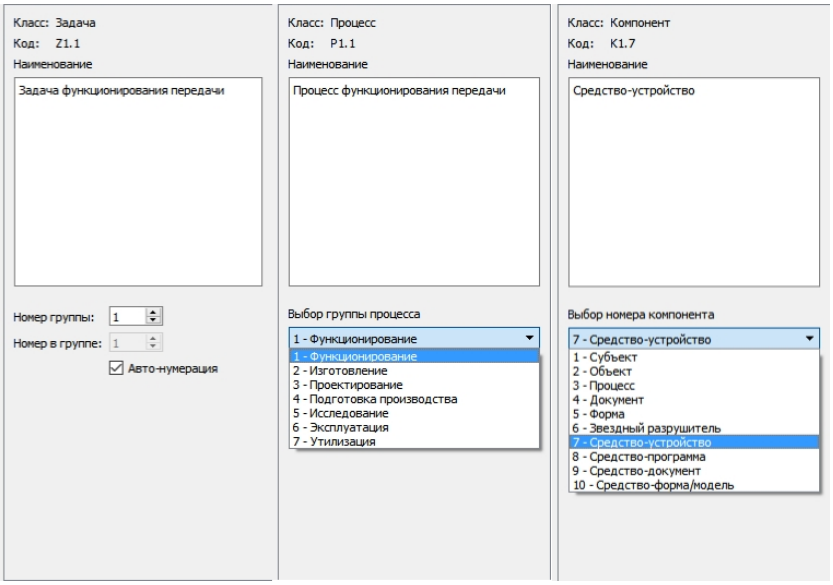


Рис. 4.37 Свойства для ПК классов «Задача», «Процесс» и «Компонент»

После создания концептуальной структуры предметной задачи необходимо перейти на вкладку *«Признаки»*, где добавляется список признаков. Для отображения объектов «узел», «сборочная единица» и «деталь» в КС в классе понятий «Объект» должны быть созданы ПК с соответствующими названиями. После окончания работы с вкладкой *«Признаки»* при переходе на вкладку *«Концептуальная структура»* будет отображена полная схема со списком признаков для каждой из ПК «узел», «сборочная единица» и «деталь» (рис. 4.38).

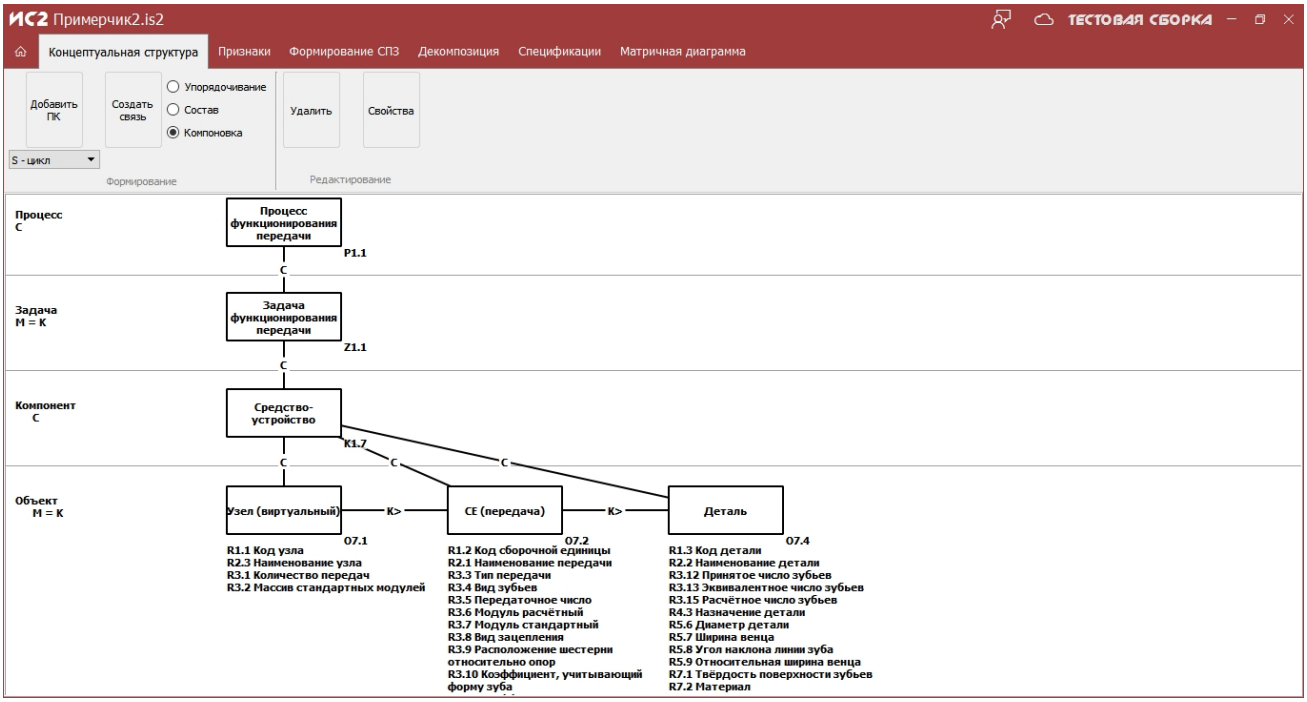


Рис. 4.38 *Концептуальная структура со списками признаков*

Признак – элемент начальной модели объектного уровня, отражающий именованное свойство объекта предметной области.

На вкладке «Признаки» предоставляется возможность присвоить каждой ПЗ следующие параметры: принадлежность к какой-либо группе и объекту, номер признака, его обозначение и название.

Кнопка «*Добавить*» **создаёт список признаков** (рис. 4.39).

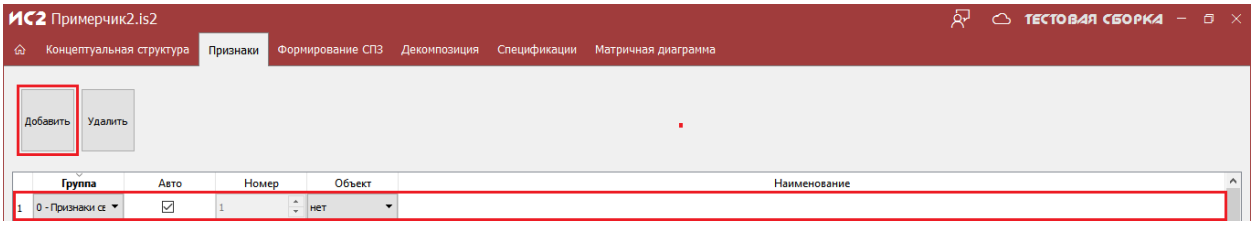


Рис. 4.39 Окно вкладки «Признаки»

После нажатия кнопки «*Добавить*» появляется строка, в которой указывается:

- принадлежность признака к той или иной группе. Группа выбирается из выпадающего списка;
- номер признака в этой группе; поставленная галочка в окошке позволяет автоматически присвоить номер для признака, снятая – выставляется вручную;
- принадлежность к объекту (наиболее известными являются узел, сборочная единица и деталь);
- наименование признака.

Кнопка «*Удалить*» **удаляет признак**, при этом объект должен быть активен.

4.4.6 Документирование диаграмм и спецификаций начальной модели

Документирование начальной модели подразделяется на два вида: графическое, включающее в себя декомпозицию и матричную диаграмму, и табличное, в которое входят формы A, B, C, D, D1, D2.

После работы с вкладками «Формирование СПД» и «Параметры» во вкладках «Декомпозиция» и «Матричная диаграмма» автоматически создаётся необходимая информация о созданной модели в графическом виде.

Декомпозиция в программе может отображаться *в нескольких вариантах*:

1. **Полная диаграмма** – отображение всей диаграммы с названиями каждого предметного действия. Для представления диаграммы в полном виде необходимо **деактивировать** кнопку «Краткая диаграмма» в верхнем левом углу (рис. 4.40);

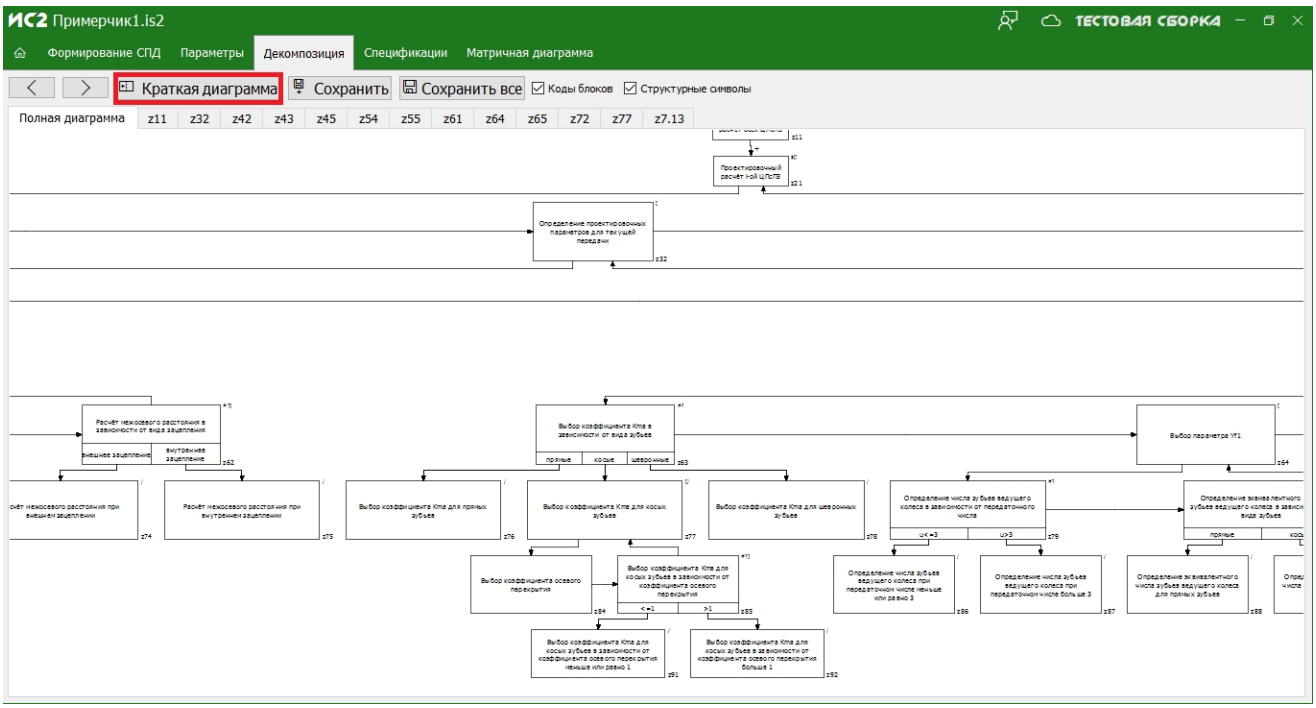


Рис. 4.40 Полное отображение системы предметных действий

2. **Краткая диаграмма** – отображение всей диаграммы с номерами предметных действий без их названий. Для представления диаграммы в кратком виде необходимо **активировать** кнопку «Краткая диаграмма» в верхнем левом углу (рис. 4.41);

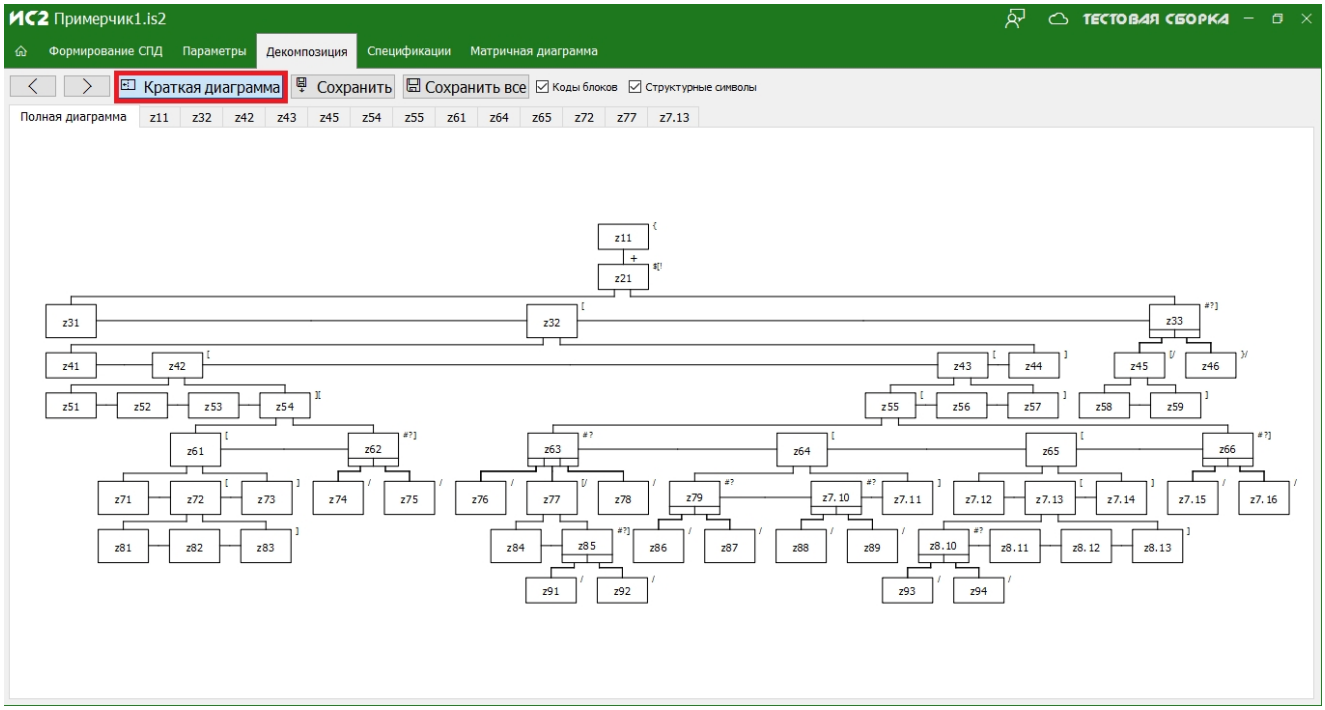


Рис. 4.41 Краткое отображение системы предметных действий

3. **По конструкциям** – отображение декомпозиции по сложным действиям. Для просмотра декомпозиции по конструкциям необходимо переключать вкладки с номерами предметных действий (рис. 4.42).

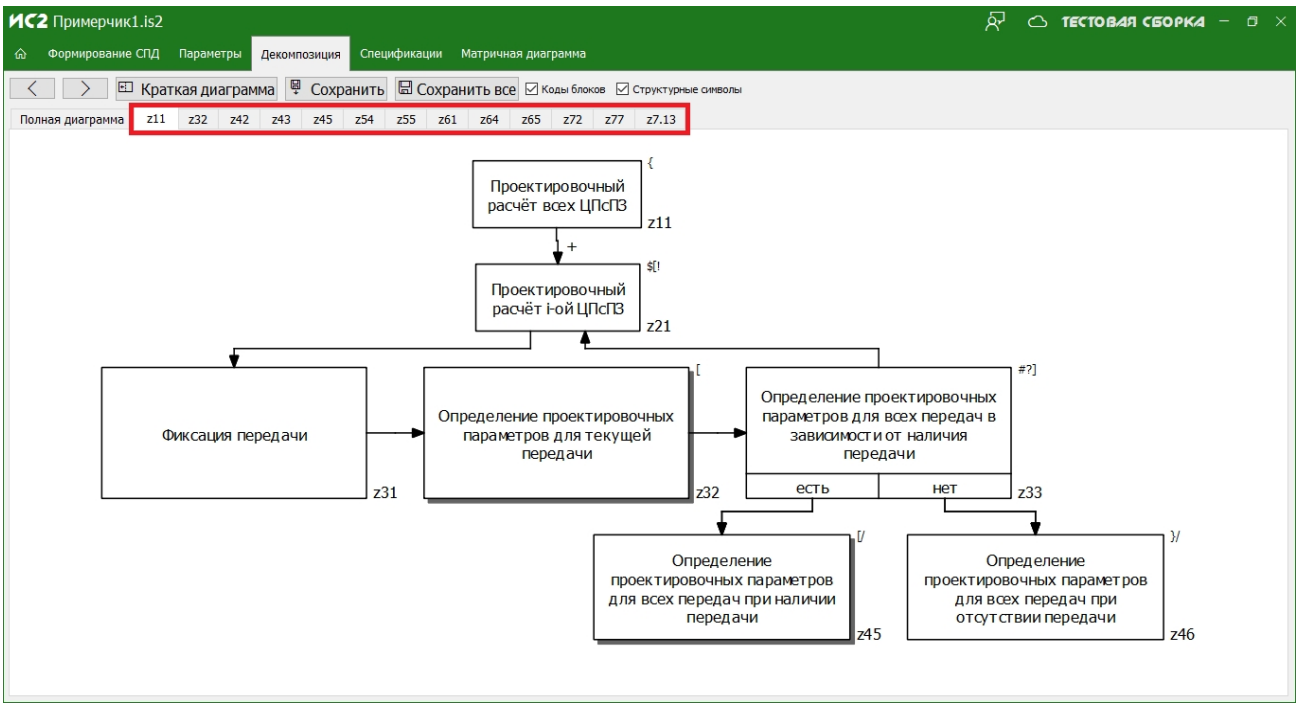


Рис. 4.42 Вкладки отображения декомпозиции по конструкциям

В разделе «*Декомпозиция*» есть возможность сохранения всей или отдельной части декомпозиции в формате изображения при помощи кнопок «*Сохранить*» и «*Сохранить все*».

Заголовки матричной диаграммы формируются в соответствии со спецификациями:

- **вертикальный заголовок** – формируется по форме А1, список кодов предметных действий, упорядоченных по принципу рекурсивного обхода декомпозиции, отображающей решение задачи (форма А1).
- **горизонтальный заголовок** – формируется по форме, список кодов параметров, упорядоченных по возрастанию кода в рамках каждого объекта.

Пересечение строки и столбца обозначает элементарную увязку П и ПД в виде специальных символов (+ . * # ? / [ ] { } ! $), расставляемых в соответствии с методикой моделирования.

Для просмотра матричной диаграммы необходимо перейти на соответствующую вкладку (рис. 4.43).

После работы с вкладками «Формирование СПД» и «Параметры» во вкладке «Спецификации» автоматически создаётся необходимая информация о созданной модели в табличном виде.

Для описания СПД модели используются две формы: А и В.

Форма А описывает структуру действий предметной задачи. В столбце «*Вид компоновки*» отображается название той конструкции, которая была создана во вкладке «*Формирование СПД*» (цикл, последовательность, альтернатива). В столбце «*Код ПД1*» происходит фиксация промежуточного ПД, а в столбцах «*Код ПД2*» и «*Код ПД3*» – перебирание всех ПД для

промежуточного, расположенных на уровне ниже. При этом «*Код ПД3*» выполняется только для последовательности, при цикле или альтернативе в этом столбце ставится «-». Для просмотра формы необходимо открыть соответствующую вкладку.

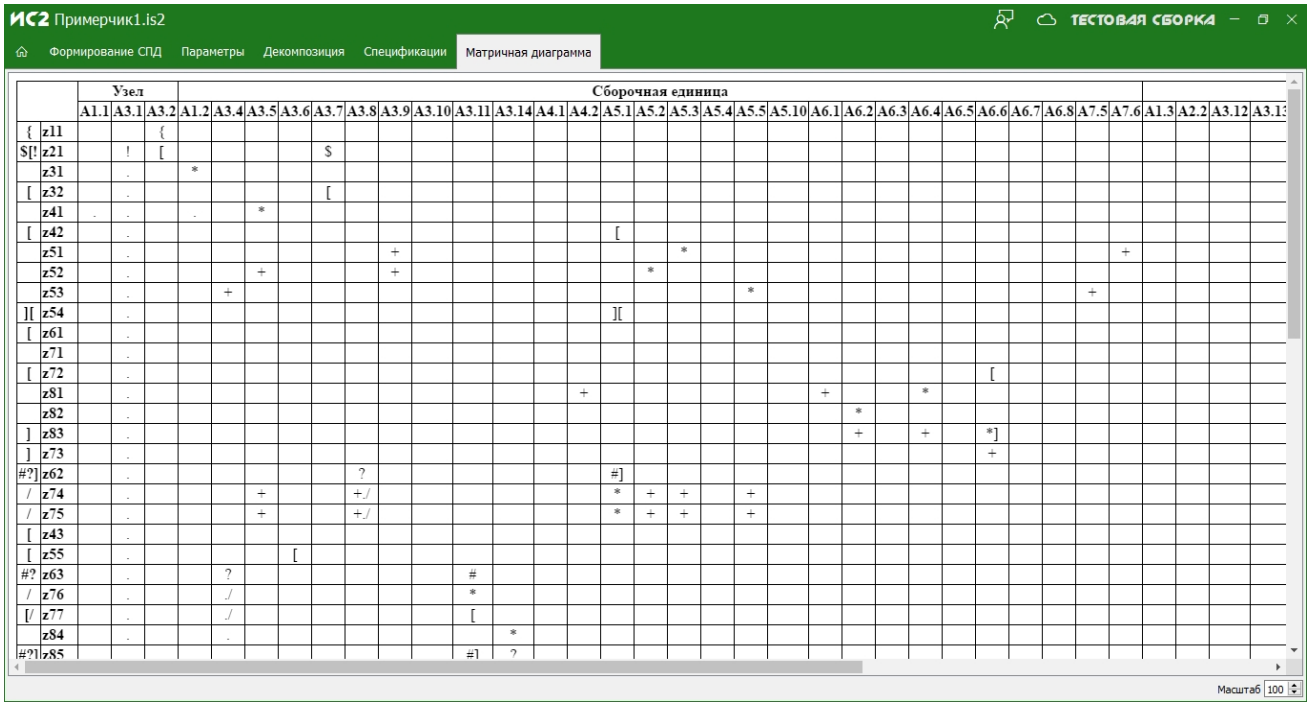


Рис. 4.43 Матричная диаграмма начальной модели

Форма B описывает предметные действия. Столбцы «*Код*» и «*Наименование*» формируются в зависимости от указанных в ПД кодировки и имени соответственно, созданных в «*Формировании СПД*». Содержимое столбцов «*Статус*» и «*Степень формализации*» указываются там же в панели «*Свойства*». Статус определяется следующим образом: ПД, в котором **раскрываются** другие ПД на уровне ниже, является промежуточным, а само расположенное на уровне ниже **нераскрывающееся** ПД – элементарным. Степень формализации для промежуточных ПД определяется **автоматически,** для элементарных – задается **вручную** в зависимости от того, каким является решение для ПД (формула, константа, таблица и т. д.). Имеется возможность отсортировать ПД в порядке возрастания их кодов, либо рекурсивным обходом, когда после промежуточного ПД происходит переход на уровень ниже (если на этом уровне вновь встречается промежуточное ПД, то после него совершается переход ещё на уровень ниже и т. д.). Для просмотра формы необходимо открыть соответствующую вкладку.

Для описания множества параметров модели используется форма D1.

Форма D1 описывает параметры предметной задачи. Столбцы «*Код*», «*Наименование*» и «*Обозначение*» формируются в зависимости от указанных во вкладке «*Параметры*» кодировки (A_{xy}, где **x** – код группы параметра, а **y** – порядковый номер параметра в группе), имени и условного обозначения параметра соответственно. Столбец «*Объект*» также создаётся в

зависимости от выбранного для параметра его принадлежности к объекту (узел, деталь или сборочная единица). Для просмотра формы необходимо открыть соответствующую вкладку.

Для описания увязки параметров и ПД модели используются две формы: D и D2.

Форма D описывает элементарные действия предметной задачи. В полях «*Код ПД*» и «*Наименование*» содержится информация о кодировках и названиях элементарных ПД, которые были созданы во вкладке «*Формирование СПД*». Столбец «*Формализация*» показывает виды решения элементарных ПД (аналитическая, эмпирическая, статистическая). В столбцах «*Функция*» и «*Аргументы*» для ПД указывается функция (*) и список зависимых от неё аргументов (+ и .). В «*Примечаниях*» поясняется, как определяется функция: задаётся эмпирически, выбирается из таблицы или рассчитывается по формуле. Для просмотра формы необходимо открыть соответствующую вкладку).

Форма D2 описывает потоки данных. В столбце «*Роли параметра*» указывается название каждого рассматриваемого параметра (функция, аргумент, аргумент по умолчанию). В поле «*Код ПД*» одно ПД указывается столько раз, сколько назначено для него параметров (функций, аргументов и аргументов по умолчанию) в панели «*Свойства*» вкладки «*Формирование СПД*». Столбец «*Код параметра*» содержит список кодировок определяемых параметров для каждого ПД. Для просмотра формы необходимо открыть соответствующую вкладку.

4.4.7 Документирование спецификаций концептуальной модели

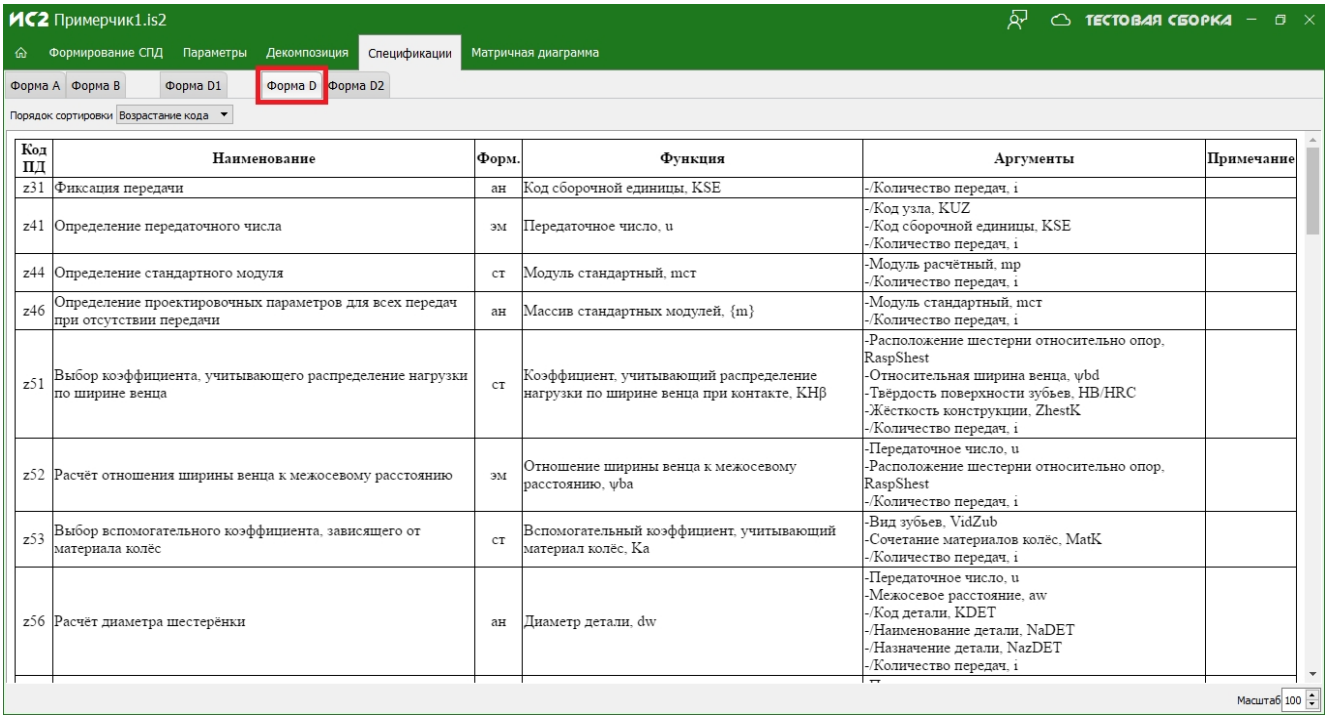
Документирование концептуальной модели подразделяется на два вида: графическое, включающее в себя декомпозицию и матричную диаграмму, и табличное, в которое входят формы F1, F2, F3, F4, F5, F6.

После работы с вкладками «*Формирование СПЗ*» и «*Признаки*» во вкладке «*Спецификации*» автоматически создаётся необходимая информация о созданной модели в табличном виде.

Для описания основной концептуальной структуры модели используются формы F1 и F2.

Форма F1 описывает предметные категории. В столбце «*Класс*» отображается класс предметной категории (S – цикл, P – процесс, Z – задача, K – компонент, O – объект, R – признак). Столбец «*Код*» показывает кодировку предметных категорий. В «*Наименовании*» указывается полное имя предметной категории. В столбце «*Тип*» определяется терминальность (T) или нетерминальность (NT) предметной категории. В «*Статусе*» рассматривается полнота (P) или неполнота (C). Столбец «*Оценка*» отражает потенциальную мощность множества экземпляров

ПК для реализации предметной задачи. Для просмотра формы необходимо открыть соответствующую вкладку.



Код ПД	Наименование	Форм.	Функция	Аргументы	Примечание
z31	Фиксация передачи	ан	Код сборочной единицы, KSE	-/Количество передач, i	
z41	Определение передаточного числа	эм	Передаточное число, u	-/Код узла, KUZ -/Код сборочной единицы, KSE -/Количество передач, i	
z44	Определение стандартного модуля	ст	Модуль стандартный, mct	-Модуль расчётный, mp -/Количество передач, i	
z46	Определение проектировочных параметров для всех передач при отсутствии передачи	ан	Массив стандартных модулей, {m}	-Модуль стандартный, mct -/Количество передач, i	
z51	Выбор коэффициента, учитывающего распределение нагрузки по ширине венца	ст	Коэффициент, учитывающий распределение нагрузки по ширине венца при контакте, KHβ	-Расположение шестерни относительно опор, RaspShest -Относительная ширина венца, ψbd -Твёрдость поверхности зубьев, HB/HRC -Жёсткость конструкции, ZhestK -/Количество передач, i	
z52	Расчёт отношения ширины венца к межосевому расстоянию	эм	Отношение ширины венца к межосевому расстоянию, ψba	-Передаточное число, u -Расположение шестерни относительно опор, RaspShest -/Количество передач, i	
z53	Выбор вспомогательного коэффициента, зависящего от материала колёс	ст	Вспомогательный коэффициент, учитывающий материал колёс, Ka	-Вид зубьев, VidZub -Сочетание материалов колёс, MatK -/Количество передач, i	
z56	Расчёт диаметра шестерёнки	ан	Диаметр детали, dw	-Передаточное число, u -Межосевое расстояние, aw -/Код детали, KDET -/Наименование детали, NaDET -/Назначение детали, NazDET -/Количество передач, i	

Рис. 4.44 Вкладки спецификаций. Форма D (D1)

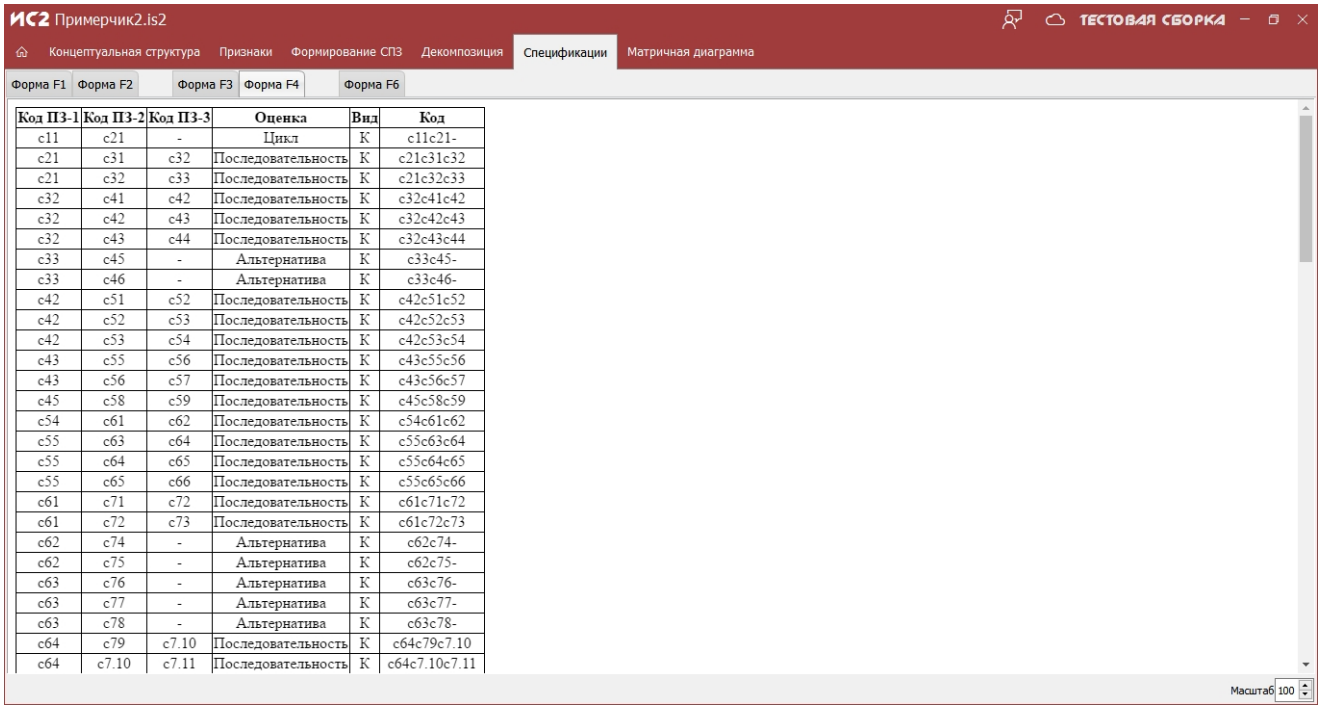
Форма F2 описывает бинарные связи предметных категорий. В поле «*Класс*» указывается кодировка классов связей ПК. В столбцах «*Код ПК1*», «*Код ПК2*», «*Код ПК3*» отображаются кодировки ПК, при этом их расположение зависит от вида связи ПК. В «*Наименовании связи ПК*» формируется конкатенация вида связи ПК и наименования исходной ПК. Поле «*Вид*» содержит вид связей ПК («У» – упорядочивание (связь фиксируется для «*Кода ПК2*» и «*Кода ПК3*»), «С» – состав (связь фиксируется для «*Кода ПК1*» и «*Кода ПК2*»), «К» – компоновка (связь фиксируется для «*Кода ПК1*», «*Кода ПК2*» и «*Кода ПК3*»)). В столбце «*Оценка*» отражаются числа (99 для упорядочивания, 1 или 99 для состава, ранг чисел или 99 для компоновки). Столбец «*Код*» содержит конкатенацию кодов первых трёх столбцов. Для просмотра формы необходимо открыть соответствующую вкладку.

Для описания СПЗ-1 модели используются формы F3 и F4.

Форма F3 описывает предметные зависимости 1-го рода. Столбцы «*Код*» и «*Наименование*» формируются в зависимости от указанных в ПЗ кодировки и имени соответственно, созданных в «*Формировании СПЗ-1*». Содержимое столбцов «*Статус*» и «*Степень формализации*» указываются там же в панели «*Свойства*». Статус определяется следующим образом: ПЗ, в которой **раскрываются** другие ПЗ на уровне ниже, является промежуточной, а сама расположенная на уровне ниже **нераскрывающаяся** ПЗ – элементарной. Степень формализации (аналитическая, алгоритмическая, статистическая, эмпирическая, информационная) для промежуточных ПЗ определяется **автоматически** (последовательность –

алгоритмическая, цикл и переключатель – аналитическая), для элементарных – **вручную** в зависимости от того, каким является решение для ПЗ (формула, константа, таблица и т. д.). В столбце «*Структурное свойство*» отображаются структурные символы для ПЗ. Имеется возможность отсортировать ПЗ в порядке возрастания их кодов, либо рекурсивным обходом, когда после промежуточной ПЗ происходит переход на уровень ниже (если на этом уровне вновь встречается промежуточная ПЗ, то после неё совершается переход ещё на уровень ниже и т. д.). Для просмотра формы необходимо открыть соответствующую вкладку.

Форма F4 описывает системы предметных зависимостей 1-го рода. (рис. 4.45) В столбце «*Оценка*» отображается название той конструкции, которая была создана во вкладке «*Формирование СПЗ*» (цикл, последовательность, альтернатива). В столбце «*Код ПЗ1*» происходит фиксация ПЗ, а в столбцах «*Код ПЗ2*» и «*Код ПЗ3*» – попарный перебор всех кодов действий, расположенных на уровне ниже. При этом «*Код ПДЗ*» выполняется только для последовательности, при цикле или альтернативе в этом столбце ставится «-». После перебора всех ПЗ операция повторяется, но теперь перебор зафиксированной в «*Коде ПЗ1*» ПЗ с



Код ПЗ-1	Код ПЗ-2	Код ПЗ-3	Оценка	Вид	Код
c11	c21	-	Цикл	К	c11c21-
c21	c31	c32	Последовательность	К	c21c31c32
c21	c32	c33	Последовательность	К	c21c32c33
c32	c41	c42	Последовательность	К	c32c41c42
c32	c42	c43	Последовательность	К	c32c42c43
c32	c43	c44	Последовательность	К	c32c43c44
c33	c45	-	Альтернатива	К	c33c45-
c33	c46	-	Альтернатива	К	c33c46-
c42	c51	c52	Последовательность	К	c42c51c52
c42	c52	c53	Последовательность	К	c42c52c53
c42	c53	c54	Последовательность	К	c42c53c54
c43	c55	c56	Последовательность	К	c43c55c56
c43	c56	c57	Последовательность	К	c43c56c57
c45	c58	c59	Последовательность	К	c45c58c59
c54	c61	c62	Последовательность	К	c54c61c62
c55	c63	c64	Последовательность	К	c55c63c64
c55	c64	c65	Последовательность	К	c55c64c65
c55	c65	c66	Последовательность	К	c55c65c66
c61	c71	c72	Последовательность	К	c61c71c72
c61	c72	c73	Последовательность	К	c61c72c73
c62	c74	-	Альтернатива	К	c62c74-
c62	c75	-	Альтернатива	К	c62c75-
c63	c76	-	Альтернатива	К	c63c76-
c63	c77	-	Альтернатива	К	c63c77-
c63	c78	-	Альтернатива	К	c63c78-
c64	c79	c7.10	Последовательность	К	c64c79c7.10
c64	c7.10	c7.11	Последовательность	К	c64c7.10c7.11

Рис. 4.45 Вкладки спецификаций. Форма F4

остальными ПЗ на уровне ниже уже происходит только в столбце «*Код ПЗ2*». «*Код ПДЗ*» в данном переборе не участвует, в этом столбце ставится «-». При этом в столбце «*Оценка*» будет отображаться «*1*» вместо названий конструкций. Столбец «*Вид*» указывает вид связи между ПЗ («К» – компоновка, «С» – состав). Последний столбец «*Код*» содержит конкатенацию кодов первых трёх столбцов. Для просмотра формы необходимо открыть соответствующую вкладку.

Столбец «*Роль ПК*» показывает, какие признаки являются функциями (Ф), аргументами (А) и аргументами по умолчанию (А-у). В поле «*Особая роль ПК*» обозначается назначение

структурных символов («[» и «]» – функция последовательности (ФП), «{» и «}» – функция цикла (ФЦ), «!» – счётчик цикла (СЦ), «$» – функция итерации (ФИ), «#» – функция альтернативы (ФА), «?» и «/» – аргумент альтернативы (АА)). Столбец «*Структурное свойство ПК*» включает в себя информацию о количестве символов для каждой ПЗ – единичная переменная («Е») или связанная в кортеже и одном экземпляре объекта («К»), в случае нескольких кортежей – «К1» и «К2». «*Объёмное свойство ПК*» отображает список значений («мульти») или одно значение («моно»). Для просмотра формы необходимо открыть соответствующую вкладку.

4.5 Применение ПК «ИС-2» и средств его развития

4.5.1 Применение ПК «ИС-2» в рамках инициативных проектов

Первая версия визуальной интегрированной среды была разработана в рамках инициативного проекта по гранду российского фонда фундаментальных исследований (РФФИ) № 13–07–00752 «Создание интегрированной среды поддержки проектирования и реализации прикладных автоматизированных систем на основе методологии автоматизации интеллектуального труда» в рамках которого использовалась для формирования и документирования алгоритмов и методик. В рамках проекта были разработанные основные положения настоящей работы.

В инициативном по гранду РФФИ №17-29-07056 «Разработка моделей и методов представления и обработки проблемно-ориентированных знаний, извлекаемых из научно-технических текстов и конструкторско-технологической документации» ПК «ИС-2» применялся при разработке комплекса методик организации и хранения электронных фондов научно-технических публикаций и технической документации состоящего из: [51] [52]

- методики формирования электронного каталога методики перевода традиционных документов в электронную форму (сканирование) и формирования расширенного электронного каталога и фонда электронных образов документов;
- методики фрагментации электронных образов документов;
- методики обработки фрагментов;
- методики разметки семантических элементов документа.

В инициативном по гранду РФФИ 17-29-07057 «Разработка методов реструктуризации и интеграции для семантических и синтаксических представлений при создании систем

автоматизации процессов проектирования и управления» ПК «ИС-2» применялся в качестве среды автоматизации проектирования и в качестве объекта разработки.

В среде были сформирован комплекс расчетных предметных задач подлежащих интеграции в единую модель. [53] [54]

В среде разработаны методики реструктуризации моделей по временной и пространственной характеристики и методики интеграции семантических и синтаксических представлений предметных задач.

С применением средств развития ПК «ИС-2» были разработаны экспериментальные версии функциональных модулей для поддержки процедур интеграции начальных, концептуальных и инфологических моделей. Экспериментальные версии работают в программном комплексе «ИС-2» (рис 4.46). [55] [56]

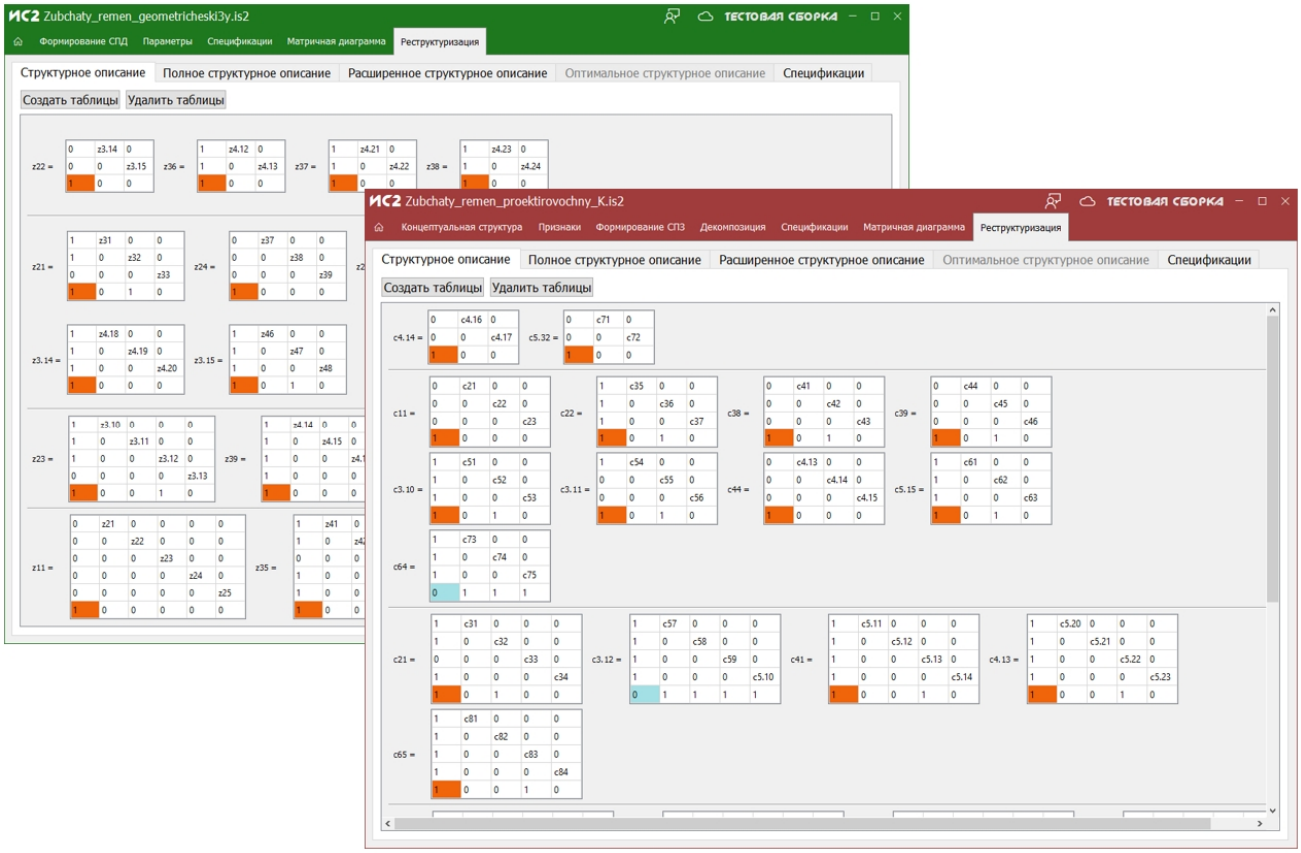


Рис. 4.46 Пример работы модуля реструктуризации модели

4.5.2 Применение ПК «ИС-2» в учебном процессе

Разработанные в рамках диссертационной работы программные средства, а именно программный комплекс «ИС-2», и его расширения были использованы в учебном процесса

кафедры информационных технологий и вычислительных систем (ИТиВС) ФГБОУ ВО «МГТУ «СТАНКИН» при подготовке бакалавров, магистров и аспирантов по направлениям:

- бакалавриат, 09.03.01 «Информатика и вычислительная техника», профили подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем», «Модели, методы и программное обеспечение анализа проектных решений»;
- магистратура, 09.04.01 «Информатика и вычислительная техника», профиль подготовки: «Управление программными продуктами и проектами»;
- аспирантура, 09.06.01 «Информатика и вычислительная техника», профиль подготовки «Системный анализ, управление и обработка информации (в машиностроении)».

В учебном процессе Кафедры ПК «ИС-2» применялся в следующих дисциплинах.

Дисциплина «Базы данных». ПК «ИС-2» применялся в качестве среды поддержки создания курсового проекта на тематику проектирования узлов деталей машин. Целью работы является формирование начального модельного представления предметной задачи, посвященной выполнению этапа проектировочного расчета зубчатой или ременной передачи. Полученная формальная модель является основой для реализации средства автоматизации расчета. [57]

Дисциплина «Концептуальное проектирование проектных задач». Средствами ПК «ИС-2» обучающиеся формируют концептуальные модели предметных задач на основе начальных моделей. Работа является продолжением курсового проекта с дисциплины «Базы данных» и выполняется как курсовой проект. [58]

Дисциплина «Проектирование автоматизированных систем». ПК «ИС-2» применяется для разработки инфологической модели автоматизированной системы расчета зубчатых передач. Обещающиеся формируют программно-независимую модель будущей АС на основе концептуальной модели из курсового проекта по дисциплине «Концептуальное проектирование проектных задач». [59]

Вышеперечисленные дисциплины позволяют обучающимся в полной мере применить ПК «ИС-2» для поэтапного проектирования автоматизированной системы машиностроительного назначения и применить на практике методы МАИТ.

Дисциплина «Анализ деятельности проектно-конструкторских организаций». Обучающиеся формируют последовательно-параллельную схему деятельности организации с помощью средств поддержки процедуры аналитической обработки моделей МАИТ, реализованной для ПК «ИС-2». [60]

Дисциплины «Технологии разработки программного обеспечения» для магистратуры и «Технология разработки программного обеспечения» для бакалавриата. В рамках лабораторных занятий и самостоятельной работы в ПК «ИС-2» обучающие формируют программно-независимые модели разрабатываемых автоматизированных систем, с последующим анализом

их корректности методами МАИТ. [61] [62]

Так же ПК «ИС-2» и средства его развития применяется при выполнении выпускных квалификационных работ бакалавров и магистров в качестве учебно-проектной среды. Студенты и сотрудники кафедры ИТиВС активно учувствуют в развитии ПК «ИС-2» постоянно обновляя текущие функциональные модули и разрабатывая новые. На рисунке 4.47 приведен прототип модуля трехмерной визуализации модели в целом, позволяющего наглядно отобразить содержательную структуру модели автоматизируемой задачи [63]. Модуль разработан магистрантом кафедры ИТиВС Глинкиным М.О. в выпускной работе на тему «Исследование и анализ графических нотаций методов моделирования и разработка модуля визуализации 3D-диаграмм для интегрированной среды «ИС-2» под руководством канд. техн. наук, доцента Новоселовой О.В.

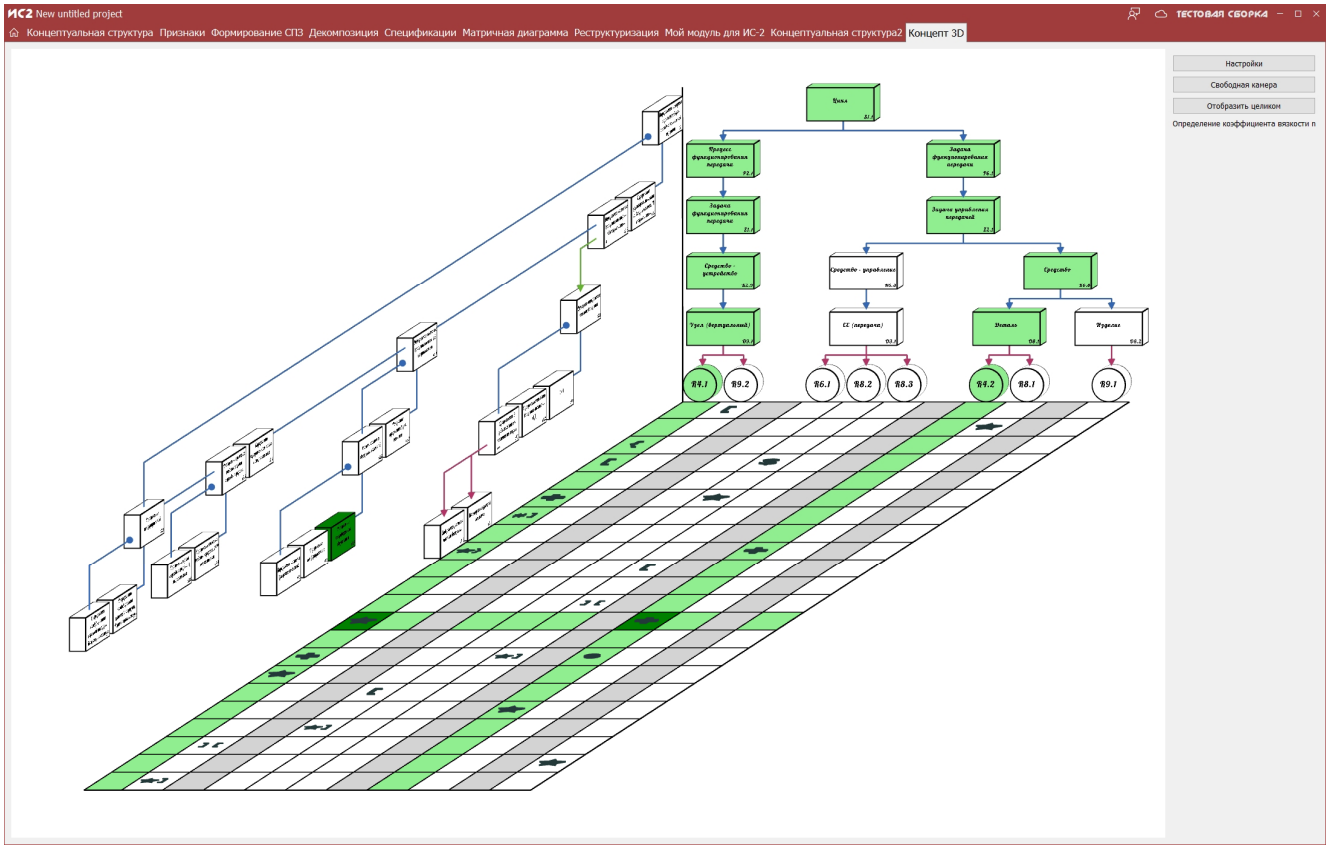


Рис. 4.47 Прототип модуля трёхмерной визуализации модели

Свидетельство о государственной регистрации версии ПК «ИС-2», используемой в учебном процессе («ИС-2-у») представлено в приложении Д.

Программные средства поддержки МАИТ, входящие в состав интегрированной среды, также развиваются в научно-квалификационных работах аспирантов Кафедры.

4.5.3 Оценка эффективности применения ИС-2

Эффективность работы в ПК «ИС-2» по временной характеристики была оценена относительно связки самых функциональных комплексов поддержки МАИТ – «Concept» и «Infology». Оценка проводилась при формировании комплекса локальных концептуальных и инфологических моделей расчетных задач при проектировании узлов деталей машин (зубчатых и ременных передач) в рассматриваемых средах. В таблице 4.1 представлено усредненное время формирования одной модели.

Таблица 4.1 Структурные свойства предметных действий

Этап	Процедура этапа	Временные показатели	
		Concept + Infology	ПК «ИС-2»
Концептуальное моделирование	Формирование графических исходной модели (вручную)	10 часов	Не требуется
	Ввод данных	2 часа	6 часов
	Сверка контрольных диаграмм и внесение правок	2 часа	Не требует сверки
	Анализ модели и внесение правок	3 час	1 час
	Документирование КМ	0,2 часа	0,3 часа
	Отображение графических диаграмм в инфологическую модель (вручную)	1,3 часа	0 – происходит автоматически
Инфологическое моделирование	Ввод данных	2 часа	Не требуется
	Анализ модели и внесение правок	1 час	0,5 часа
	Сверка контрольных диаграмм и внесение правок	0,5 часа	Не требуется
	Документирование	0,2 часа	0,3 часа
	Сумма	**22,2 часа**	**8,1 часов (-64%)**

Итоговое время на формирование локальных моделей, затрачиваемое при работе в ПК «ИС-2», снизилось с 22,2 часов до 8,1 часов за счет ряда факторов:

1. Отсутствие необходимости ручного формирована предварительной модели. Весь процесс моделирование проходит в ПК «ИС-2».

2. Отсутствие шага сверки предварительных контрольных диаграмм, построенных вручную, с созданными в вычислительной среде.

3. Полностью автоматического переноса данных модели между средствами концептуального и инфологического моделирования.

4. Автоматизированного анализа корректности модели, выполняемого в функциональных

модулях ПК «ИС-2». Ранее этот этап проводился в ручном режиме.

Однако, время необходимое на ввод данных в среду увеличилось, вследствие сращивания процессов ввода и проектирования модели. Ранее, перед внесением информации в вычислительную среду, аналитик располагал уже составленной моделью задачи. В ПК «ИС-2» необходимости в предварительном моделировании нет, поэтому творческий процесс создания модели происходит непосредственно в визуальных редакторах.

Так же незначительно увеличилось время документирования моделей, так как предыдущие средства поддержки МАИТ были интегрированы с текстовым редактором Microsoft Word. Недостаток можно устранить при разработке модулей интеграции для ПК «ИС-2».

Оценка эффективности применения средств развития ПК «ИС-2» выполнялась на основании сравнения сложности разработки существующих средств поддержки процедур формирования моделей МАИТ – комплексов «Model», «Concept» и «Infology», и аналогичных им по функциональности конфигураций ПК «ИС-2», собранных на базе соответствующих функциональных модулей.

Из-за давности разработки программных комплексов и отсутствию документации, сопровождающей процесс разработки, автор настоящей работы может оперировать только исходным кодом этих систем. В качестве критерия оценки трудозатрат был выбран объем программного кода, требуемого для создания средств поддержки.

Рассмотренные созданные ранее средства поддержки процедур формирования начальной, концептуальной и инфологической моделей созданы на базе различных версий Microsoft Access и использованием языка Visual Basic for Applications (VBA). ПК «ИС-2» разработан на языке C++. Различные программно-технические среды создания затрудняют подобное сравнение, так как одни и те же алгоритмические конструкции, написанные на разных языках, могут иметь различающийся объем. Поэтому сравнение лишь оценочное.

Результат сравнения объема кода специфической части модулей ПК «ИС-2» и созданными ранее комплексами «Model», «Concept» и «Infology» представлен в таблице 4.2.

Из представленных данных видно, что объем кода модулей ПК «ИС-2» в три раза меньше, чем у рассмотренных программных комплексов. Такая разница обусловлена особенностью «ИС-2», заключающейся в использовании универсальных компонентов. Исходные коды «Model», «Concept» и «Infology» содержат множество функциональных пересечений, так как разрабатывались независимо друг от друга. В то же время функциональные модули «ИС-2» лишены этого недостатка, так как входят в единую среду и используют включенные в нее общие решения.

Такое строение среды повышает эффективность реализации будущих средств поддержки процедур методологии. Например, при выполнении инициативного проекта по гранду РФФИ 17-

29-07057 «Разработка методов реструктуризации и интеграции для семантических и синтаксических представлений при создании систем автоматизации процессов проектирования и управления» однократно реализовав инвариантные части процедур интеграции и реструктуризации компонентов модельных представлений в универсальных компонентах ПК «ИС-2», стало возможно их использование в функциональных модулях ПК «ИС-2». Таким образом объём работы программистов при разработке модулей поддержки этих процедур для трех моделей сократился втрое.

Таблица 4.2 Сравнение объема программного кода средств поддержки процедур формирования моделей

Процедура	Объем программного кода существующих средств	Объем специфического программного кода функциональных модулей ПК «ИС-2»
Формирование начальной модели	~160 кбайт (Model)	~50 кбайт
Формирование концептуальной модели	~160 кбайт (Model)	~100 кбайт
Формирование инфологической модели	~350 кбайт (Infology)	~110 кбайт
Сумма	**820 кбайт**	**260 кбайт (-68%)**

4.6 Выводы по главе

Визуальная интегрированная среда реализована в программном комплексе «ИС-2». Практика применения ПК в проектной деятельности показала эффективность её работы относительно существующих средств поддержки процедур МАИТ за счет использования визуальных редакторов для формирования моделей и автоматизации процессов переноса информации между средствами поддержки процедур МАИТ.

Особенность структуры ПК, заключающаяся в применении универсальных компонентов, повысило эффективность реализации.

ЗАКЛЮЧЕНИЕ

Выполненные исследования и практическая работа позволили получить следующие выводы и результаты:

1. Диссертация представляет научно-квалификационную работу, в которой содержится решение задачи, заключающейся в повышении эффективности процесса создания автоматизированных систем за счет разработки метода моделирования визуальной интегрированной среды проектирования автоматизированных систем в рамках методологии автоматизации интеллектуального труда, а также создания средств поддержки управления её развитием.

2. Анализ существующих методов, подходов и инструментальных средств их поддержки позволил установить, что по сравнению с методами традиционного, структурного и объектно-ориентированного подхода методология автоматизации интеллектуального труда имеет ряд преимуществ: ориентирована на промышленный способ создания автоматизированных систем; наличие формализованных модельных представлений, описывающих проектируемую систему на каждом этапе создания АС; наличие универсального представления, обеспечивающего порождение и интеграцию прикладных моделей под предметные задачи.

3. Сравнительный анализ инструментальных средств поддержки методов объектно-ориентированного и когнитивного подходов показал, что средства первого не позволяют охватить весь процесс создания АС, а их объединение в единую информационно-связную инструментальную среду требует значительных усилий. В то же время, разработанная ранее интегрированная среда поддержки создания САПР по МАИТ лишена этого недостатка, однако её проектирование и реализация выполнялись без учета особенностей структур моделей МАИТ и с использованием устаревших на сегодняшний день программно-технических средств, что делает неэффективным её дальнейшее применение и развитие.

4. Выявлены взаимосвязи между характеристиками содержательных функций проектирования АС, функциями управления функционированием визуальной интегрированной среды и функциями её развития.

5. На основании выделенных взаимосвязей разработан метод моделирования новой визуальной интегрированной среды, описывающий среду как множество специализированных и универсальных файлов различных типов и связей между ними, позволяющий описать состав и структуру функционирования среды и обеспечивающий процессы её формирования и эффективного развития.

6. Разработана методика управления развитием визуальной интегрированной среды,

позволяющая осуществлять поэтапную модернизацию её функциональных модулей. Данная методика обеспечивает регламентацию процесса конфигурирования среды путем изменения её файлового состава. При этом, конфигурирование специализированных файлов, приводит к изменениям отдельного модуля среды, а конфигурирование универсальных – к изменениям, которые затрагивают группу модулей.

7. Разработаны инструментальные средства поддержки управления развитием, предоставляемые разработчикам программные средства для создания и интеграции новых модулей в среду. Также были разработаны прототипы модулей среды для формирования и документирования локальных моделей на этапах предпроектного обследования и проектирования автоматизированных систем.

8. Методические и программные средства были использованы: в рамках инициативных проектов при поддержке Российского Фонда Фундаментальных Исследований № 13-07-00752 «Создание интегрированной среды поддержки проектирования и реализации прикладных автоматизированных систем на основе методологии автоматизации интеллектуального труда», № 17-29-07056 «Разработка моделей и методов представления и обработки проблемно-ориентированных знаний, извлекаемых из научно-технических текстов и конструкторско-технологической документации», № 17-29-07057 «Разработка методов реструктуризации и интеграции для семантических и синтаксических представлений при создании систем автоматизации процессов проектирования и управления», в учебном процессе кафедры Информационных технологий и вычислительных систем ФГОУ ВО «МГТУ «СТАНКИН» при подготовке бакалавров, магистров и аспирантов по направлениям подготовки 09.03.01, 09.04.01, 09.06.01 «Информатика и вычислительная техника» в качестве учебно-проектного средства поддержки дисциплин и в деятельности организации ООО «Специальная интеграция».

СПИСОК СОКРАЩЕНИЙ

АС – автоматизированная система
ВИС – визуальная интегрированная среда
ИЛМ – инфологическая модель
ИС – интегрированная среда
ИСЕ – именованная структурная единица
КМ – концептуальная модель
МАИТ – методология автоматизации интеллектуального труда
МД – матричная диаграмма
НМ – начальная модель
ОИЛС – основная инфологическая структура
ОКС – основная концептуальная структура
ПД – предметное действие
Пд – предметный доступ
Пд-1 – предметный доступ первого рода
Пд-2 – предметный доступ второго рода
ПЗ – предметная зависимость
ПЗ-1 – предметная зависимость первого рода
ПЗ-2 – предметная зависимость второго рода
ПМ – предметная манипуляция
ПМ-1 – предметная манипуляция первого рода
ПМ-2 – предметная манипуляция второго рода
ПО – программное обеспечение
САПР – система автоматизированного проектирования
СПД – система предметных действий
СПд – система предметных доступов
СПд-1 – система предметных доступов первого рода
СПд-2 – система предметных доступов второго рода
СПЗ – система предметных зависимостей
СПЗ-1 – система предметных зависимостей первого рода
СПЗ-2 – система предметных зависимостей второго рода
СПК – схема предметных категорий
СПМ – система предметных манипуляций
СПМ-1 – система предметных манипуляций первого рода
СПМ-2 – система предметных манипуляций второго рода

СПИСОК ЛИТЕРАТУРЫ

1. Соловьев, Н.А. Системы автоматизации разработки программного обеспечения: учебное пособие / Н.А. Соловьев, Е.Н. Чернопрудова, Оренбургский гос. ун-т. – Оренбург: ОГУ, 2012. – 191 с..
2. Новоселова, О.В. Исследование методологий и методов проектирования автоматизированных систем различного назначения. / О.В. Новоселова, Г.Д. Волкова, О.Г.Григорьев // //Журнал «Вестник МГТУ «СТАНКИН» – М.: ФГБОУ ВО «МГТУ «СТАНКИН», №1 (19), 2012. – с.104-106.
3. Абрамова, Л.В. Инструментальные средства информационных систем: учебное пособие / Л.В. Абрамова; Сев. (Арктич.) федер. ун-т им. М.В. Ломоносова. – Архангельск: ИПЦ САФУ, 2013. – 118 с.
4. Жуковский О. И. Информационные технологии и анализ данных : учебное пособие / О. И. Жуковский. — Томск : Эль Контент, 2014. — 130 с.
5. Гриценко Ю. Б. Архитектура предприятия : учебное пособие / Ю. Б. Гриценко. — Томск: Эль Контент, 2011. — 206 с..
6. Схиртладзе, А. Г. Проектирование единого информационного пространства виртуальных предприятий : учебник / А. Г. Схиртладзе, А. В. Скворцов, Д. А. Чмырь. – Изд. 2-е, стер. – М. ; Берлин : Директ-Медиа, 2017. – 616 с.: ил.
7. Методология функционального моделирования IDEF0 : руководящий документ / ПНЦ "Прикладная локистика" – ИПК Издательство стандартов – 2000 – 75 с.
8. Бритов, Г. С. Методы построения OSTN-диаграмм IDEF3-технологий / Г. С. Бритов // Системный анализ и логистика. – 2014. – № 11. – С. 3-12.
9. Сенченко П. В. Организация баз данных : учебное пособие / П. В. Сенченко. — Томск : ФДО, ТУСУР, 2015. — 170 с..
10. Захарова Е.Я. Информационные системы : учеб. пособие / Е.Я. Захарова, О.В. Милёхина. - Новосибирск : Изд-во НГТУ, 2010.-126 с.
11. Исакова А. И. И. Научная работа : учебное пособие / А. И. Исакова. – Томск : ФДО, ТУСУР, 2016. – 109 с..
12. Омарова, П. З. Моделирование бизнес-процессов на предприятии при помощи среды моделирования ARIS / П. З. Омарова, К. М. Магомедова // Научное обозрение. Педагогические науки. – 2019. – № 2-2. – С. 91-95.
13. Самсонова, М. В. Управление процессами : учебно-практическое / М. В. Самсонова. – Ульяновск : УлГТУ, 2014. – 187 с.

14. Каменнова, М. Моделирование бизнеса. Методология ARIS. Практическое руководство. / М. Каменнова, А. Громов, М. Ферапонтов, А. Шматалюк. – М.: Весть-МетаТехнология, 2002. – 333с..
15. Скворцов, В. И. Технологические основы использования системы ARIS Toolset 7.0. / В.И. Скворцов – М.: Издательство ДИАЛОГ-МИФИ, 2006 – 288 с.
16. OMG Meta Object Facility (MOF) Core Specification, Version 2.5.1 / Object Management Group. - OMG Document Number: formal/2016-11-01, Date: November 2016, Standard document URL: http://www.omg.org/spec/MOF/2.5.1.
17. OMG Unified Modeling Language(OMG UML), Version 2.5.1 / Object Management Group. - OMG Document Number: formal/2017-12-05, Date: December 2017, Normative Reference: http://www.omg.org/spec/UML/2.5.1.
18. Information integration for concurrent engineering (IICE). IDEF4 object-oriented design method report. Version 2.0 // Knowledge Based Systems, inc. – 1995. – 169 p.
19. Natalya F. Noy and Deborah L. McGuinness. Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001- 0880, March 2001.
20. Information Integration for Concurrent Engineering (IICE). IDEF5 Method Report. / [Perakath C. Benjamin, Ph.D, at al] // Knowledge Based Systems, Inc. – 1995. – 187 p.
21. Varun Grover, William J. Kettinger. Process Think: Winning Perspectives for Business Change in the Information Age. - 2000.
22. Орлова Ю. А. Планирование и учет ресурсов предприятия / Ю. А. Орлова — «БИБКОМ», 2012.
23. SNOMED clinical terms: overview of the development process and project status. /M. Q. Stearns, C. Price, K. A. Spackman, A. Y. Wang.// National Center for Biotechnology Information - URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2243297/.
24. Evaluation of Embeddings of Laboratory Test Codes for Patients at a Cancer Center / Lorenzo A. Rossi C.S.J.M.F.Z./.0.D.H.O.4.1.0.
25. Волкова, Г.Д. Автоматизация концептуального моделирования проектно-конструкторских задач при создании САПР машиностроительного назначения. / Г.Д. Волкова, М.В. Щукин // Материалы международной конференции «Информационные технологии в образовании, технике и медицине». Волгоград, 2000г., с.31-34.
26. Волкова, Г.Д. Разработка новых методов и средств формирования и интеграции взаимосвязанных семантических и синтаксических представлений проектно-

конструкторских задач с целью повышения эффективности создания САПР машиностроительного назначения, дис. … д-ра техн. наук: 05.13.12 / Галина Дмитриевна Волкова. – М., 1997. - 608 с..

27. Лежебоков, А.А. Л401 Программные средства и механизмы разработки информационных систем: учебное пособие / А.А. Лежебоков; Южный федеральный университет. – Таганрог: Издательство Южного федерального университета, 2016. – 84 с..
28. Назаров, И. Р. Применение методологии разработки программного обеспечения "архитектура, управляемая моделью" / И. Р. Назаров, А. В. Аникин // Сборник научных трудов Новосибирского государственного технического университета. – 2016. – № 2(84). – С. 107-115
29. 20.03.22 О.п.R.S.A.D..L.U.2.0.0./.C.I./.U.H.I.C.S.A.5.R.S.A.P.O.(.o..
30. Леоненков, А.В. Визуальное моделирование в среде IBM Rational Rose 2003, 2-е издание / А.В. Леоненков. - Национальный Открытый Университет "ИНТУИТ", 2016. - 193 с.
31. Eclipse Modeling Framework (EMF) // Eclipse. 2018. URL: http://www.eclipse.org/modeling/emf/ (дата обращения: 23.07.2018).
32. Microsoft Visual Studio URL: https://visualstudio.microsoft.com/ru/ (дата обращения: 5.8.2018).
33. LINQ to SQL Tools in Visual Studio // Microsoft Developer Network. URL: https://msdn.microsoft.com/en-us/library/bb384429.aspx (дата обращения: 2018.6.4).
34. CASEBERRY. Автоматизированное проектирование и программирование / ООО "Институт Информационных Систем" 2018. URL: http://caseberry.net/ (дата обращения: 14.05.2018).
35. Бычкова, Н.А. Разработка метода и инструментальных средств визуального моделирования и документирования системы знаний предметных задач при проектировании сапр машиностроительного назначения, дис.. канд-та. техн. наук: 05.13.12 / Наталья Александровна Бычкова. – М., 2006. - 275 с..
36. Скородумов, О.И. Разработка моделей и инструментальных средств формирования, визуализации и обработки инфологических представлений при проектировании САПР машиностроительного назначения, выпускная квалификационная (магистр) работа, специальность 552800 «Информатика и вычислительная техника» / О.И. Скородумов. - МГТУ "СТАНКИН" - 2004.
37. Новоселова, О.В. Моделирование предметных задач на начальных этапах автоматизации проектной деятельности: учеб. пособие / О.В. Новоселова. – М.: ФГБОУ ВО «МГТУ

«СТАНКИН», 2016. – 100 с.: ил.

38. Гаврилов, А.Г. Разработка средств управления функционированием и конфигурированием интегрированной среды поддержки проектирования прикладных автоматизированных систем (ПАС) / А.Г. Гаврилов, В.П. Максимов // Труды XV научной конференции «Математическое моделирование и информатика», М.: ИЦ ФГБОУ ВПО МГТУ «СТАНКИН», 2013 – с.31-33.
39. Волкова, Г.Д. Методология автоматизации интеллектуального труда / Г.Д. Волкова. – М.: ЯнусК, 2013. – 104 с.
40. Волкова, Г.Д. Концептуальное моделирование проектных задач: учеб. пособие / Г.Д.Волкова.- М.: ФГБОУ ВО «МГТУ «СТАНКИН», 2015.- 117 с.:ил.
41. Волкова, Г.Д. Проектирование автоматизиро-ванных систем в машиностроении : учебное пособие. / Г.Д. Волкова, О.В. Новоселова, Е.Г. Семячкова – М.: МГТУ «Станкин», 2002 – 162с.
42. Протасова, С. В. Разработка метода и средств поддержки взаимосвязанного моделирования проектных и управленческих процессов при автоматизации деятельности проектно-конструкторской организации, : специальность 05.13.12 "Системы автоматизации проектирования (по отраслям)" : диссертация на соискание ученой степени кандидата технических наук / Протасова Светлана Витальевна. – Москва, 2007. – 207 с..
43. Тюрбеева, Т. Б. Разработка метода моделирования процессов жизненного цикла прикладных автоматизированных систем, обеспечивающего формирование нормативно-методической среды их поддержки :, специальность 05.13.01 "Системный анализ, управление и обработка информации (по отраслям)" : автореферат диссертации на соискание ученой степени кандидата технических наук / Тюрбеева Татьяна Борисовна. – Москва, 2015. – 22 с..
44. Волкова, Г.Д. Методология автоматизации интеллектуального труда. Часть 1 / Г.Д. Волкова // Межотраслевая информационная служба. - 2009. - №1(150). - С.4-30.
45. Волкова, Г.Д. Методология автоматизации интеллектуального труда. Часть 2 / Г.Д. Волкова // Межотраслевая информационная служба. - 2009. - №2(147). - С.9-20.
46. Волкова, Г.Д. Методология автоматизации интеллектуального труда. Часть 3 / Г.Д. Волкова // Межотраслевая информационная служба. - 2009. - №3(148). - С.10-23.
47. Волкова, Г.Д. Методология автоматизации интеллектуального труда. Часть 4 / Г.Д. Волкова // Межотраслевая информационная служба. - 2010. - №4(149). - С.15-30.
48. Володин, Д. А. Разработка метода и средств формирования и развития интегрированной

среды поддержки создания САПР машиностроительного назначения : специальность 05.13.12 "Системы автоматизации проектирования (по отраслям)" , : диссертация на соискание ученой степени кандидата технических наук / Володин Дмитрий Александрович. – Москва, 2008. – 312 с..

49. Abramyan, M. E. User interface development based on Windows Forms class library : text-book / M. E. Abramyan ; Southern Federal University. – Rostov-on-Don ; Taganrog : Southern Federal University Press, 2021. – 278 p.,.
50. Шлее, М. Qt 5.3. Профессиональное программирование на C++. / Макс Шлее — СПб.: БХВ-Петербург, 2015. — 928 с.: ил.
51. Volkova, G. D. Organization of Informational and Intellectual Resources at Industrial Enterprises / G. D. Volkova, T. B. Tyurbeeva, P. So // Russian Engineering Research. – 2019. – Vol. 39. – No 12. – P. 1050-1052. – DOI 10.3103/S1068798X19120244.
52. Volkova, G. D. Application of cognitive technologies for modeling of design and technological knowledge andautomation of design activities / Galina Volkova, Tatyana Turbeeva, Oleg Grigoriev, [et al.] // Материалы Международной научно-технической конференции «Современные направления и перспективы развития технологий обработки и оборудования в машиностроении 2020 (ICMTMTE 2020)», Журнал IOP Conference Series: Materials Science and Engineering Number 3. November 2020. 032080. IOP Conference Series: Materials Science and Engineering, Vol 971., Журнал IOP Conference Series: Materials Science and Engineering Number 3. November 2020. 032080. IOP Conference Series: Materials Science and Engineering, Vol 971..
53. Рожкова, О. А. Начальное моделирование в интегрированной среде «ИС-2» для задач проектирования цепных передач/, О. А. Рожкова // Прикладная математика и информатика: современные исследования в области естественных и технических наук : Материалы VI Международной научно-практической конференции (школы-семинара) молодых ученых, Тольятти, 23–25 апреля 2020 года.
54. Рожкова, О. А. Разработка начальной модели для проектных задач с использованием интегрированной среды "ИС-2" (на примере конических зубчатых передач) / О.А. Рожкова // Информационные технологии в моделировании и управлении: подходы, методы, решения : Материалы II Всероссийской научной конференции с международным участием. В 2 частях.
55. Новоселова, О. В. Аналитическая обработка динамической составляющей концептуальной модели / О. В. Новоселова, А. А. Ружейников // Информационные технологии в моделировании и управлении: подходы, методы, решения : Сборник научных статей II

Всероссийской научной конференции с международным участием. В 2 частях, Тольятти, 22–24 апреля 2019 года. – Тольятти: Издатель Качалин Александр Васильевич, 2019. – С. 511-516..

56. Новоселова, О. В. Аналитическая обработка инфологических моделей прикладных задач при проектировании информационно-активных систем, / О. В. Новоселова, А. А. Ружейников // Моделирование нелинейных процессов и систем : Материалы пятой международной конференции, Москва, 16–20 ноября 2020 года. – Москва: Общество с ограниченной ответственностью Издательство «Янус-К», 2021. – С. 255-258.,.
57. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г.: Б1.Б12 Рабочая программа Базы данных, рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол № 11-2018/2019 от 11.05.2019г, направление подготовки 09.03.01, профили "Модели, методы и программное обеспечение анализа проектных решений", "Программное обеспечение средств вычислительной техники и автоматизированных систем" - ФГБОУ ВО "МГТУ "СТАНКИН".
58. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., Б1.В.11 Рабочая программа Концептуальное моделирование проектных задач : рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол № 11 от 11.05.2019г, направление подготовки 09.03.01, профиль "Модели, методы и программное обеспечение анализа проектных решений" - ФГБОУ ВО "МГТУ "СТАНКИН".
59. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., Б1.В.12 Рабочая программа Проектирование автоматизированных систем : рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол № 11-2018/2019 от 11.05.2019г, направление подготовки 09.03.01, профиль "Модели, методы и программное обеспечение анализа проектных решений" - ФГБОУ ВО "МГТУ "СТАНКИН".
60. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., 1.В.ДВ.7.1 Рабочая программа Анализ деятельности проектно-конструкторских организаций рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол № 11-2018/2019 от 11.05.2019г, направление подготовки 09.03.01,

профиль "Модели, методы и программное обеспечение анализа проектных решений" - ФГБОУ ВО "МГТУ "СТАНКИН".

61. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., Б1.В.ДВ.2.1 Рабочая программа Технология разработки программного обеспечения : рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол № 11-2018/2019 от 11.05.2019г, профиль "Модели, методы и программное обеспечение анализа проектных решений" - ФГБОУ ВО "МГТУ "СТАНКИН".
62. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., Б1.В.ДВ.2.2 Рабочая программа Технологии и средства разработки ПО программного проекта : рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол №11-2018/2019 от 11.05.2019 г, направление подготовки 09.04.01, профиль "Управление программными продуктами и проектами" - ФГБОУ ВО "МГТУ "СТАНКИН".
63. Глинкин, М. О. Визуализации взаимосвязи статических и динамических структур в интегрированной среде «ИС-2» / М.О. Глинкин, О.В. Новоселова // Прикладная математика и информатика: современные исследования в области естественных и технических наук : Материалы VIII Международной научно-практической конференции (школы-семинара) молодых ученых, Тольятти, 22–24 апреля 2021 года. – Тольятти: Тольяттинский государственный университет, 2021..
64. Гаврилов, А.Г. Разработка средств управления функционированием интегрированной среды поддержки проектирования прикладных автоматизированных систем (ПАС) / А.Г. Гаврилов // Материалы студенческой научно-практической конференции «Автоматизация и информационные технологии» (АИТ-2013), М.: ФГБОУ ВПО МГТУ «СТАНКИН», 2013 – с. 142-146.
65. Гаврилов, А.Г. Разработка средств автоматизированного развертывания интегрированной среды проектирования автоматизированных систем / А.Г. Гаврилов, В.А. Коленкин // Журнал «Уральский научный вестник» – Уральск: ТОО Уралнаучкнига, №8, volume 1, 2017. с. 27 – 29.
66. Гаврилов, А.Г. Разработка средств автоматизации процесса проектирования прикладных автоматизированных систем (ПАС). / А.Г. Гаврилов // Труды XVII научной конференции «Математическое моделирование и информатика». / Под ред. Д.Ю. Рязанова. – М.: ИЦ ФГБОУ ВО МГТУ «СТАНКИН», 2015. – 280 с. в 2 т.: том 1 – с. 65-68..

67. Гаврилов, А.Г. Особенности разработки средств управления функционированием интегрированной среды для проектирования прикладных автоматизированных систем / А.Г. Гаврилов // Materials of the X International scientific and practical conference, «Fundamental and applied science», - 2014. Volume 16. Mathematics. Physics. Modern information technologies. Sheffield. Science and education LTD – с. 90 – 92.
68. Гаврилов, А.Г. Особенности разработки интегрированной среды для проектирования прикладных автоматизированных систем / А. Г. Гаврилов // Высокие технологии, исследования, финансы, промышленность: сборник статей Восемнадцатой международной научно-практической конференции “Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности и экономике“, Vol. СПб.: Изд-во Политехн. ун-та, 2014. – 200с. – с.25-28.
69. Гаврилов, А.Г. Особенности визуализации элементов для представления процессов в графическом редакторе интегрированной среды / А.Г. Гаврилов, О.В. Новоселова // Машиностроение и машинознание (26), година XI, книга 1, 2016 издательство на технический университет – Варна. с. 113-116.
70. Гаврилов, А.Г. Моделирование интегрированной среды поддержки создания прикладных автоматизированных систем / А.Г. Гаврилов, О.В. Новоселова, Г.Д Волкова // Журнал «Известия высших учебных заведений. Поволжский регион», Пенза, 2014. с.81-91.
71. Гаврилов, А.Г. Метод моделирования визуальной интегрированной среды поддержки создания прикладных автоматизированных систем. // А.Г.Гаврилов, Волкова Г.Д. // Журнал «Вестник МГТУ «СТАНКИН» – М.: ФГБОУ ВО «МГТУ «СТАНКИН», №4 (39), 2016. – 136с.: ил. – с.85-89.
72. Гаврилов, А.Г. Особенности разработки модулей для интегрированной среды поддержки процесса создания прикладных автоматизированных систем / А.Г. Гаврилов // Материалы студенческой научно практической конференции «Автоматизация и информационные технологии (АИТ-2015)». Сборник тезисов. – М.: МГТУ «Станкин», 2015. – 321 с., сс.194-197.
73. Гаврилов, А.Г. Инструментальная среда поддержки процессов создания систем автоматизированного проектирования. / А.Г. Гаврилов ,Г.Д. Волкова, О.В. Новоселова // Научно-технический журнал «Информационные технологии в проектировании и производстве», ISSN/Код НЭБ 2073-2597 / 20732597, № 2, 2018г – с.30-36.
74. Гаврилов, А. Г. Разработка визуального графического редактора функциональных и динамических составляющих моделей автоматизированных систем на всех этапах их создания / А.Г. Гаврилов ,О.В. Новоселова , Г.Д. Волкова // Журнал «Вестник МГТУ

«СТАНКИН» – М.: ФГБОУ ВО «МГТУ «СТАНКИН», №2 (41), 2017. – 136с.: ил. – с. 83-88.

75. Гаврилов, А.Г. Формирование требований к автоматизированной поддержке концептуального моделирования в интегрированной среде / А.Г. Гаврилов, О.В. Новоселова, Г.Д. Волкова // Материалы VI всероссийской научно-практической конференции «Машиностроение – традиции и инновации (МТИ-2013)», М.: МГТУ СТАНКИН, 2013 – с. 119 – 122.
76. Гаврилов, А.Г. Требования к интегрированной среде поддержки процесса создания прикладных автоматизированных систем / А.Г. Гаврилов, О.В. Новоселова, Г.Д. Волкова // Высокие технологии, фундаментальные исследования, финансы: сборник статей Шестнадцатой международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности и экономике», No. СПб.: Изд-во Политехн. ун-та, 2013 – с. 16-19.
77. Гаврилов, А.Г. Метод управления развитием визуальной интегрированной среды проектирования автоматизированных систем / А.Г. Гаврилов // «News of Science and Education» – Sheffield – SCIENCE AND EDUCATION LTD, №8, volume 2, 2017 – с. 33 – 36.
78. Щукин, М.В. Разработка метода и средств поддержки аналитической обработки, визуализации и документирования концептуальных представлений при проектировании САПР машиностроительного назначения, дис.. канд-та. техн. наук: 05.13.12 / Щукин Максим Владимирович. – М., 2003. - 346 с..
79. Волкова, Г.Д. Методология автоматизации проектноконструкторской деятельности в машиностроении. : Учеб. пос. / Г.Д. Волкова / - М.: МГТУ "СТАНКИН", 2000 - 81с.
80. Новоселова О.В. Разработка метода и средств поддержки автоматического отоб-ражения концептуальных представлений в инфологические при проектировании САПР машиностроительного назначения., дис.. канд-та техн. наук. / О.В. Новоселова – М.: 1997, 264с..
81. Сирота И.М. Разработка методов и средств поддержки визуального концептуаль-ного моделирования проектно-конструкторских задач при создании САПР машино-строительного назначения., дис.. канд-та техн. наук. / И.М. Сирота – М.: 1999, 198 с..
82. Курышев С.М. Разработка методов и средств формирования и моделирования представлений проектно-конструкторских задач на этапе предпроектного обследования организации при создании САПР машиностроительного назначения., дис.. канд-та техн.

наук. 05.13.12 / С.М. Курышев – М.: 1999, 180 с..

83. Соловьев, Н.А. Системы автоматизации разработки программного обеспечения: учебное пособие / Н.А. Соловьев, Е.Н. Чернопрудова, Оренбургский гос. ун-т. – Оренбург: ОГУ, 2012. – 191 с..
84. Ковалевская Е.В. МЕТОДЫ ПРОГРАММИРОВАНИЯ: учебно-методический комплекс / Е.В. Ковалевская, Н.В. Комлева. – М. : Изд. центр ЕАОИ, 2011. – 320 с..
85. Свидетельство о государственной регистрации программы для ЭВМ № 2017618975 Российская Федерация. Программный комплекс ИС-2-У : № 2017616041 : заявл. 23.06.2017 : опубл. 14.08.2017 / А. Г. Гаврилов.
86. Свидетельство о государственной регистрации программы для ЭВМ № 2021616009 Российская Федерация. Модуль интеграции динамической составляющей концептуальной модели : № 2021615079 : заявл. 09.04.2021 : опубл. 15.04.2021 /. А. С. Сидоров, А. Г. Гаврилов, О. В. Новоселова, Г. Д. Волкова.
87. Свидетельство о государственной регистрации программы для ЭВМ № 2021616013 Российская Федерация. Модуль реструктуризации динамической составляющей концептуальной модели : № 2021615065 : заявл. 09.04.2021 : опубл. 15.04.2021 /. А. А. Ружейников, А. Г. Гаврилов, О. В. Новоселова, Г. Д. Волкова.
88. Свидетельство о государственной регистрации программы для ЭВМ № 2021616067 Российская Федерация. Модуль интеграции функциональной составляющей инфологической модели : № 2021615061 : заявл. 09.04.2021 : опубл. 15.04.2021. А. С. Сидоров, А. Г. Гаврилов, О. В. Новоселова, Г. Д. Волкова.
89. Свидетельство о государственной регистрации программы для ЭВМ № 2021618780 Российская Федерация. Модуль реструктуризации функциональной составляющей инфологической модели : № 2021615081: заявл. 09.04.2021 : опубл. 01.06.2021 /. А. С. Сидоров, А. Г. Гаврилов, О. В. Новоселова, Г. Д. Волкова.
90. Семячкова, Е. Г. Разработка методов и средств поддержки процессаинфологического моделирования при создании САПР машиностроительного назначения : специальность 05.13.12 "Системы автоматизации проектирования (по отраслям)" , : автореферат диссертации на соискание ученой степени кандидата технических наук / Семячкова Елена Геннадьевна. – Москва, 1996. – 27 с..
91. Сидоров, А. С. Интеграция описаний процессов, представленных в виде спецификаций, на этапе концептуального моделирования / А. С. Сидоров // Прикладная математика и информатика:, современные исследования в области естественных и технических наук :

Материалы VI Международной научно-практической конференции (школы-семинара) молодых ученых, Тольятти, 23–25 апреля 2020 года. – Тольятти: Тольяттинский государственный университет, 2020. – С. 100-104..

92. Метод отображения семантических статических конструкций в синтаксические при проектировании информационно-активных систем / Г. Д. Волкова, О. В. Новоселова, Е. Г. Семячкова, Т. Б. Тюрбеева // Моделирование нелинейных процессов и систем : Сборник тезисов четвёртой международной конференции, Москва, 15–17 октября 2019 года / Московский Государственный Технологический Университет «СТАНКИН». – Москва: Общество с ограниченной ответственностью Издательство «Янус-К», 2019. – С. 66-68..
93. Еремин, Г. В. Процесс интеграции информационных описаний процесса решения задачи в виде спецификаций / Г. В. Еремин, О. В. Новоселова, О. В. Новоселова // Прикладная математика и информатика: современные исследования в области естественных и технических наук : Сборник научных статей IV научно-практической международной конференции (школы-семинара) молодых ученых: в двух частях, Тольятти, 23–25 апреля 2018 года. – Тольятти: Издатель Качалин Александр Васильевич, 2018. – С. 39-43..
94. Концептуальное моделирование процессов конструирования сложных машиностроительных изделий / Т. Б. Тюрбеева, Г. Д. Волкова, Е. Г. Семячкова [и др.] // Автоматизация и управление в машиностроении. – 2018. – № 2(31). – С. 11-22..
95. Об утверждении тем и руководителей выпускных квалификационных работ. Приказ №435/1 от 28.09.2016 – ФГБОУ ВО «МГТУ «СТАНКИН».
96. Об утверждении тем и руководителей выпускных квалификационных работ. Приказ №555/1 от 03.12.2015 – ФГБОУ ВО «МГТУ «СТАНКИН».
97. О назначении тематик выпускных квалификационных работ, профилей и руководителей. Приказ №599/1 от 30.10.2020 – ФГБОУ ВО «МГТУ «СТАНКИН».
98. Об утверждении тем и руководителей выпускных квалификационных работ. Приказ №696/1 от 19.10.2018 – ФГБОУ ВО «МГТУ «СТАНКИН».
99. Об утверждении тем и научных руководителей магистерских диссертаций Приказ №652/1 от 07.11.2017 – ФГБОУ ВО «МГТУ «СТАНКИН».
100. Об утверждении тем и научных руководителей магистерских диссертаций Приказ №651/1 от 07.11.2017 – ФГБОУ ВО «МГТУ «СТАНКИН».
101. Об утверждении тем и научных руководителей магистерских диссертаций Приказ №787/1 от 24.10.2019 – ФГБОУ ВО «МГТУ «СТАНКИН».
102. Об утверждении тем и научных руководителей магистерских диссертаций Приказ №786/1

от 24.10.2019 – ФГБОУ ВО «МГТУ «СТАНКИН».

103. Пай, С. Моделирование проблемно-ориентированных знаний, зафиксированных в научных публикациях и технической документации / С. Пай, Г. Д. Волкова, Т. Б. Тюрбеева // World science: problems and innovations : сборник статей XXXVII Международной научно-практической конференции, енза, 30 ноября 2019 года. – Пенза: "Наука и Просвещение" (ИП Гуляев Г.Ю.), 2019. – С. 38-41..

104. Рабочая программа и фонд оценочных средств, входящих в состав рабочих программ : утверждены проректором по образовательной деятельности от 23.05.2019 г., Б1.В.ДВ.5.2 Рабочая программа Методы анализа и проектирования предметных задач : рекомендована заседанием кафедры Информационных технологий и вычислительных систем протокол №11-2018/2019 от 11.05.2019 г, направление подготовки 09.04.01, профиль "Управление программными продуктами и проектами" - ФГБОУ ВО "МГТУ "СТАНКИН".

Приложение А

Структура моделей АС на ранних этапах создания и их унифицированные элементы

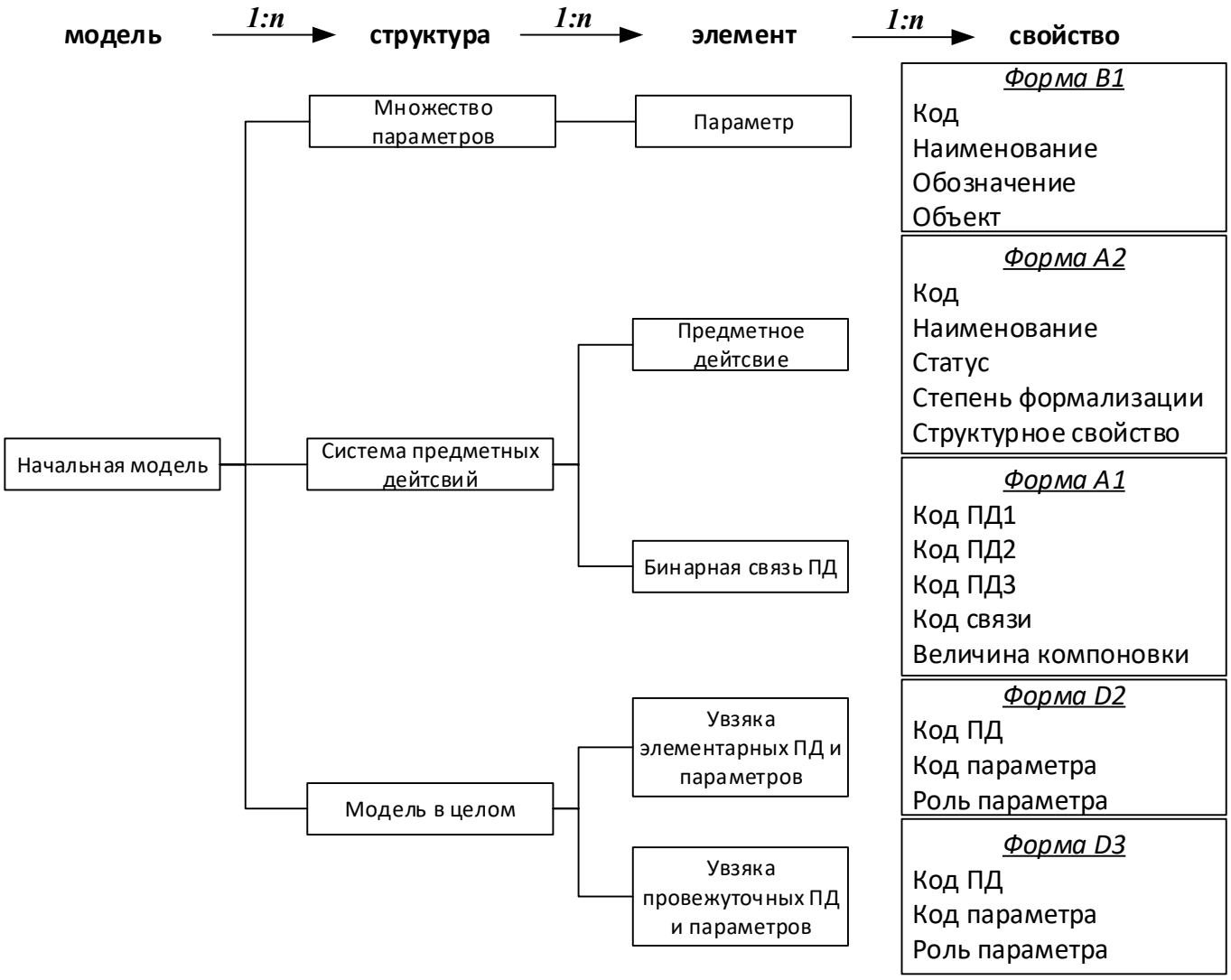


Рис. А1. Структура и свойства элементов начальной модели

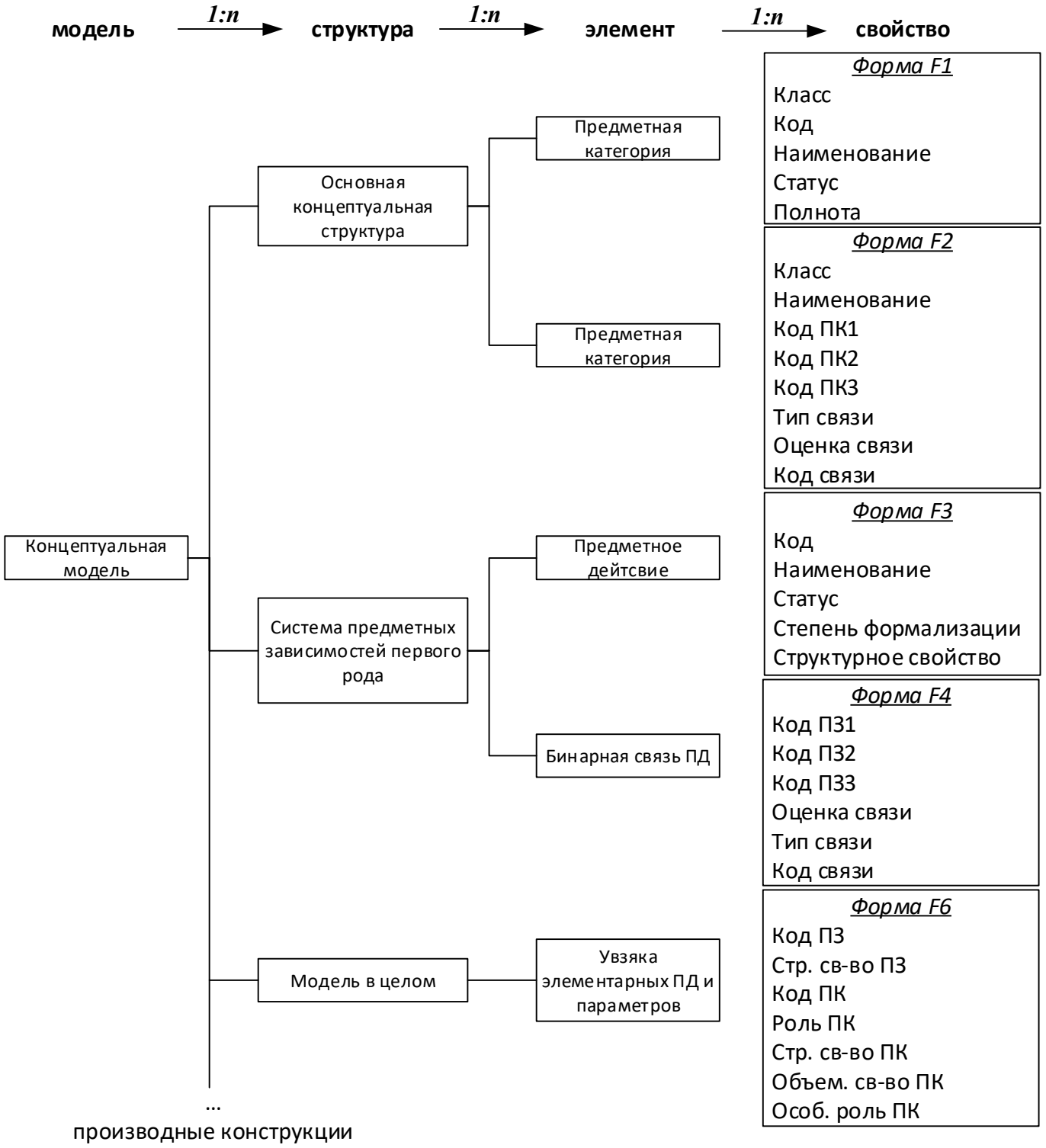


Рис. А2 Структура и свойства элементов концептуальной модели

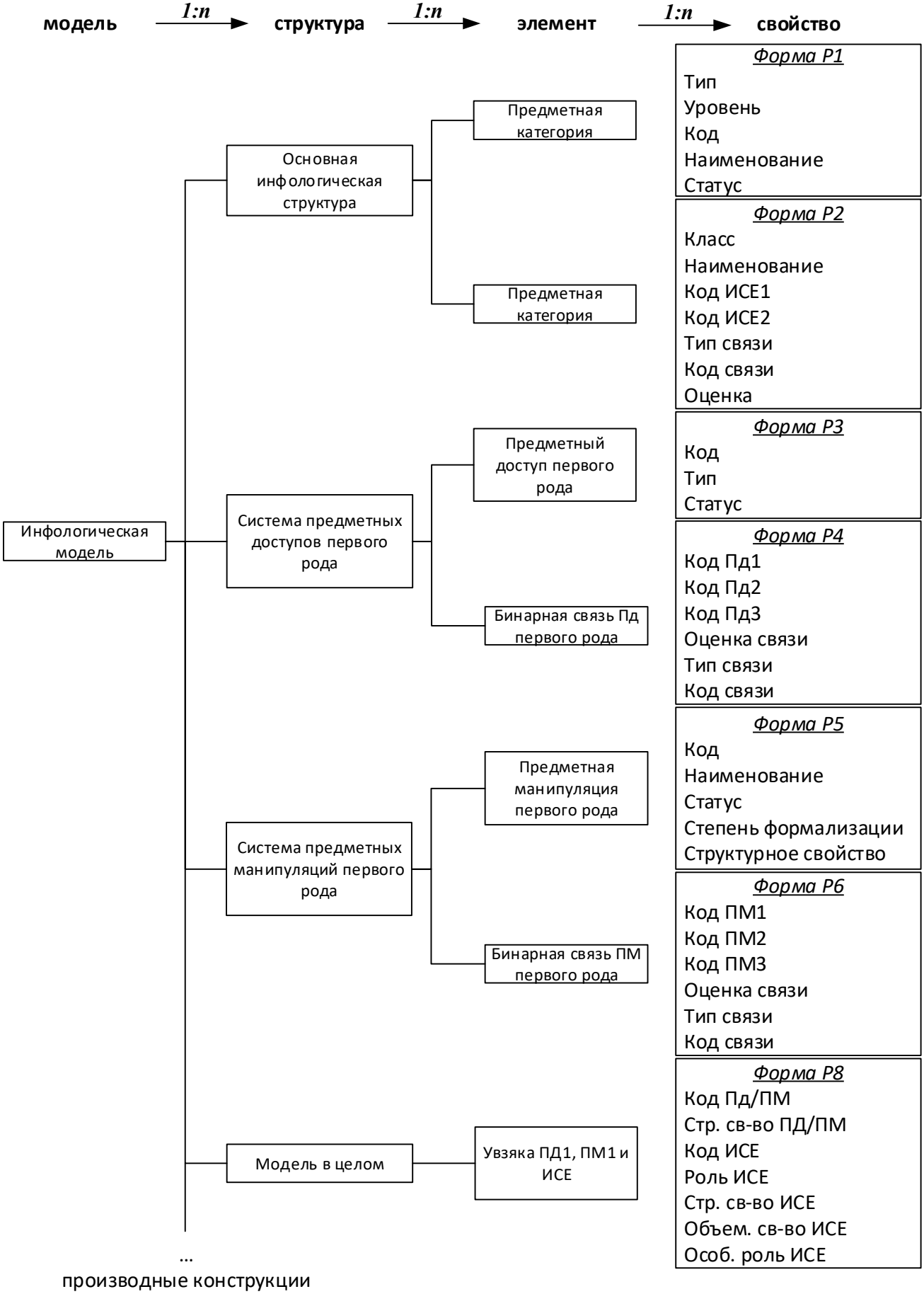


Рис. А3 Структура и свойства элементов инфологической модели

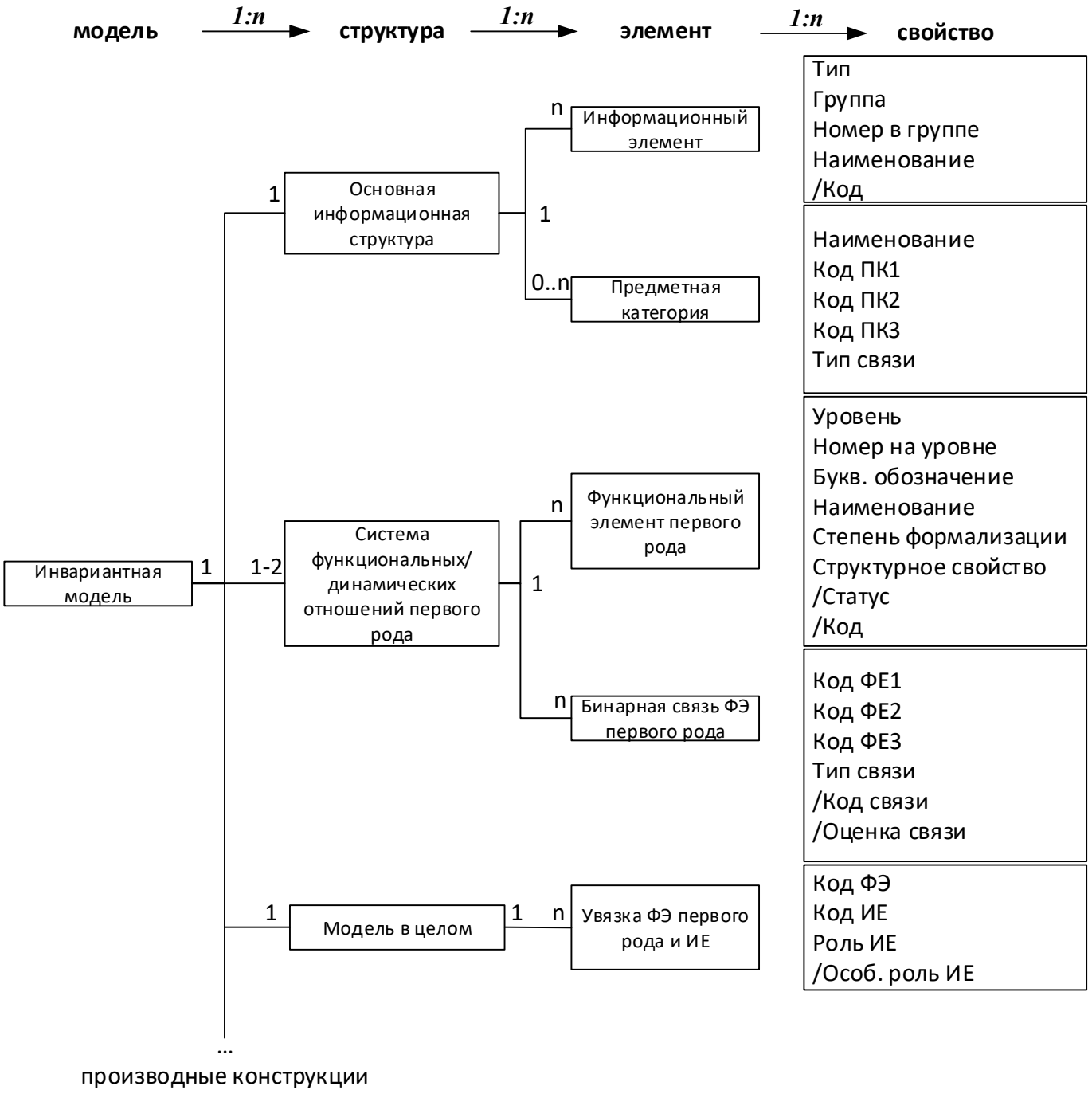


Рис. А4 Структура и свойства унифицированной модели

Приложение Б

Методика управления развитием ВИС

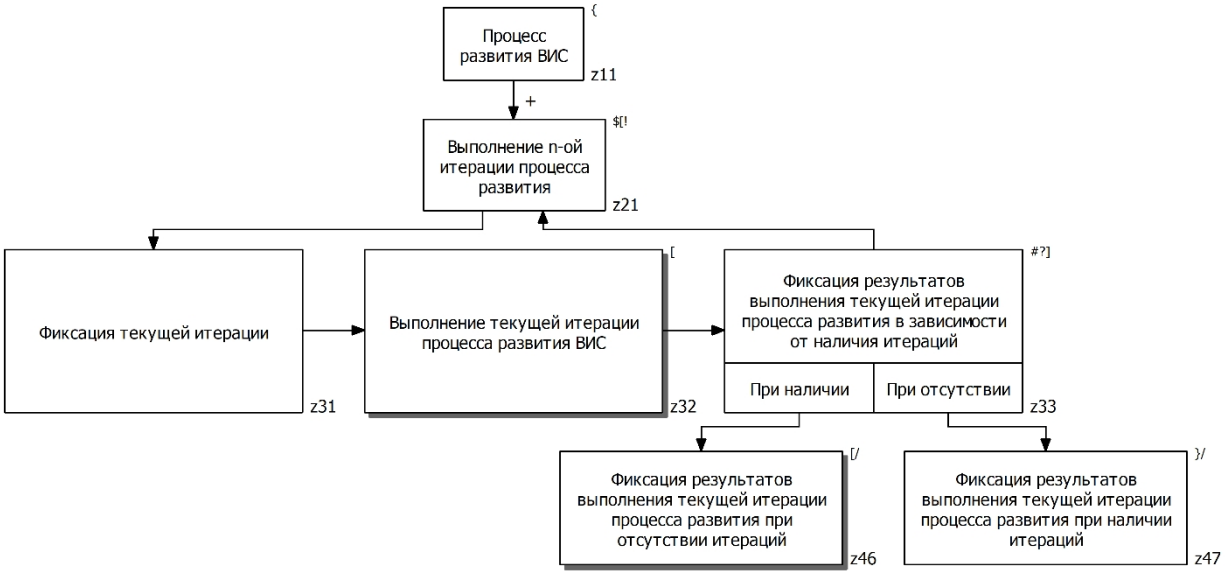


Рис.Б.1 Декомпозиция блока z11



Рис. Б.2 Декомпозиция блока z32

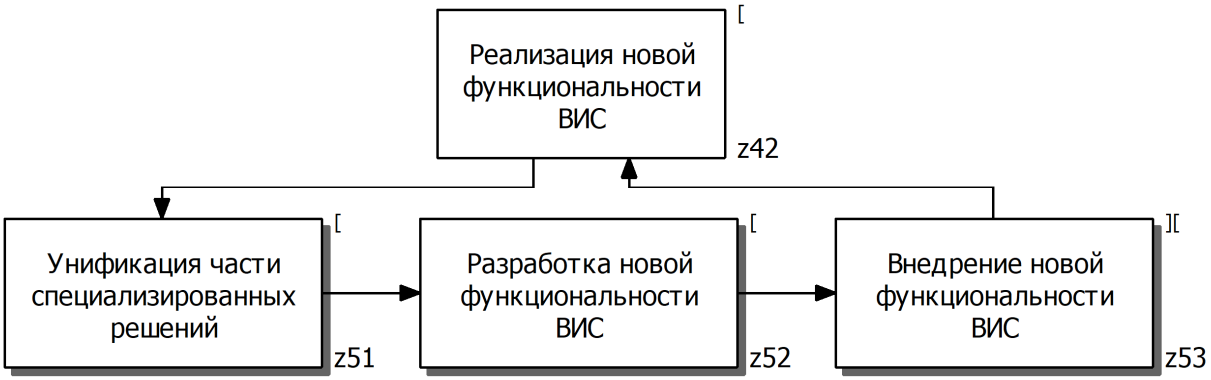


Рис. Б.3 Декомпозиция блока z42

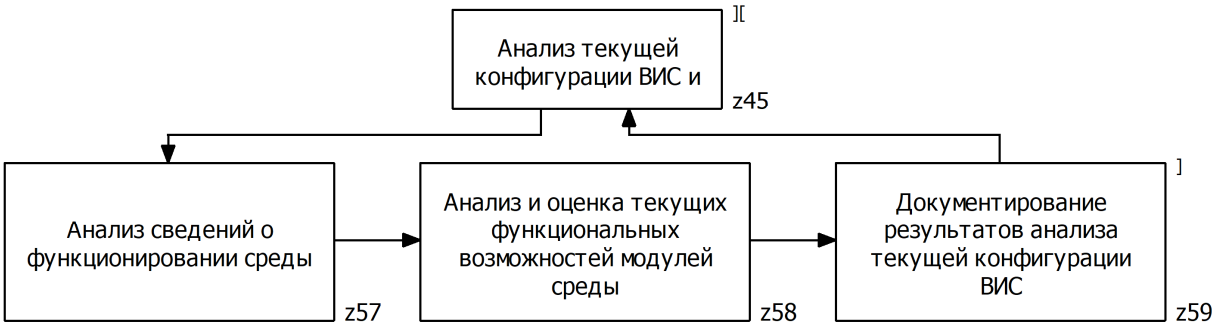


Рис. Б.4 Декомпозиция блока z45

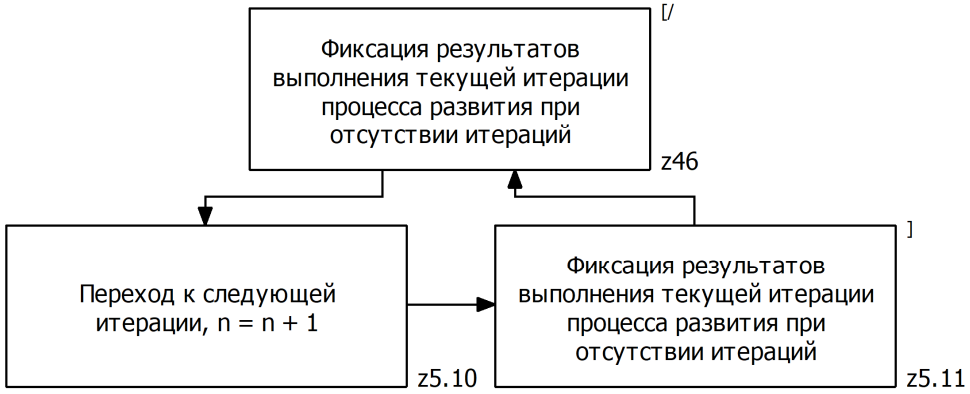


Рис. Б.5 Декомпозиция блока z46



Рис. Б.6 Декомпозиция блока z51

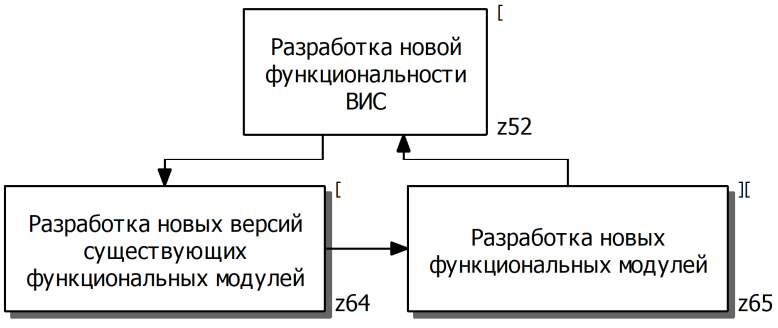


Рис. Б.7 Декомпозиция блока z52

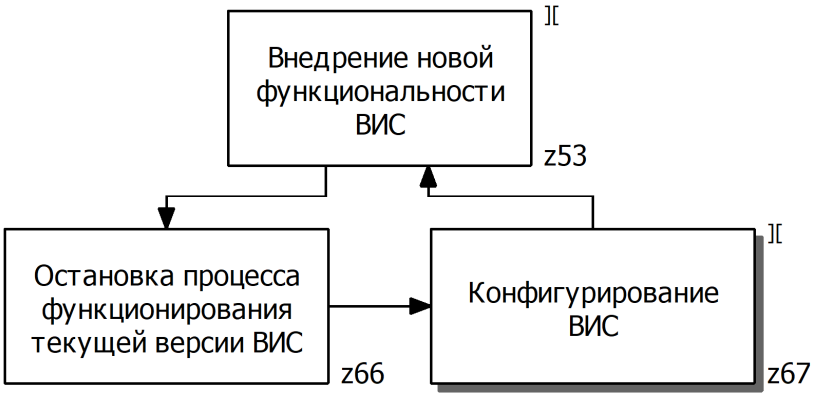


Рис. Б.8 Декомпозиция блока z53

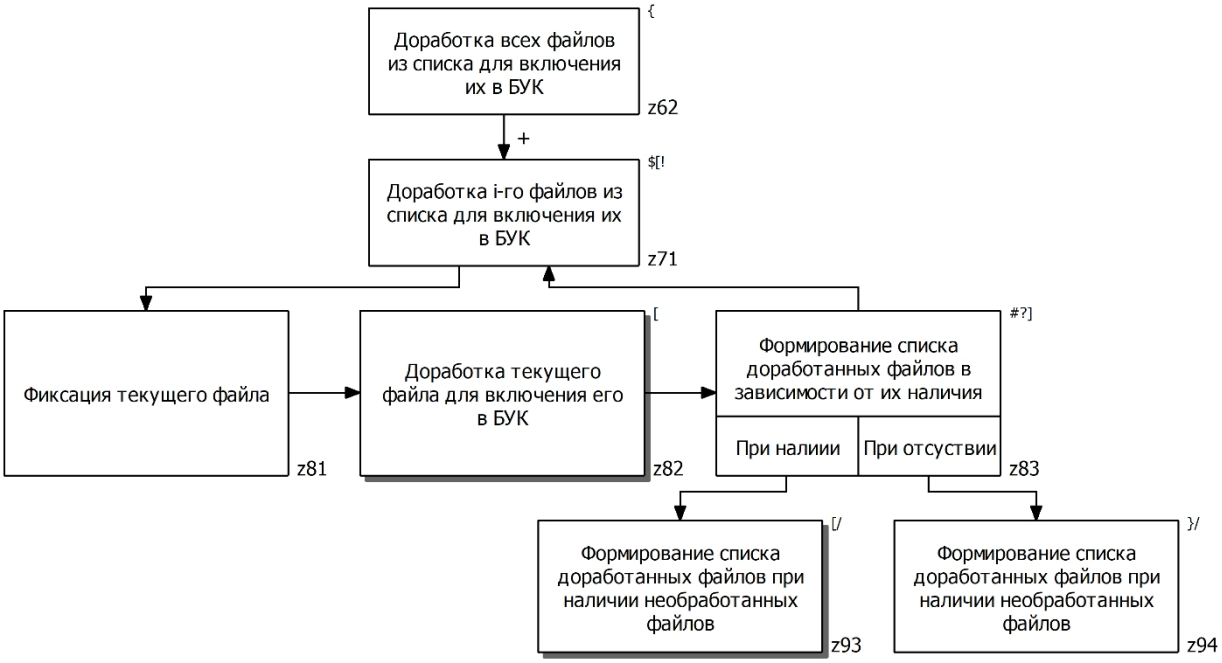


Рис. Б.9 Декомпозиция блока z62

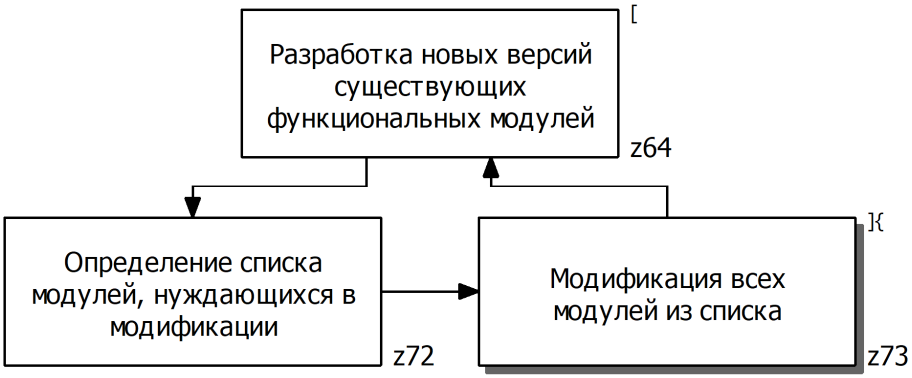


Рис.10 Декомпозиция блока z64

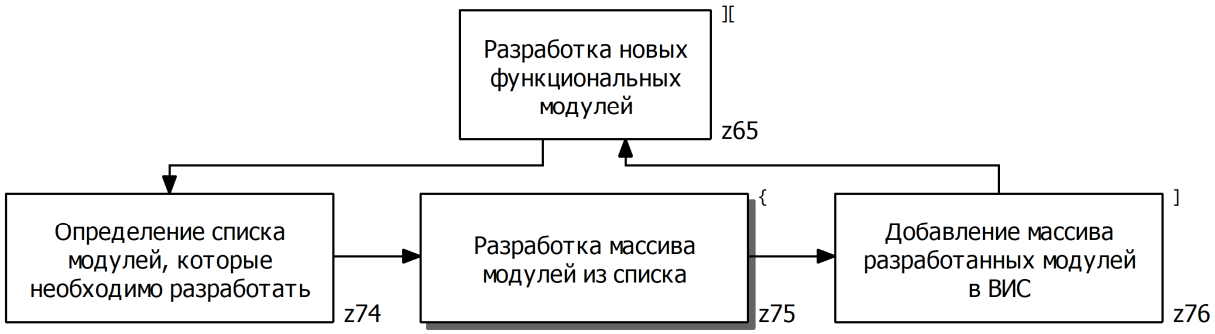


Рис. Б.11 Декомпозиция блока z65

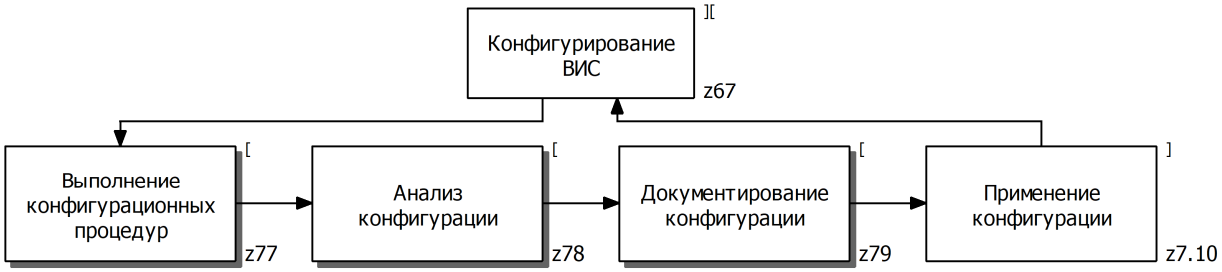


Рис. Б.12 Декомпозиция блока z67

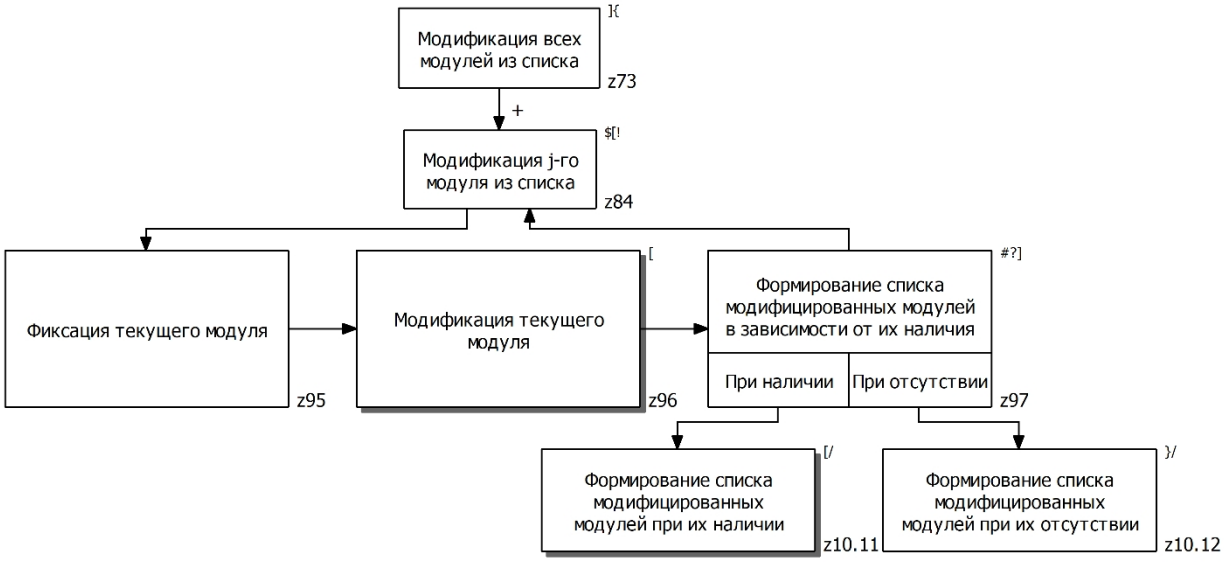


Рис. Б.13 Декомпозиция блока z73



Рис. Б.14 Декомпозиция блока z75

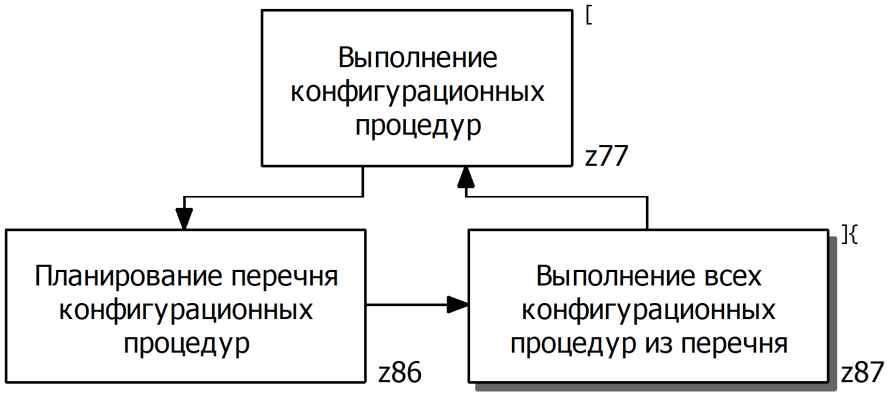


Рис. Б.15 Декомпозиция блока z77

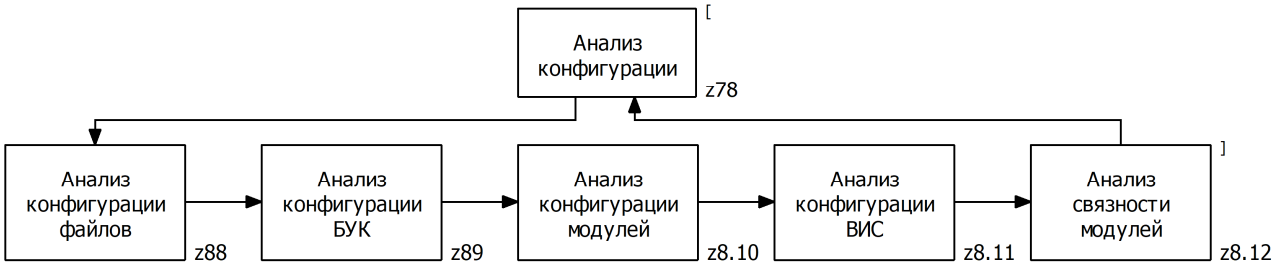


Рис. Б.16 Декомпозиция блока z78

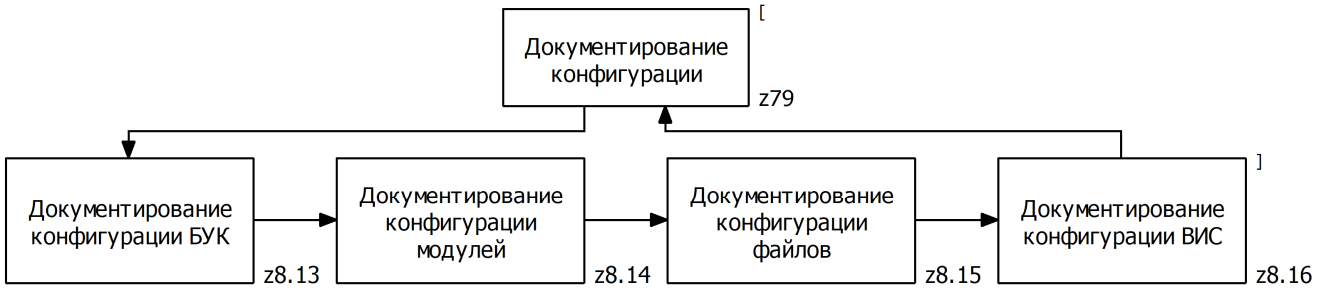


Рис. Б.17 Декомпозиция блока z79

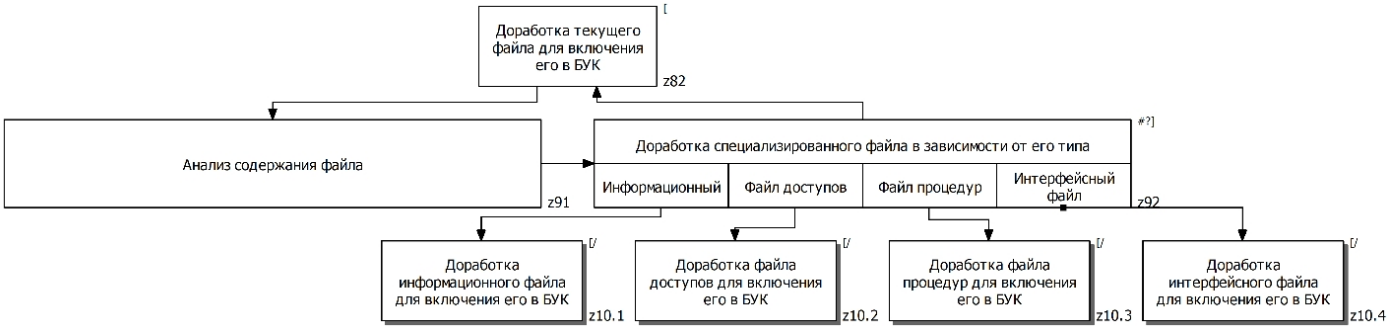


Рис. Б.18 Декомпозиция блока z82

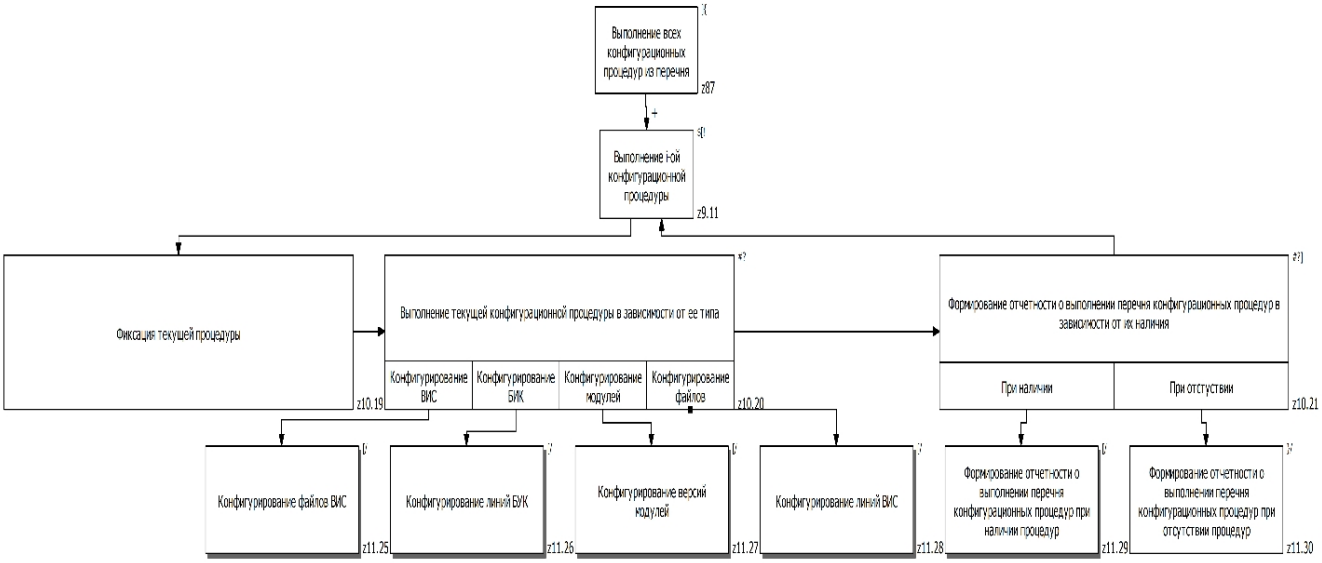


Рис. Б.19 Декомпозиция блока z87

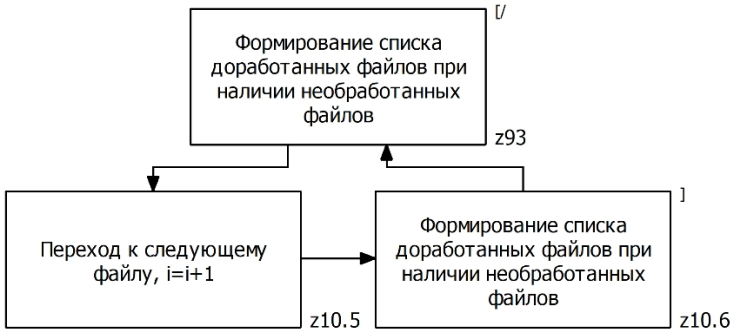


Рис. Б.20 Декомпозиция блока z93

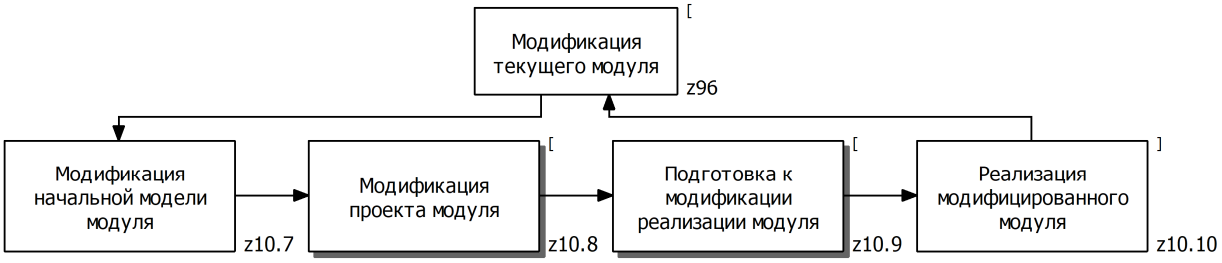


Рис. Б.21 Декомпозиция блока z96

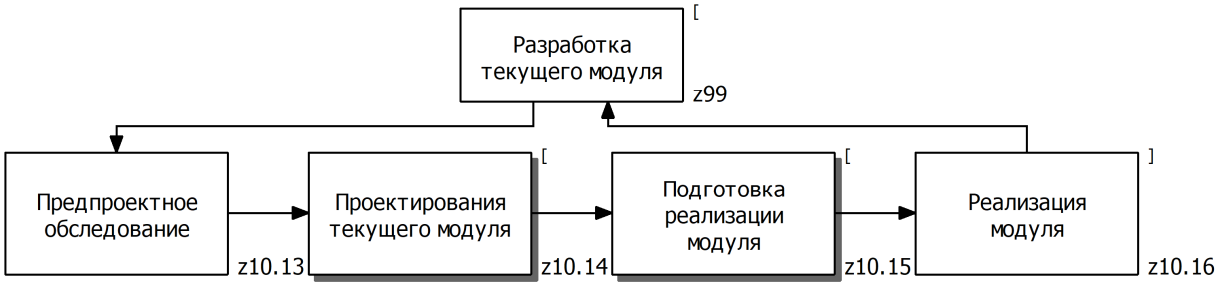


Рис. Б.22 Декомпозиция блока z99

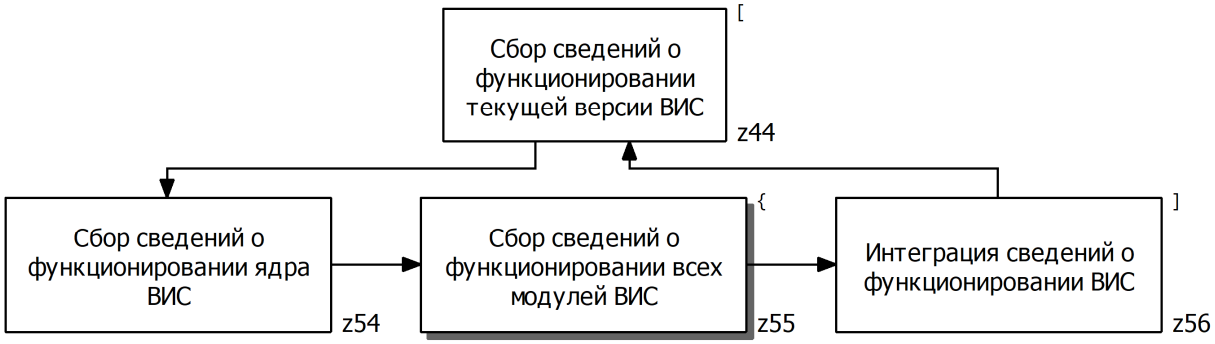


Рис. Б.23 Декомпозиция блока z44

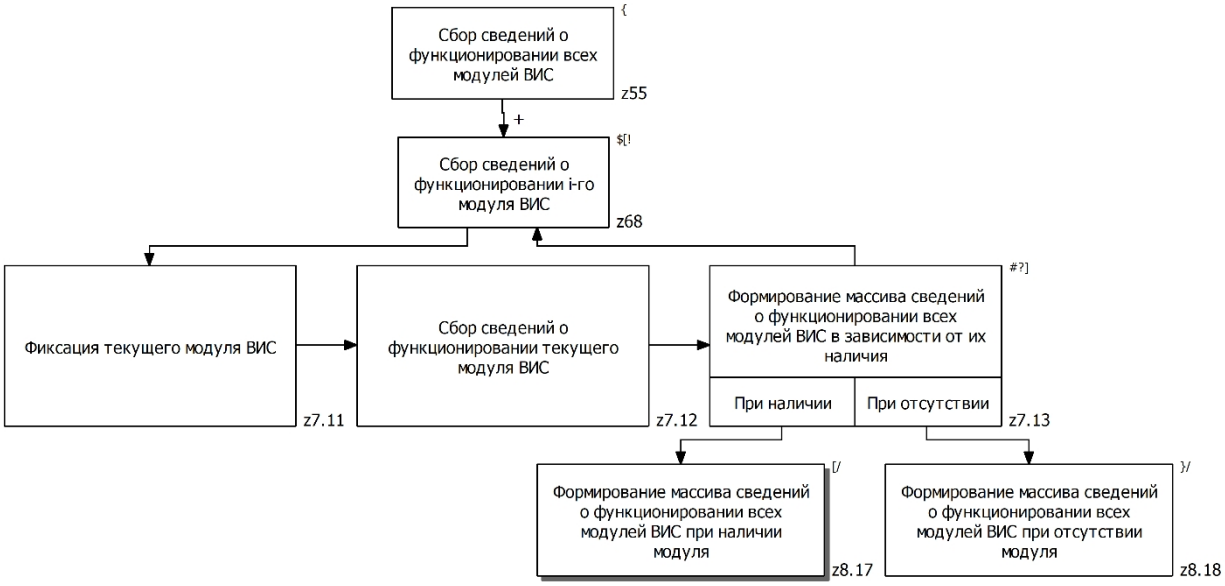


Рис. Б.24 Декомпозиция блока z55

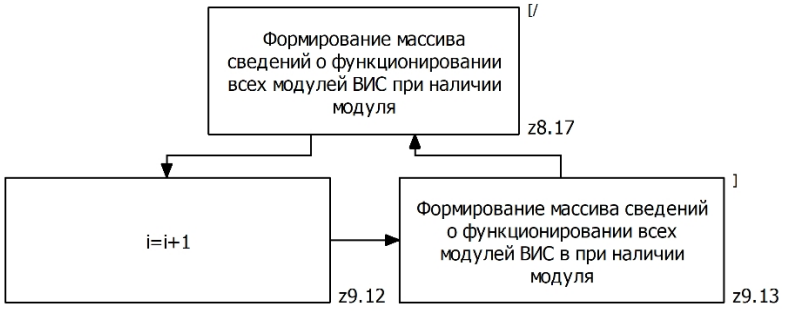


Рис. Б.25 Декомпозиция блока z8.17

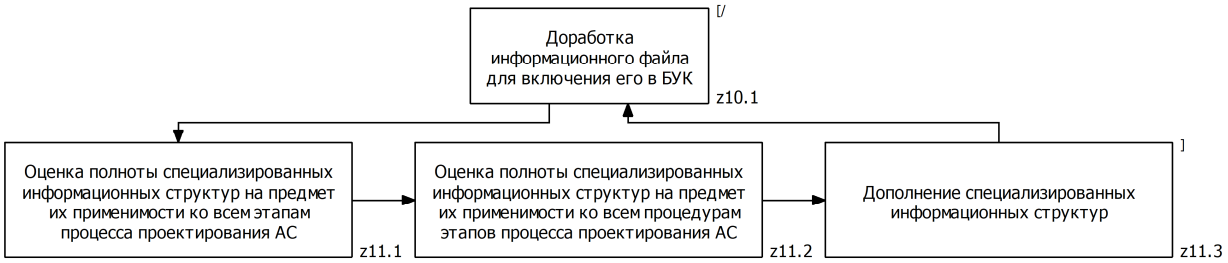


Рис. Б.26 Декомпозиция блока z10.1

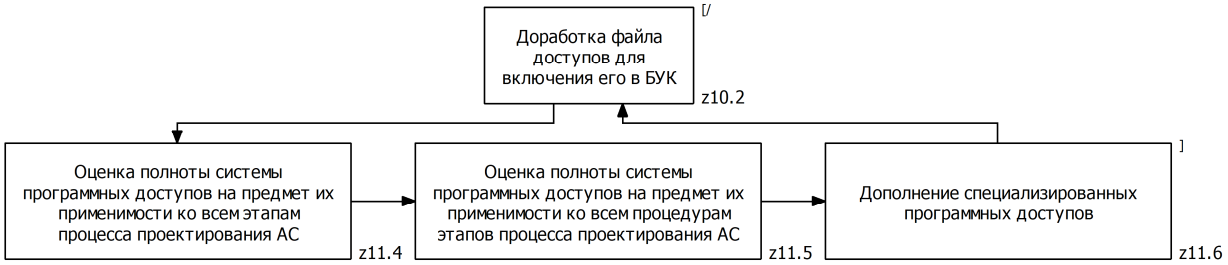


Рис. Б.27 Декомпозиция блока z10.2

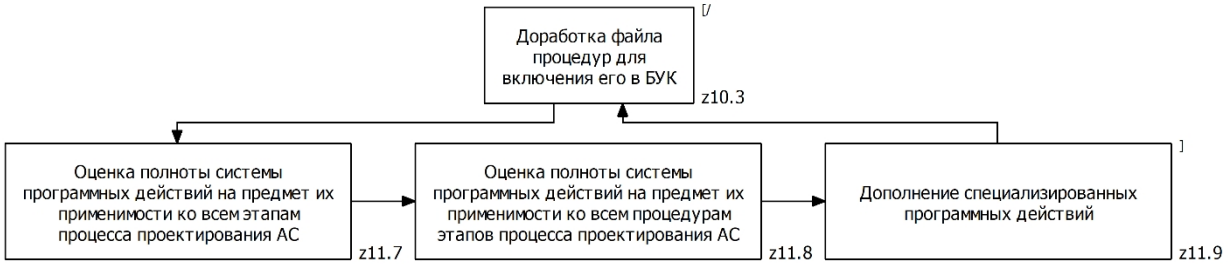


Рис. Б.28 Декомпозиция блока z10.3

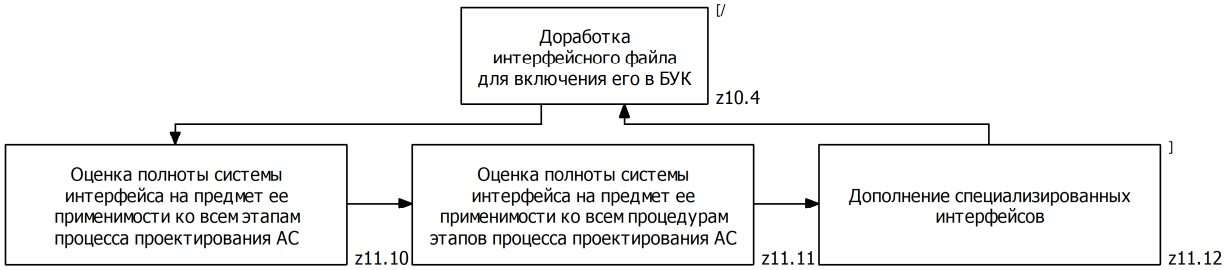


Рис. Б.29 Декомпозиция блока z10.4

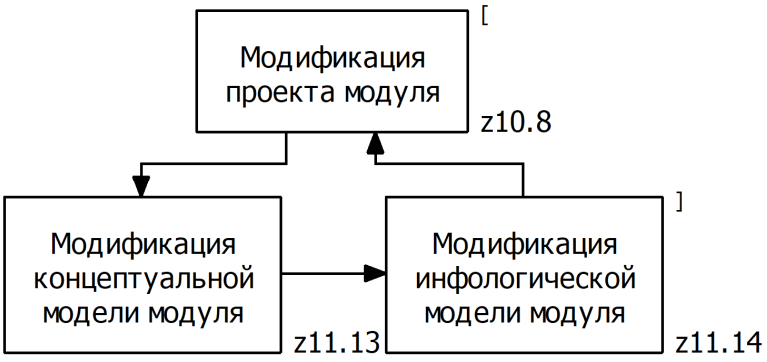


Рис. Б.30 Декомпозиция блока z10.8

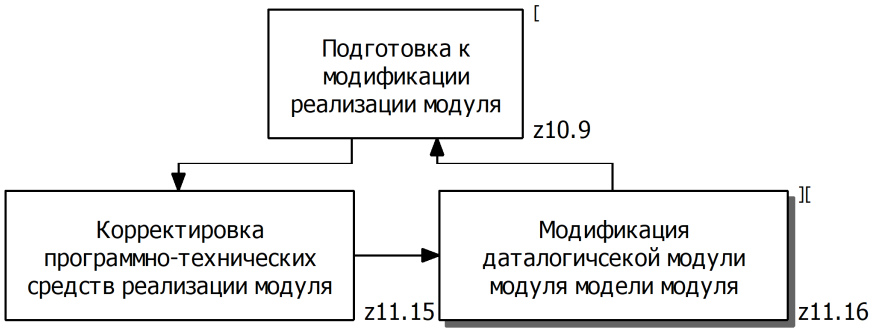


Рис. Б.31 Декомпозиция блока z10.9

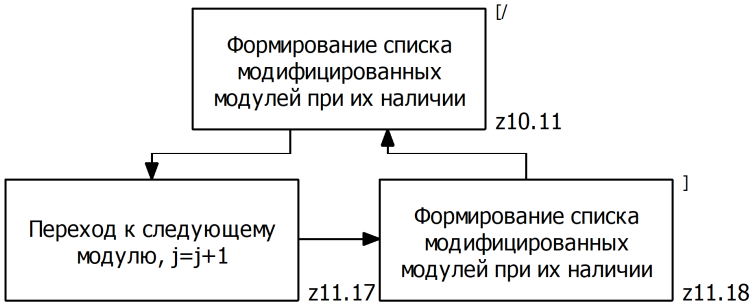


Рис. Б.32 Декомпозиция блока z10.11

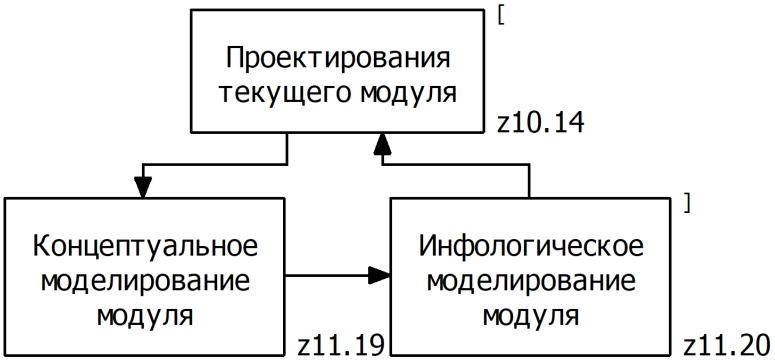


Рис. Б.33 Декомпозиция блока z10.14

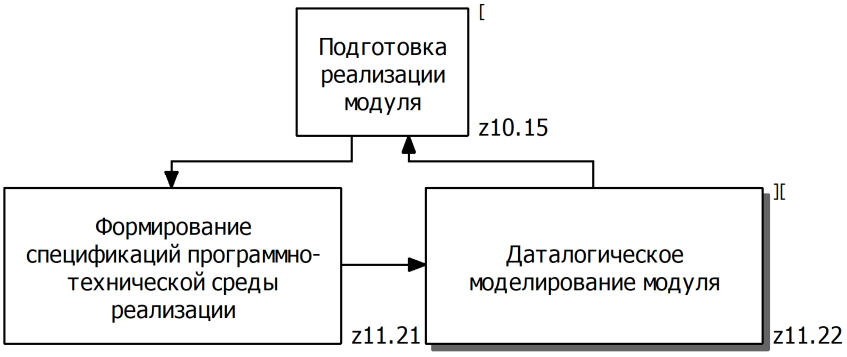


Рис. Б.34 Декомпозиция блока z10.15

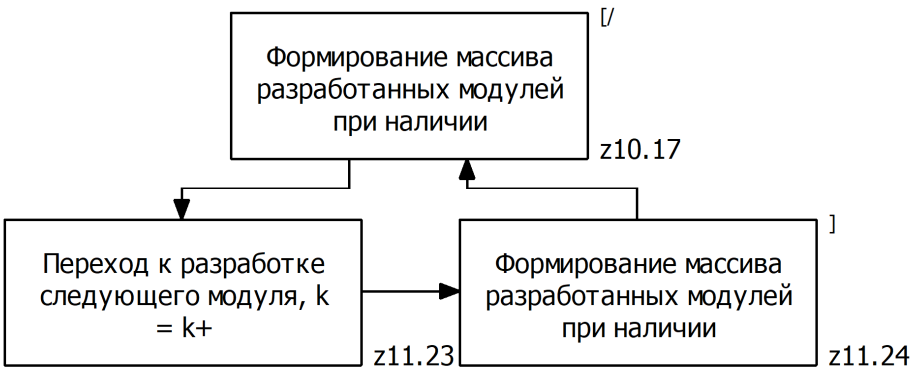


Рис. Б.35 Декомпозиция блока z10.17

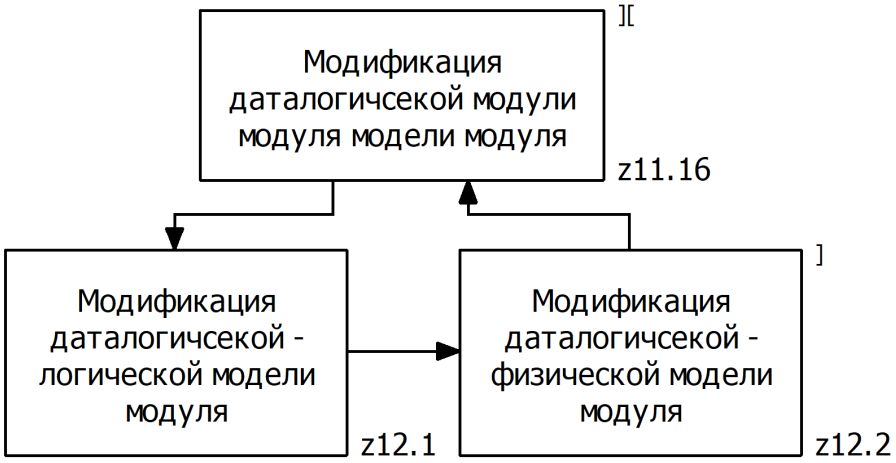


Рис. Б.36 Декомпозиция блока z11.16

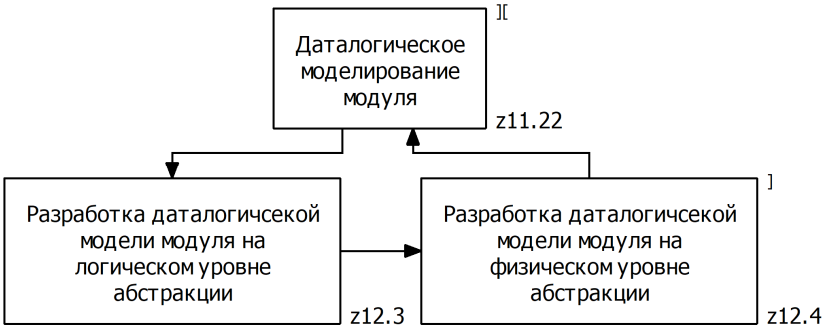


Рис. Б.37 Декомпозиция блока z11.22

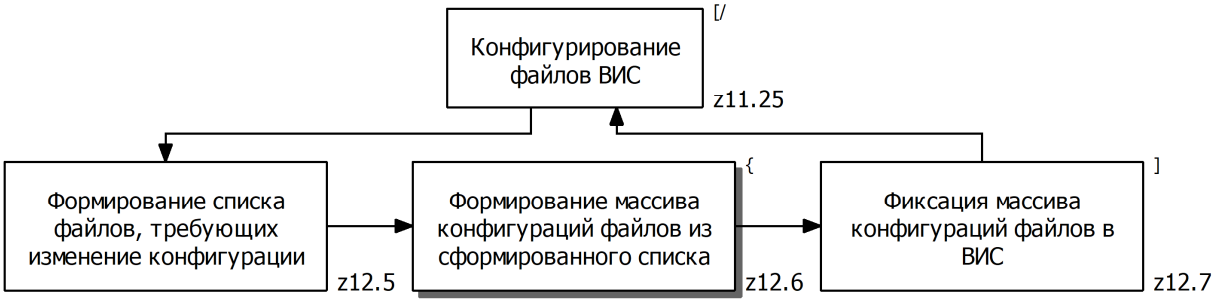


Рис. Б.38 Декомпозиция блока z11.25

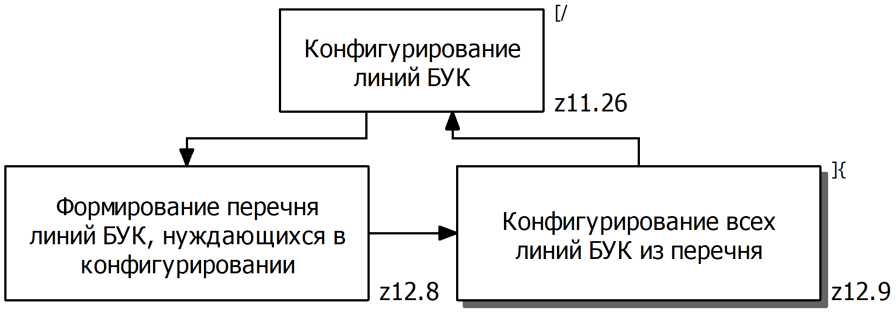


Рис. Б.39 Декомпозиция блока z11.26

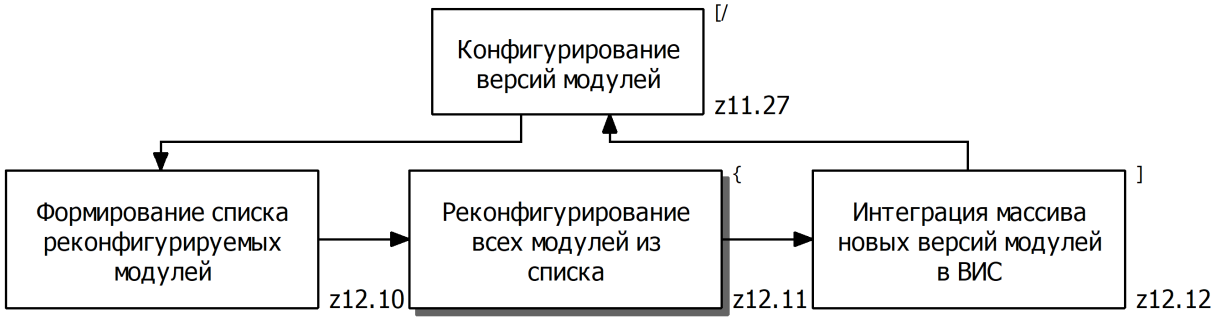


Рис. Б.40 Декомпозиция блока z11.27

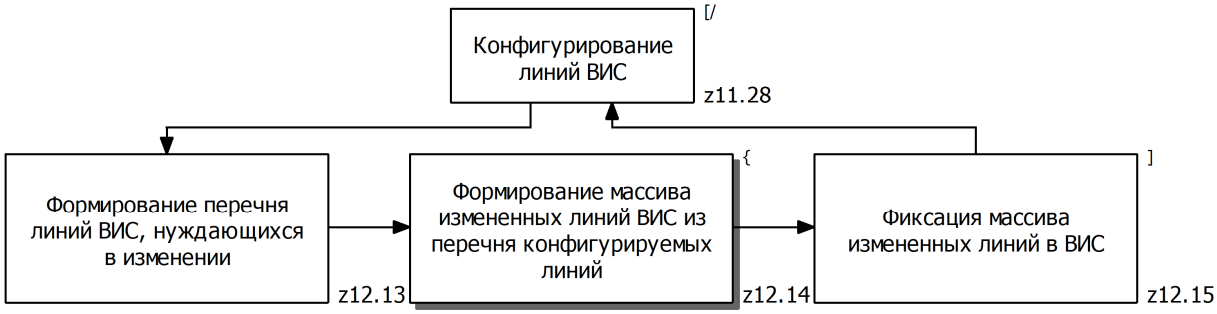


Рис. Б.41 Декомпозиция блока z11.28

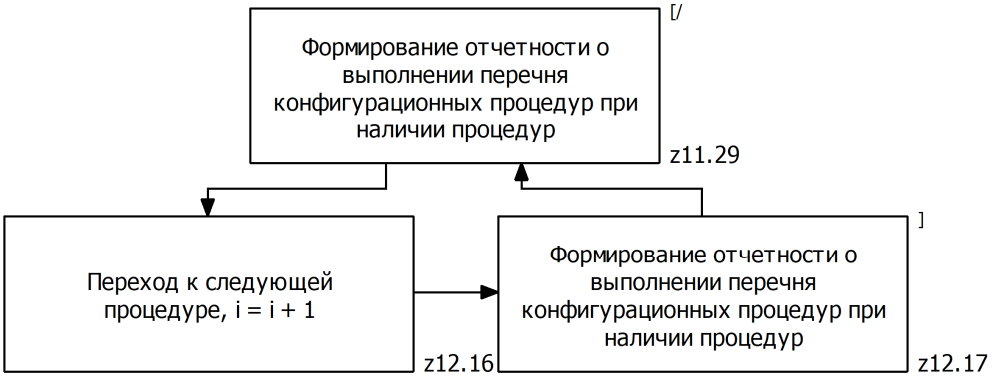


Рис. Б.42 Декомпозиция блока z11.29

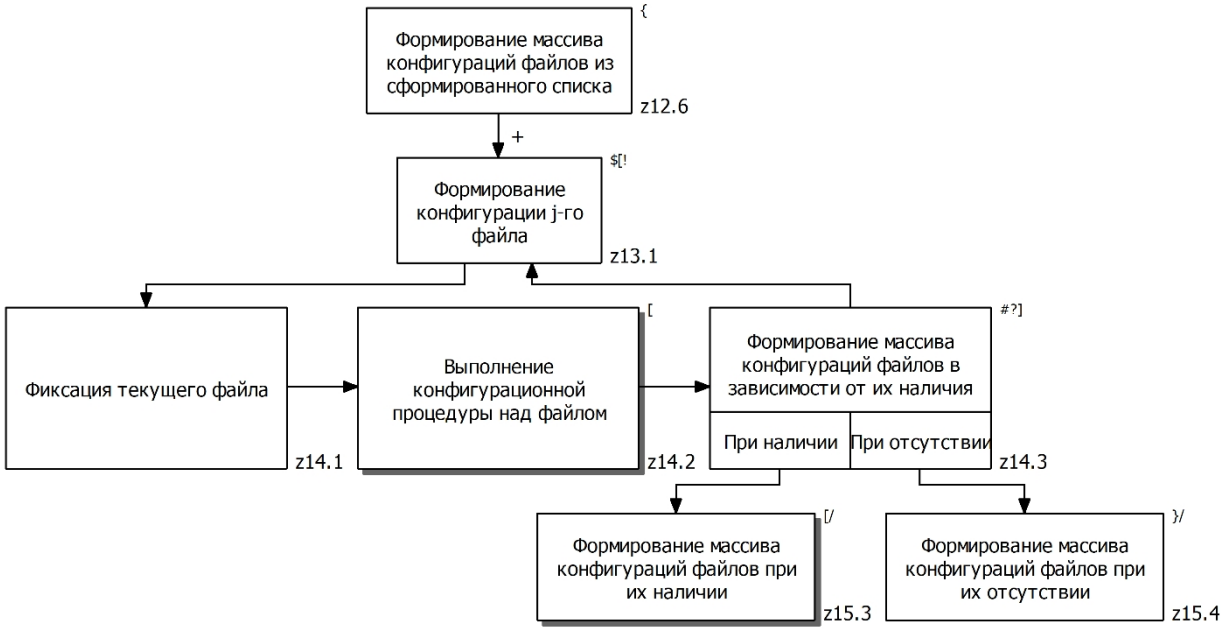


Рис. Б.43 Декомпозиция блока z12.6

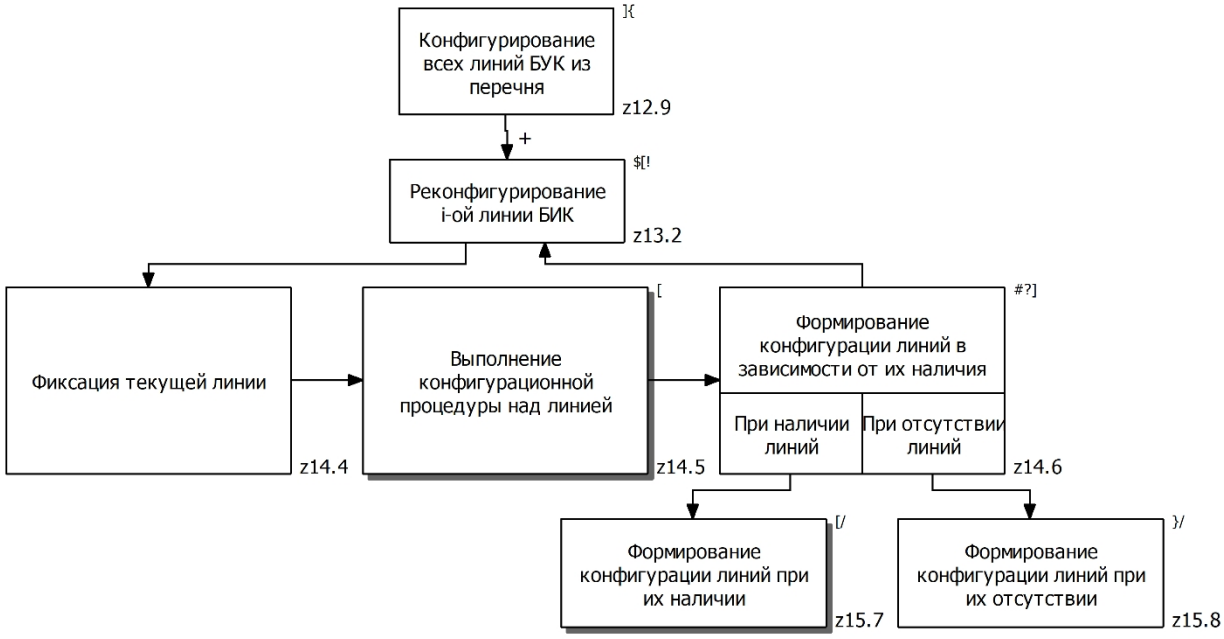


Рис. Б.44 Декомпозиция блока z12.9



Рис. Б.45 Декомпозиция блока z12.11

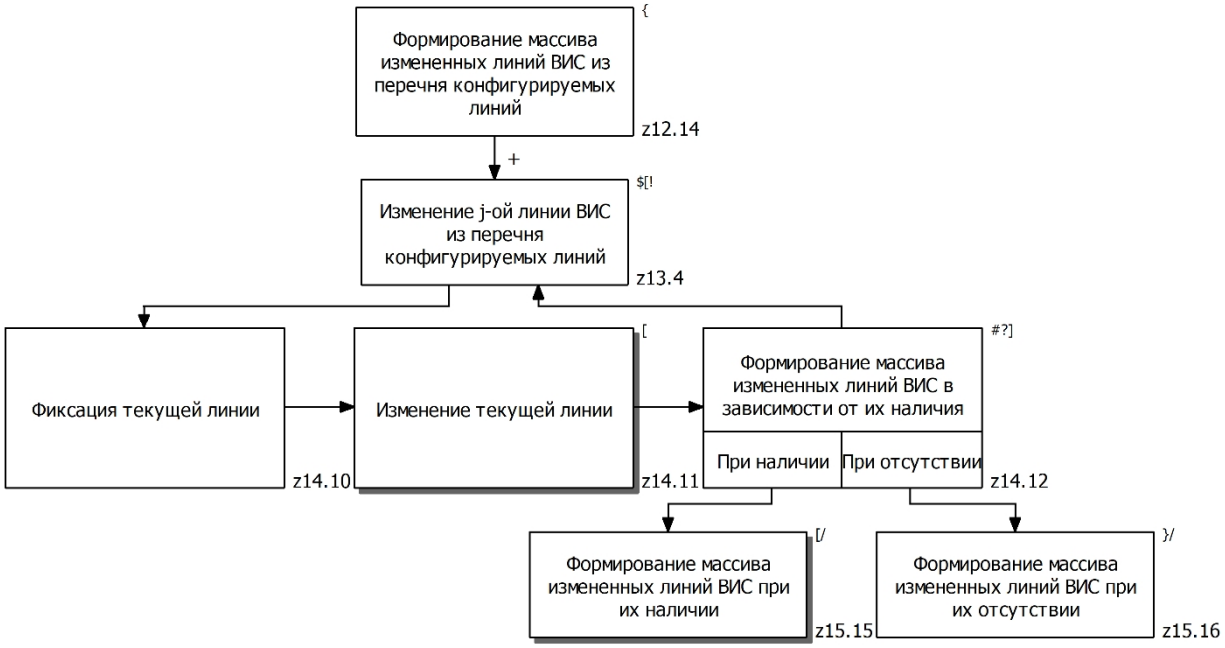


Рис. Б.46 Декомпозиция блока z12.14

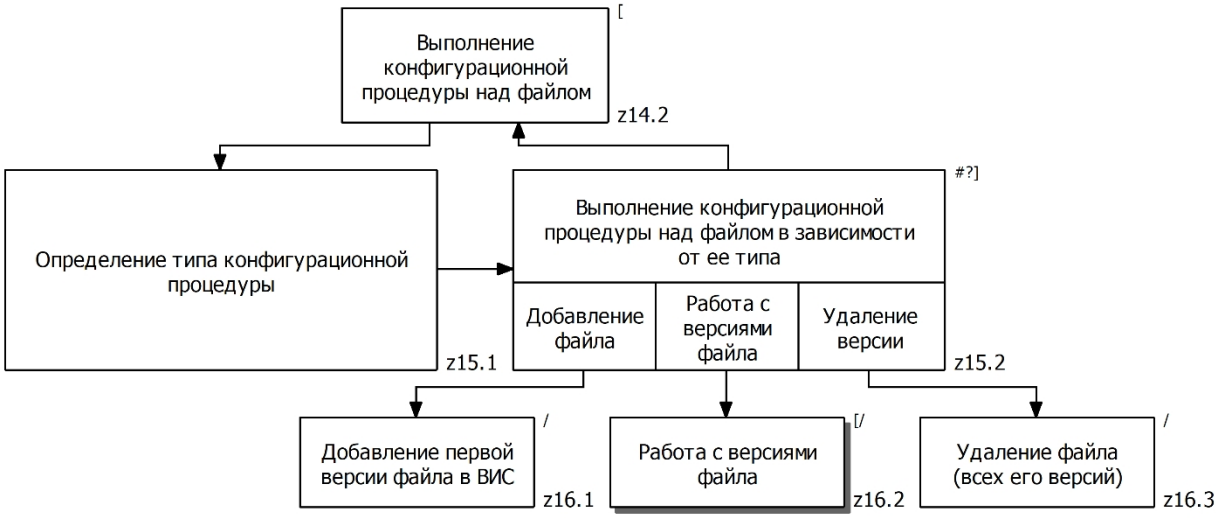


Рис. Б.47 Декомпозиция блока z14.2

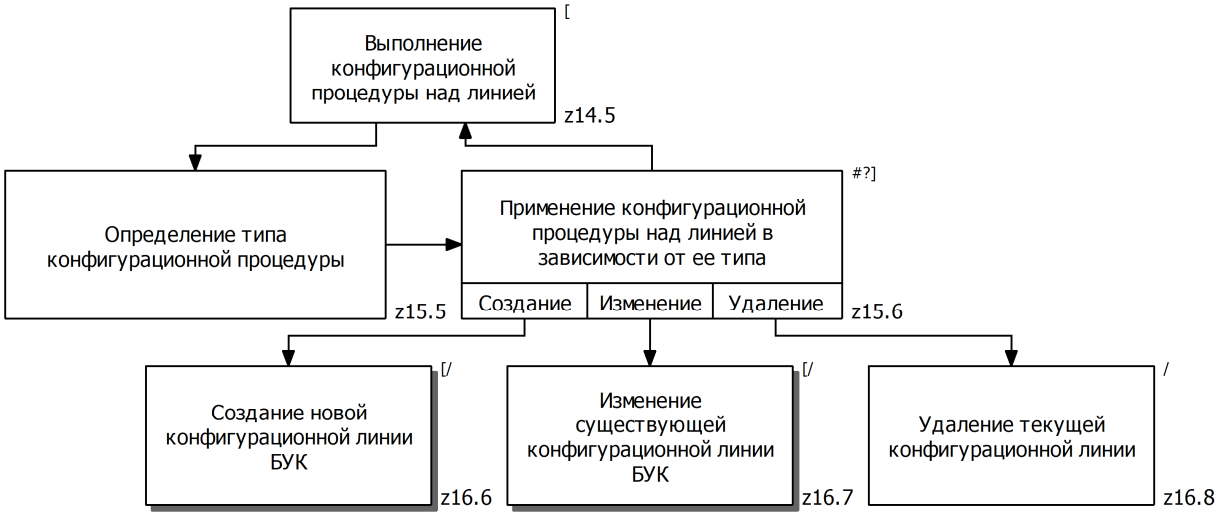


Рис. Б.48 Декомпозиция блока z14.5

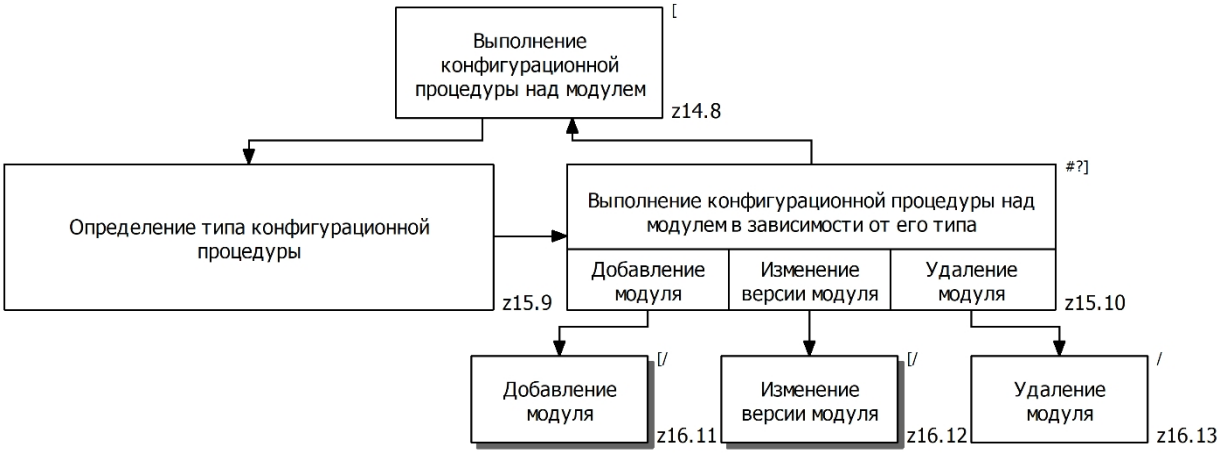


Рис. Б.49 Декомпозиция блока z14.8

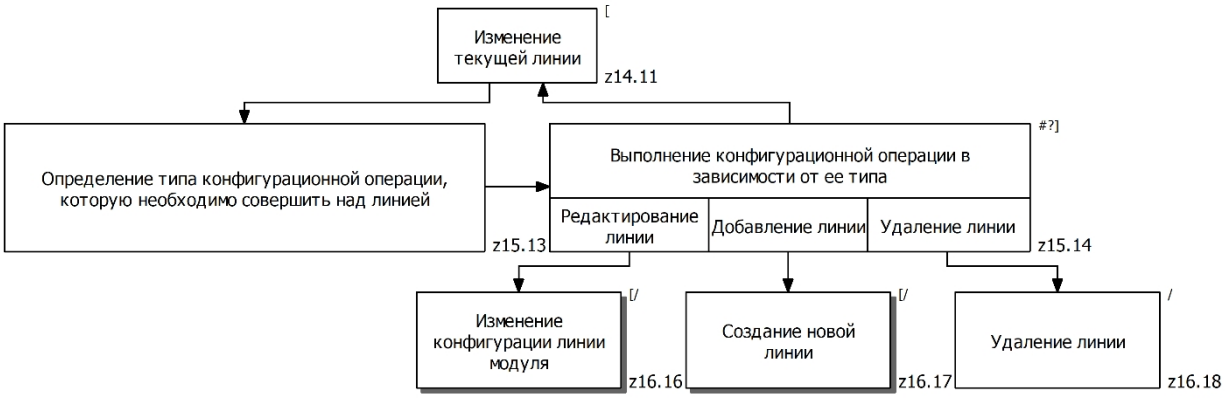


Рис. Б.50 Декомпозиция блока z14.11

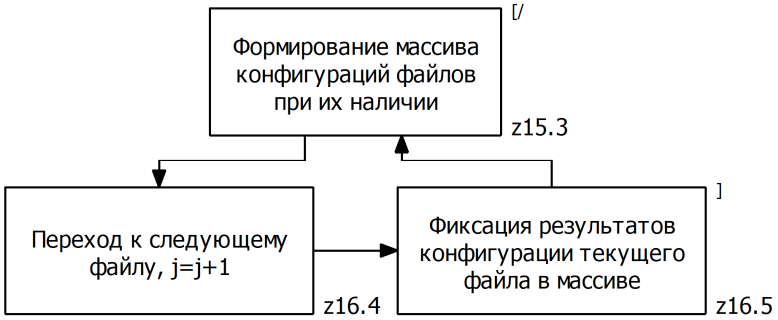


Рис. Б.51 Декомпозиция блока z15.3

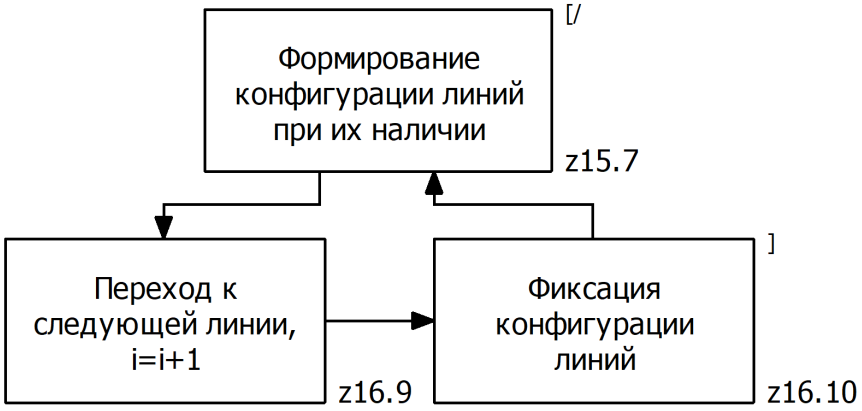


Рис. Б.52 Декомпозиция блока z15.7

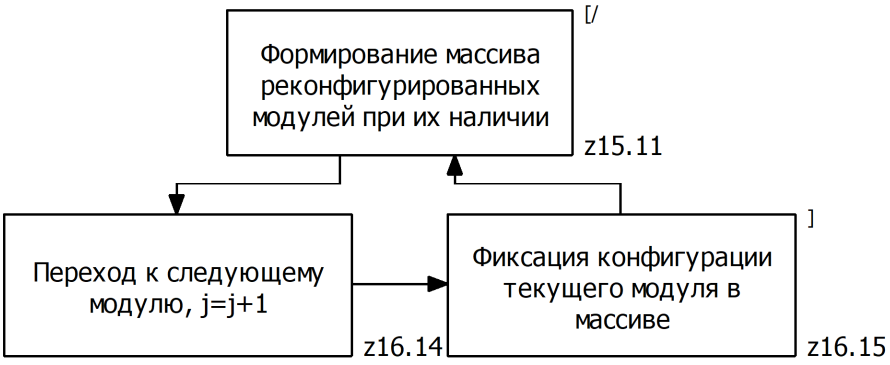


Рис. Б.53 Декомпозиция блока z15.11

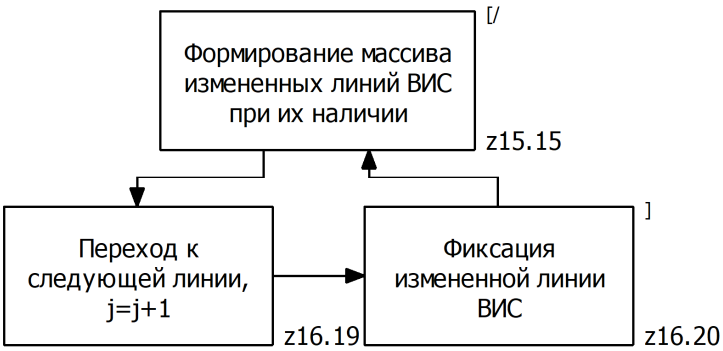


Рис. Б.54 Декомпозиция блока z15.15

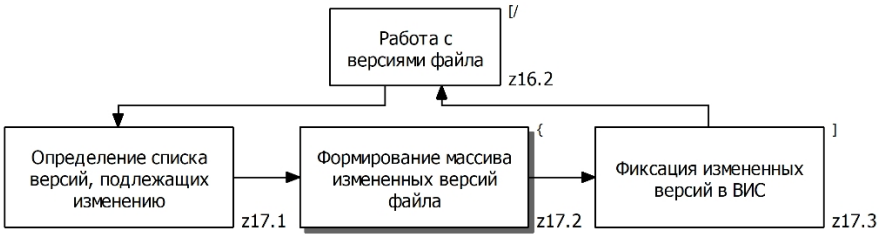


Рис. Б.55 Декомпозиция блока z16.2

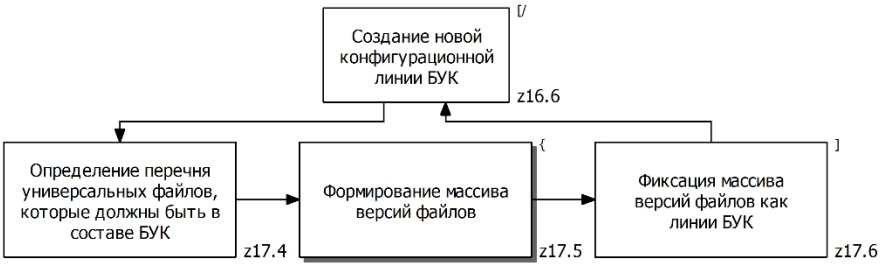


Рис. Б.56 Декомпозиция блока z16.6

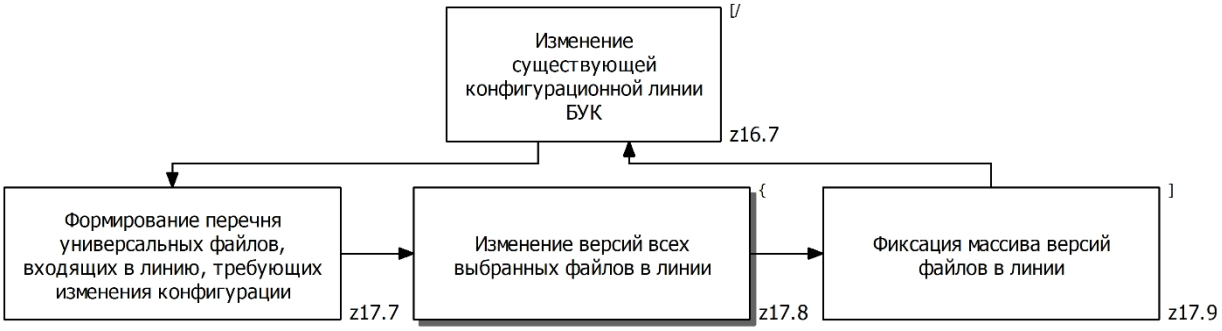


Рис. Б.57 Декомпозиция блока z16.7

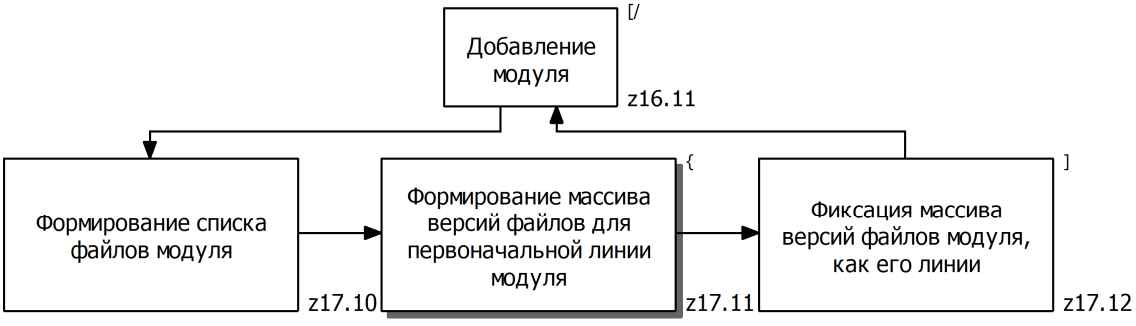


Рис. Б.58 Декомпозиция блока z16.11

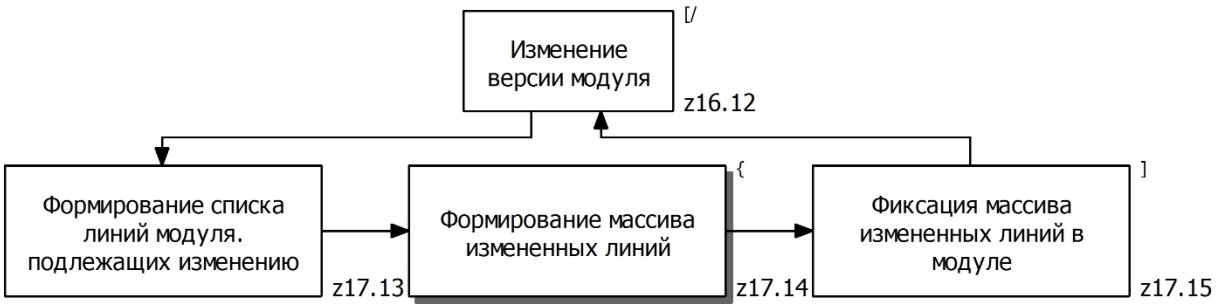


Рис. Б.59 Декомпозиция блока z16.12



Рис. Б.60 Декомпозиция блока z16.16

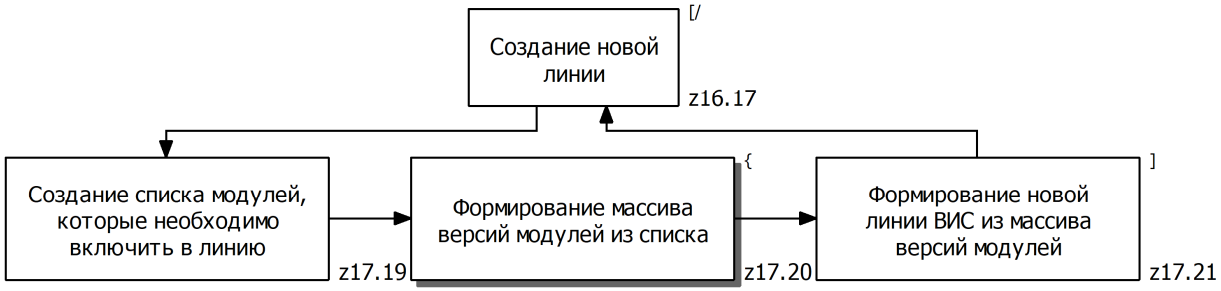


Рис. Б.61 Декомпозиция блока z16.17



Рис. Б.62 Декомпозиция блока z17.2

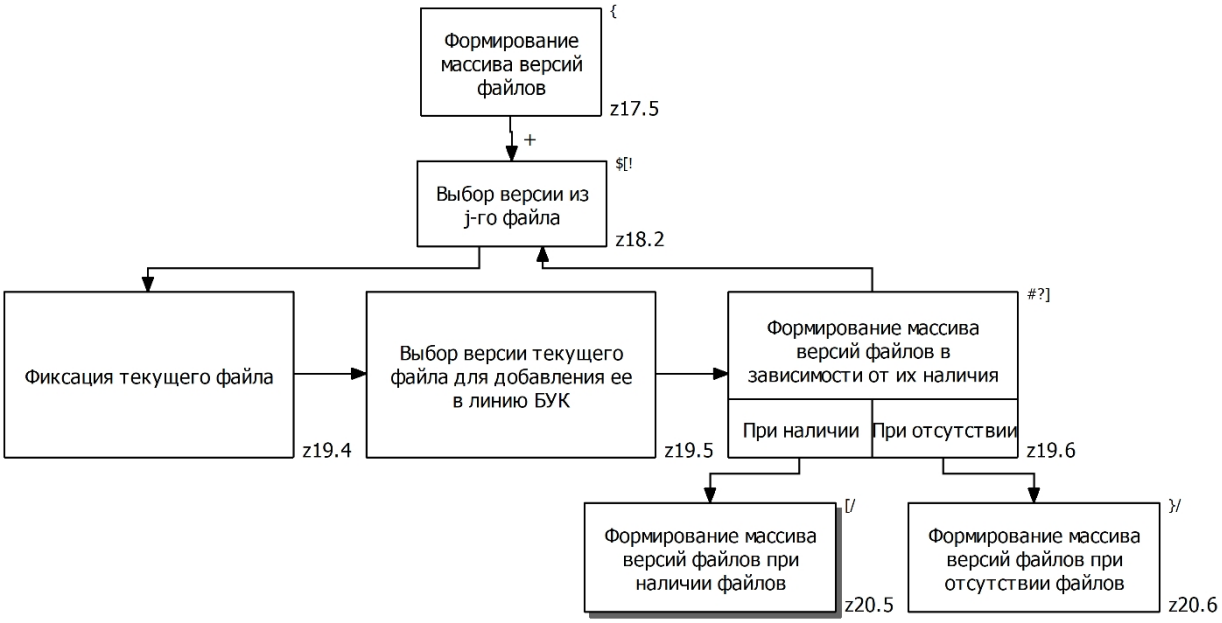


Рис. Б.63 Декомпозиция блока z17.5

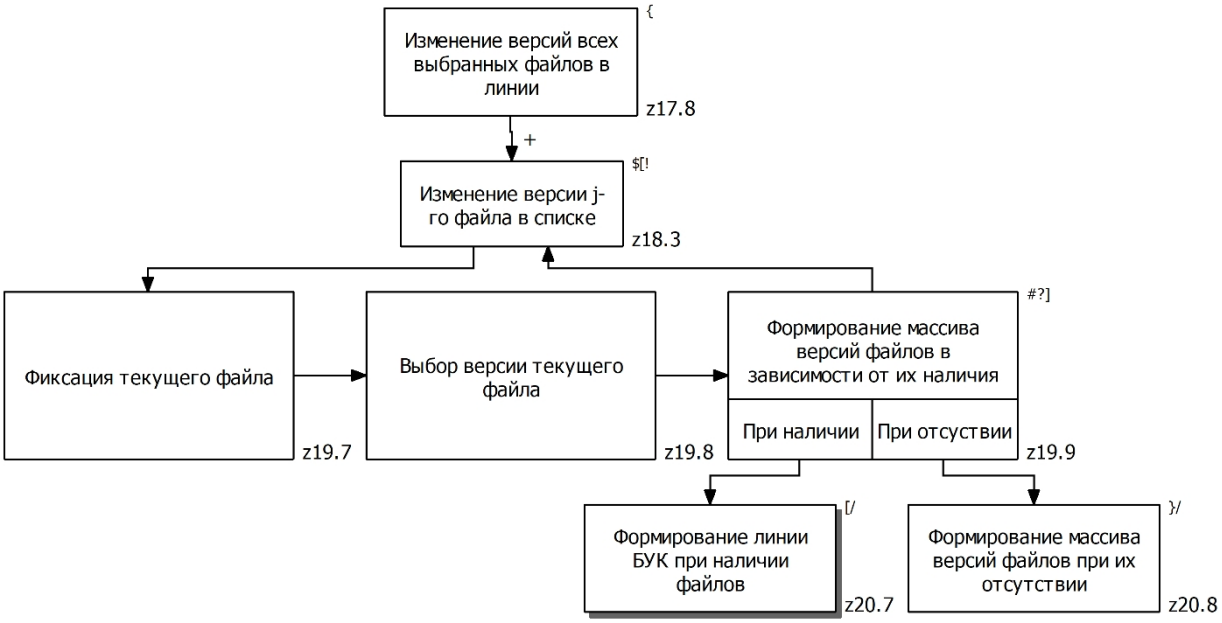


Рис. Б.64 Декомпозиция блока z17.8

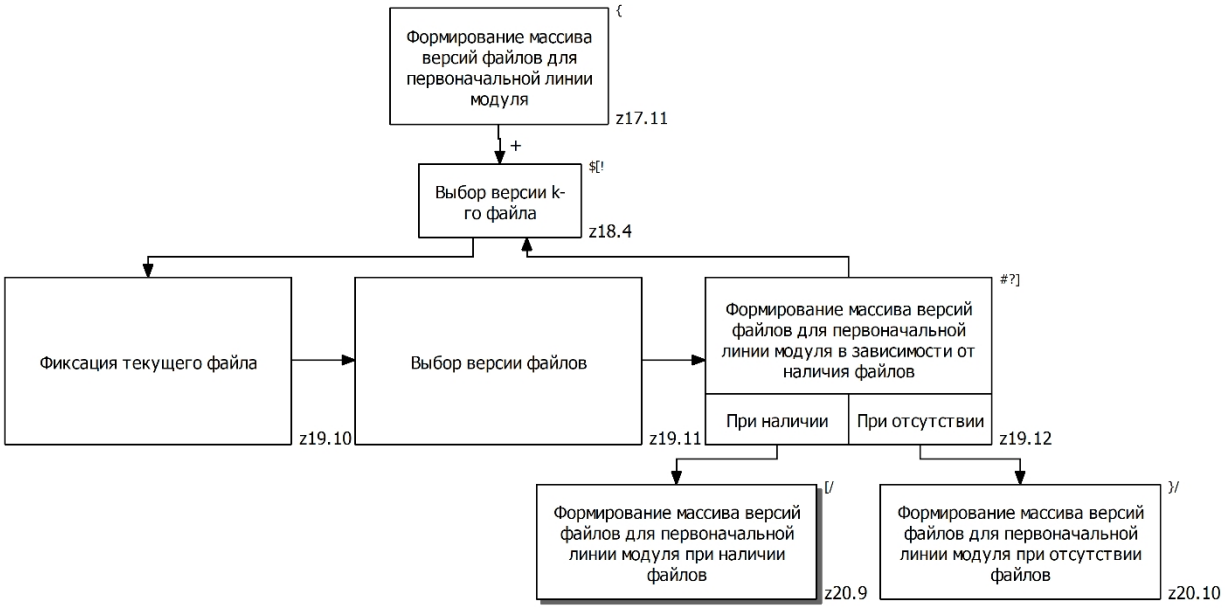


Рис. Б.65 Декомпозиция блока z17.11

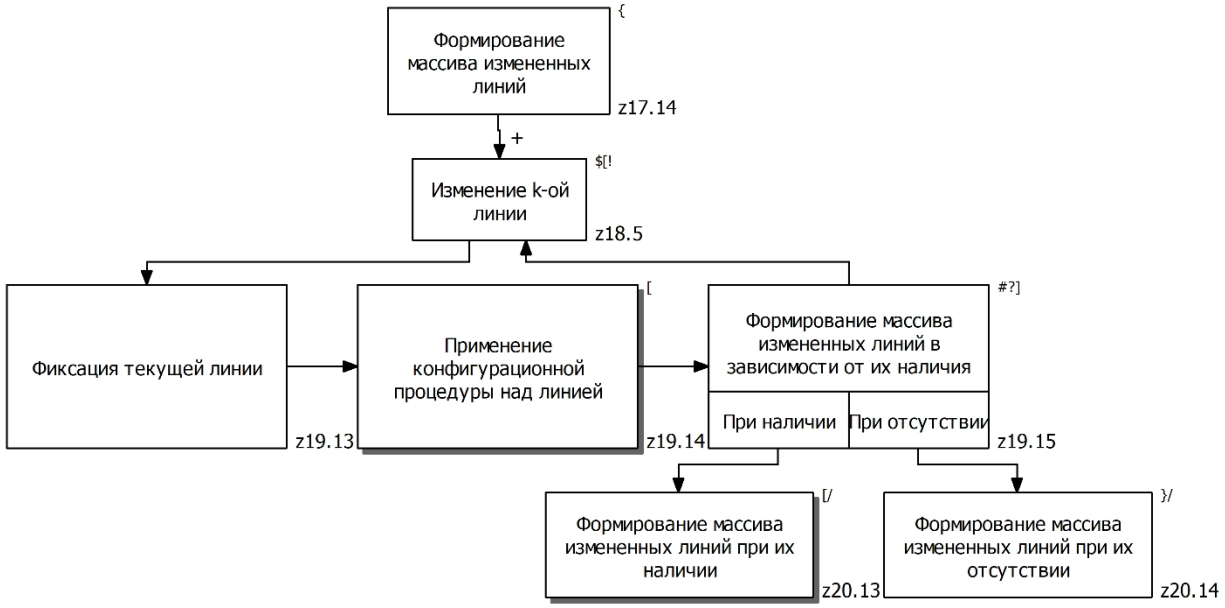


Рис. Б.66 Декомпозиция блока z17.14



Рис. Б.67 Декомпозиция блока z17.17

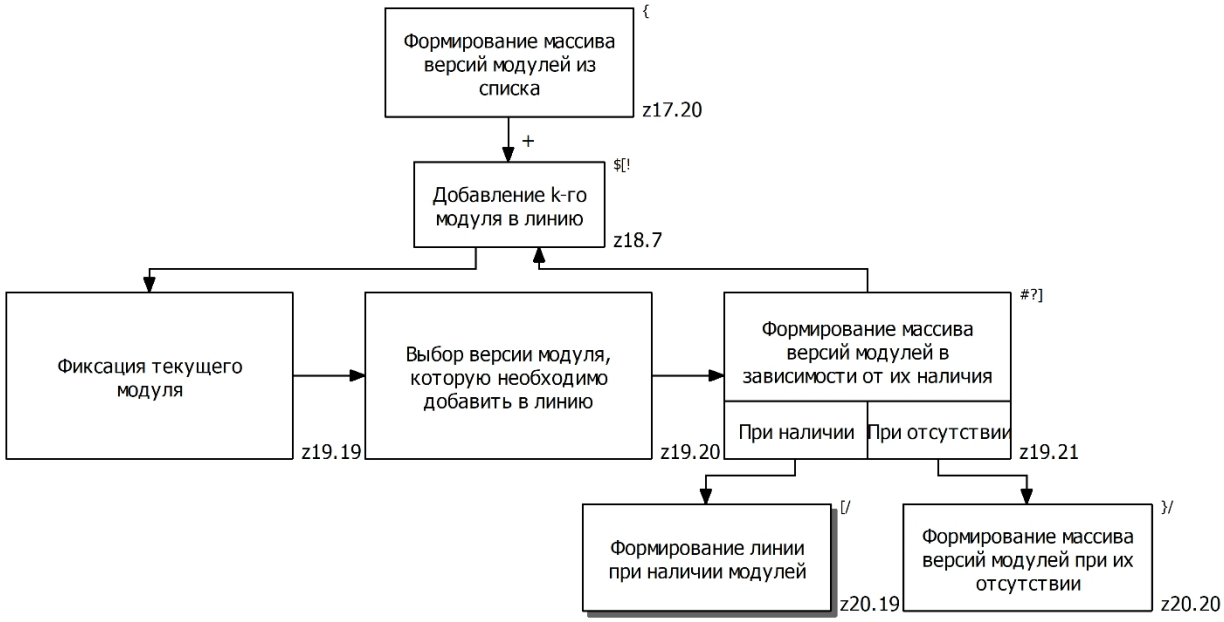


Рис. Б.68 Декомпозиция блока z17.20

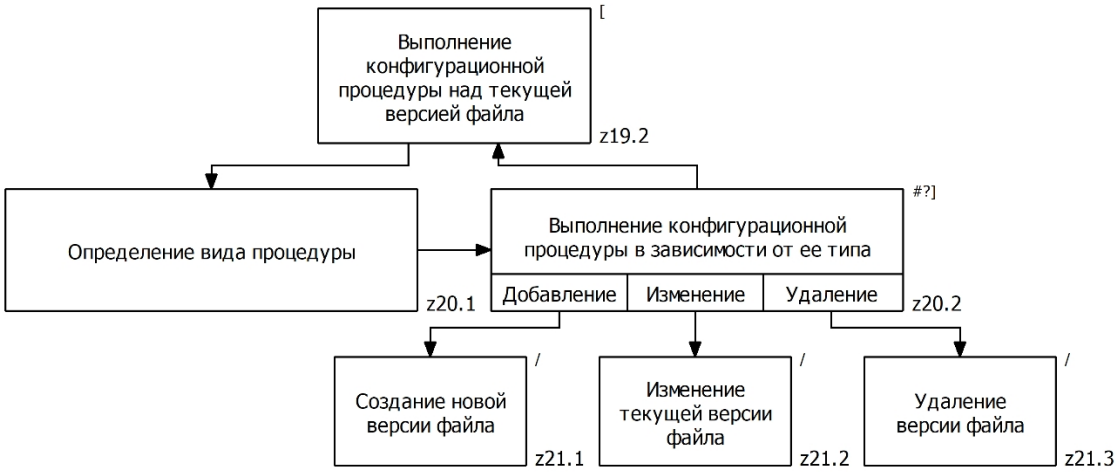


Рис. Б.69 Декомпозиция блока z19.2

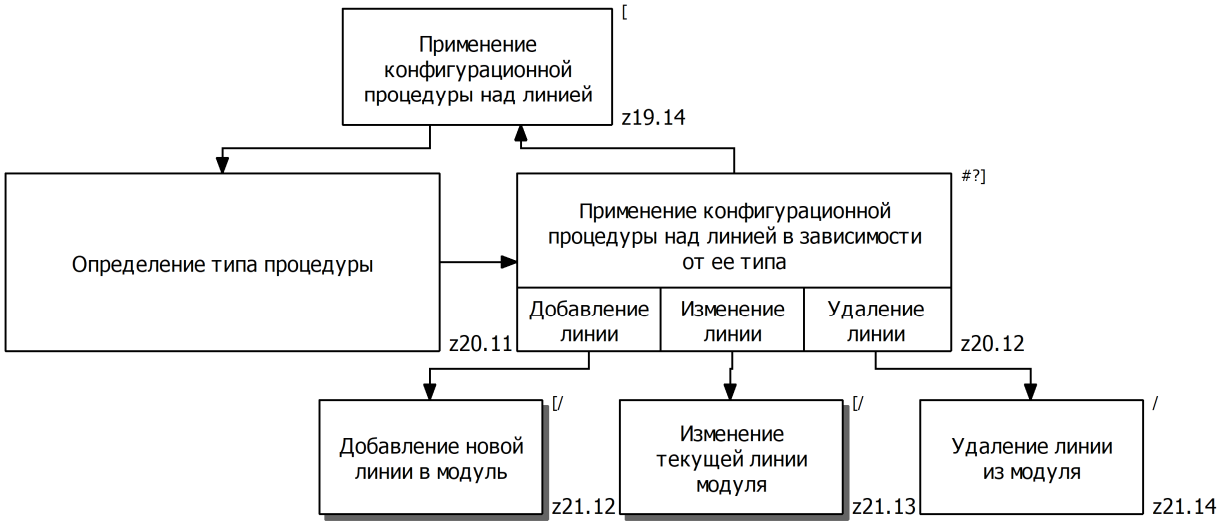


Рис. Б.70 Декомпозиция блока z19.14

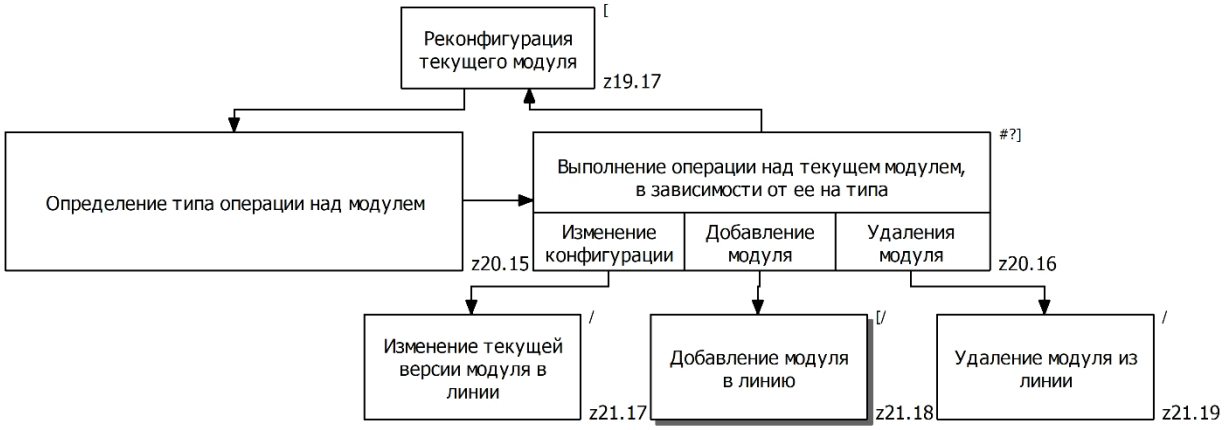


Рис. Б.71 Декомпозиция блока z19.17

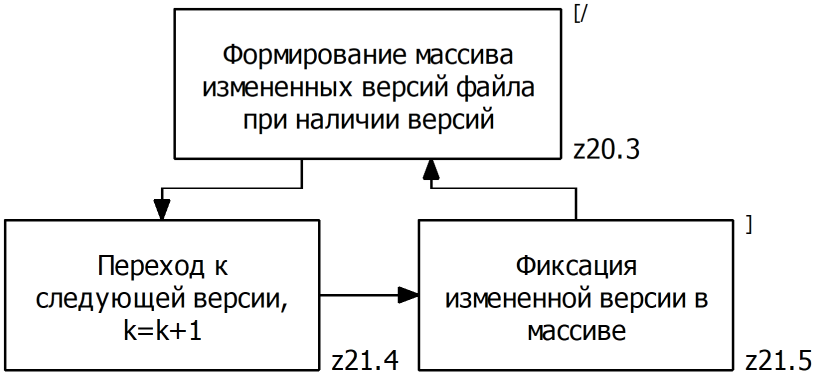


Рис. Б.72 Декомпозиция блока z20.3

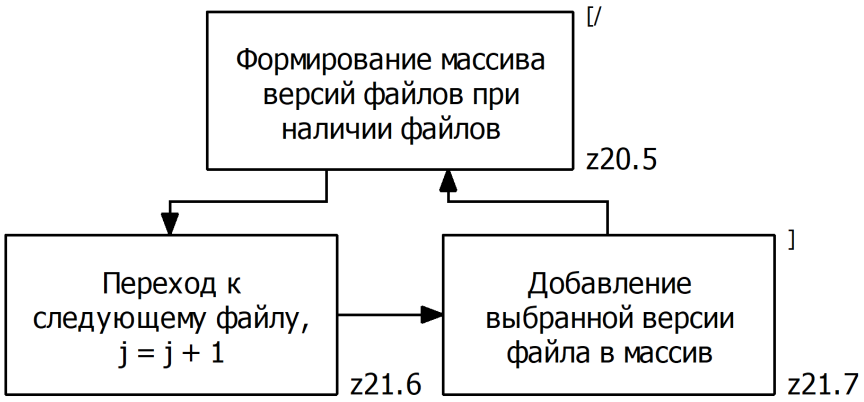


Рис. Б.73 Декомпозиция блока z20.5

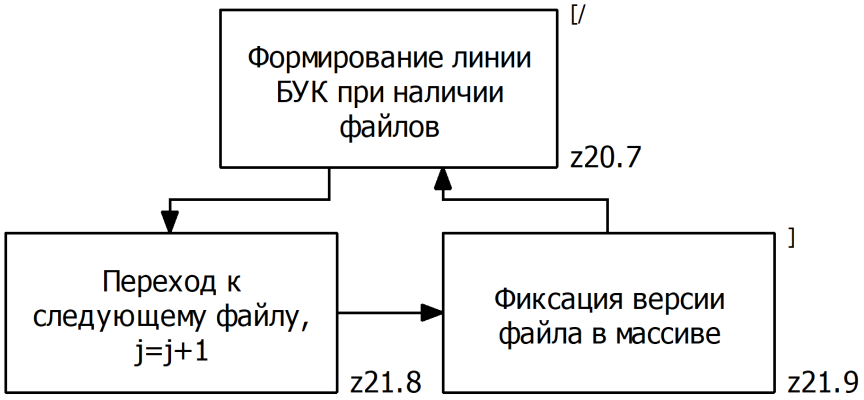


Рис. Б.74 Декомпозиция блока z20.7



Рис. Б.75 Декомпозиция блока z20.9

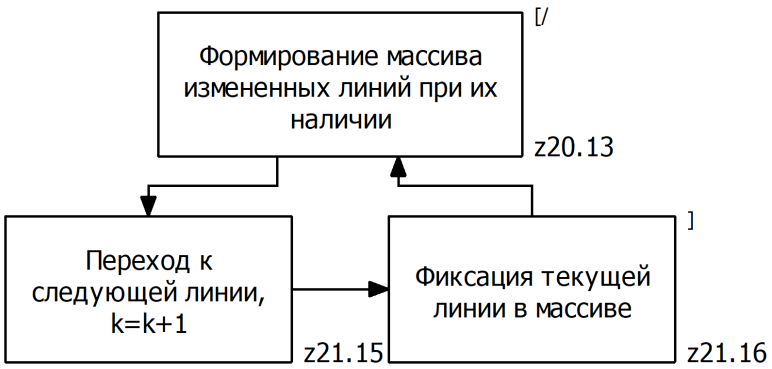


Рис.76 Декомпозиция блока z20.13

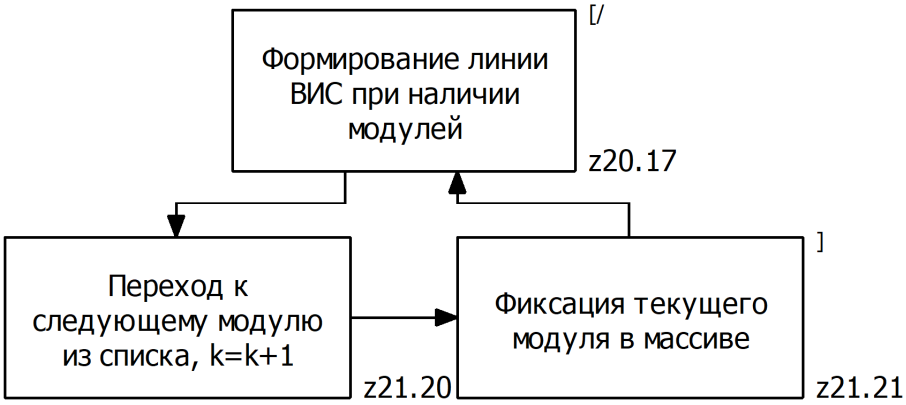


Рис. Б.77 Декомпозиция блока z20.17

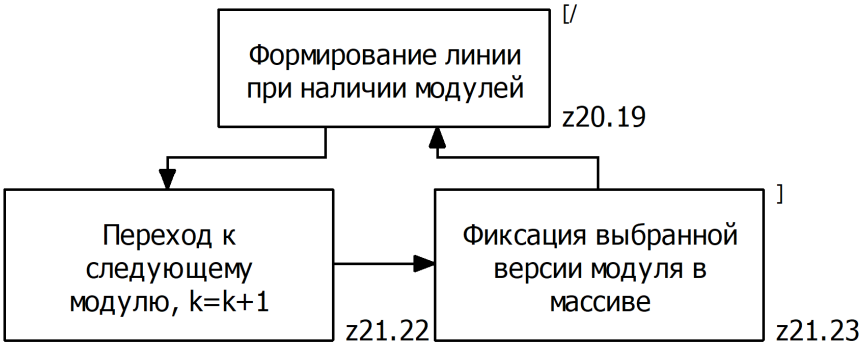


Рис. Б.78 Декомпозиция блока z20.19

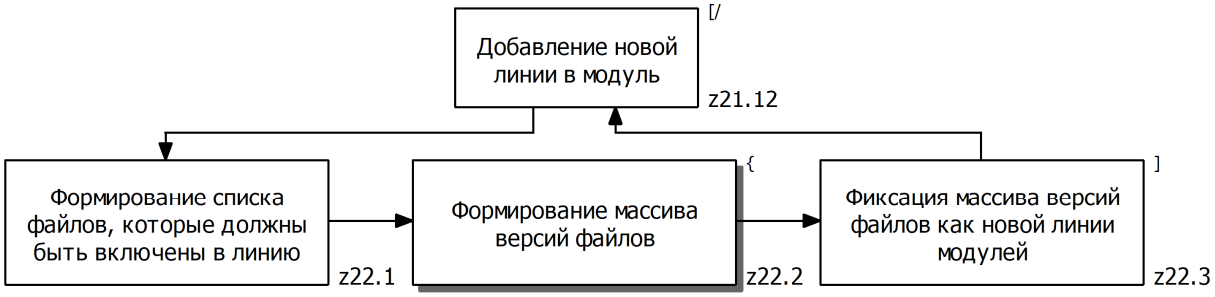


Рис. Б.79 Декомпозиция блока z21.12

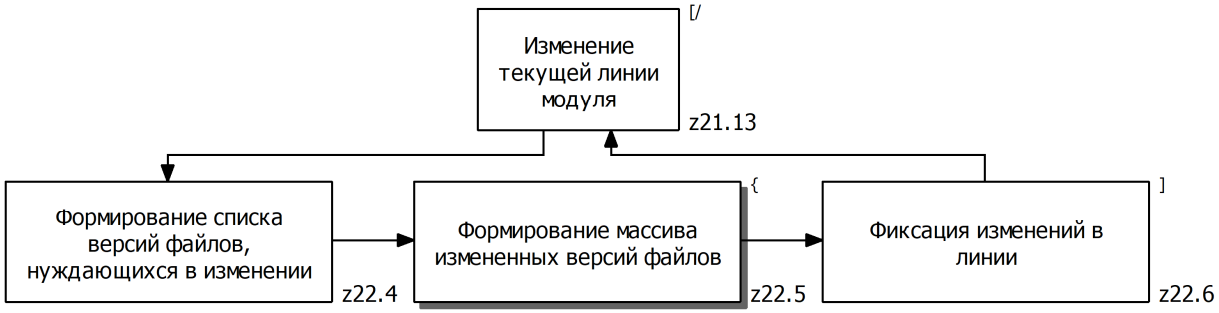


Рис. Б.80 Декомпозиция блока z21.13

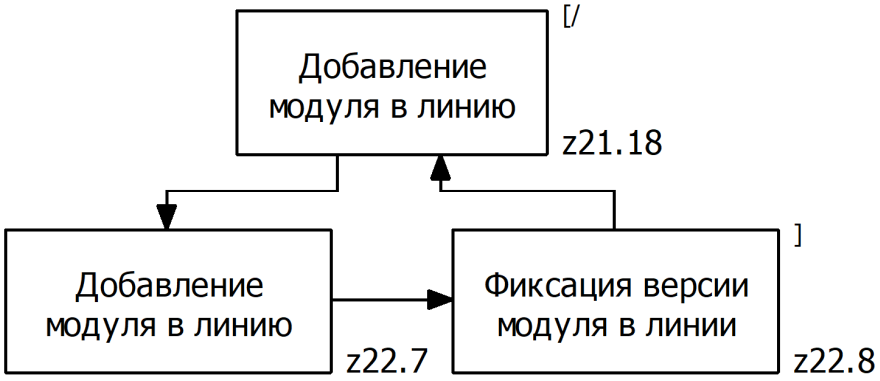


Рис. Б.81 Декомпозиция блока z21.18

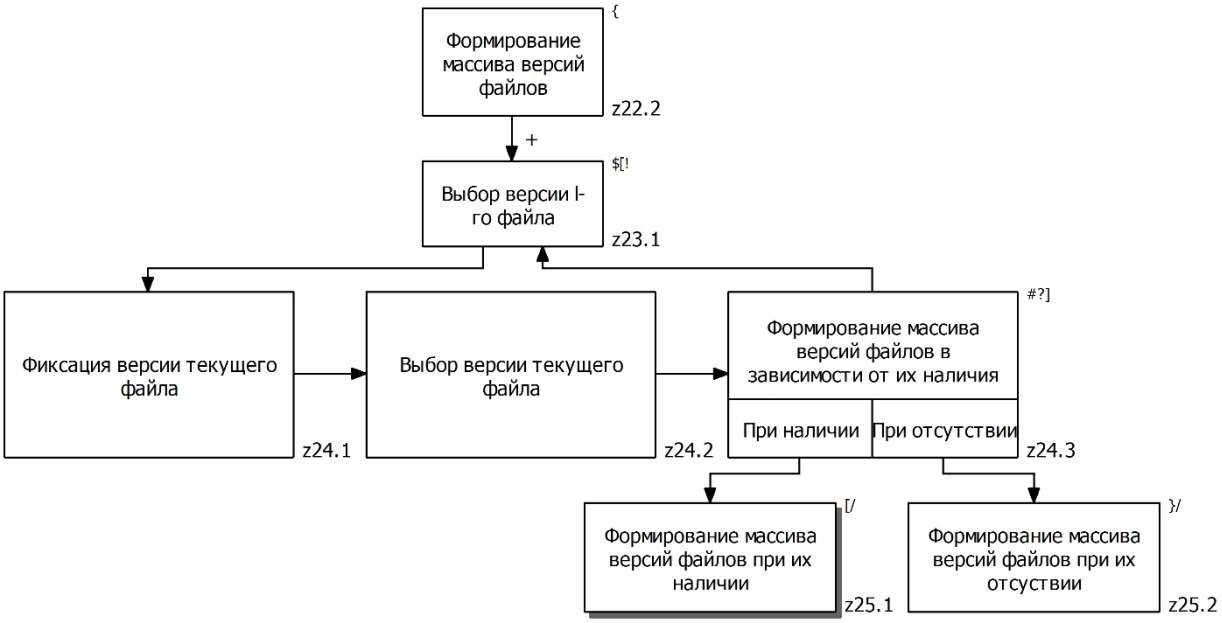


Рис. Б.82 Декомпозиция блока z22.2

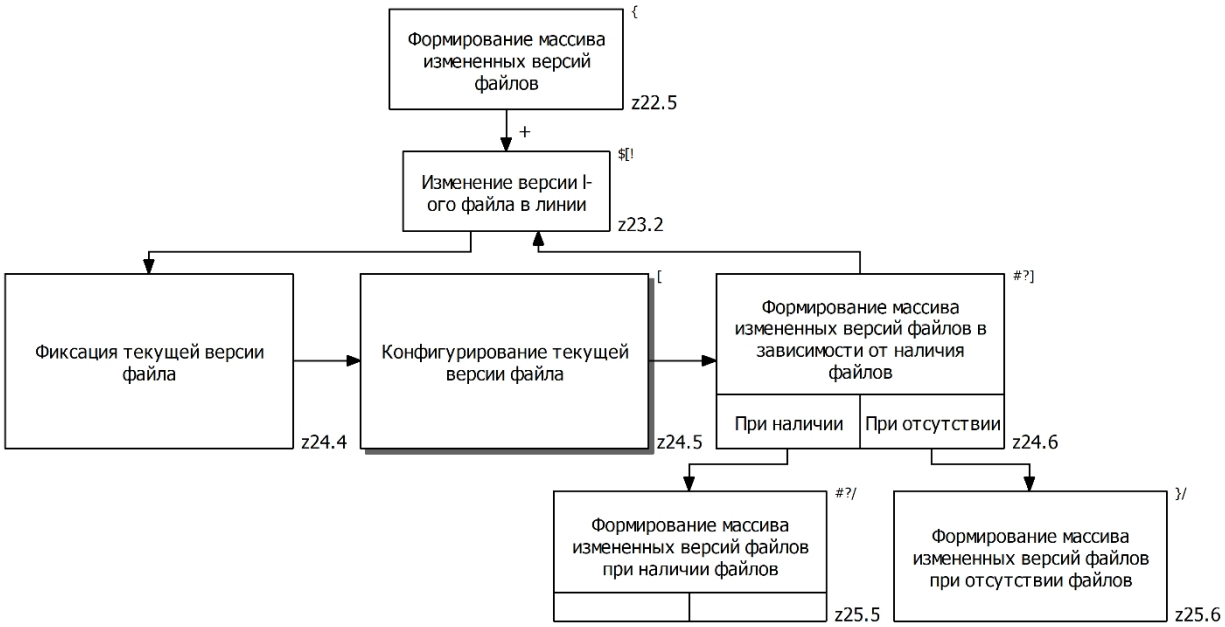


Рис. Б.83 Декомпозиция блока z22.5

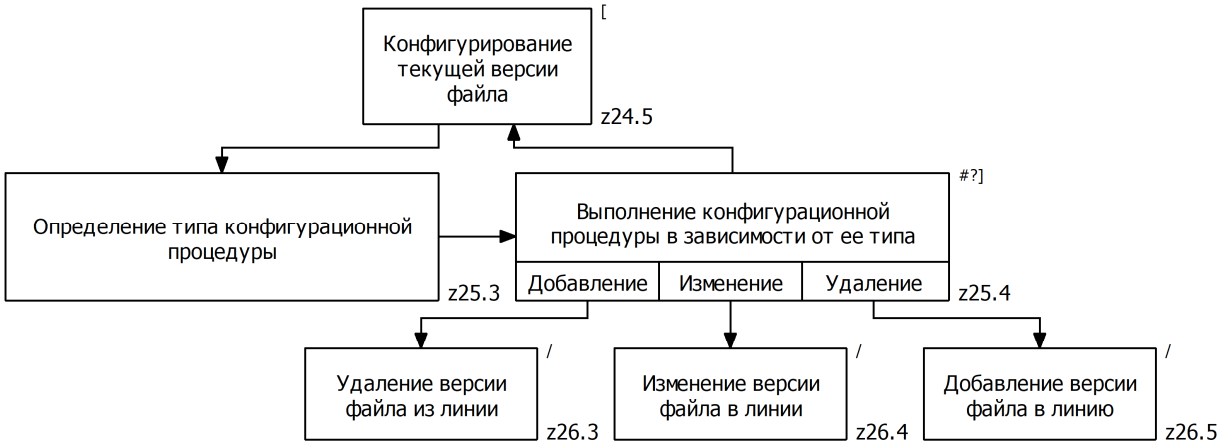


Рис. Б.84 Декомпозиция блока z24.5

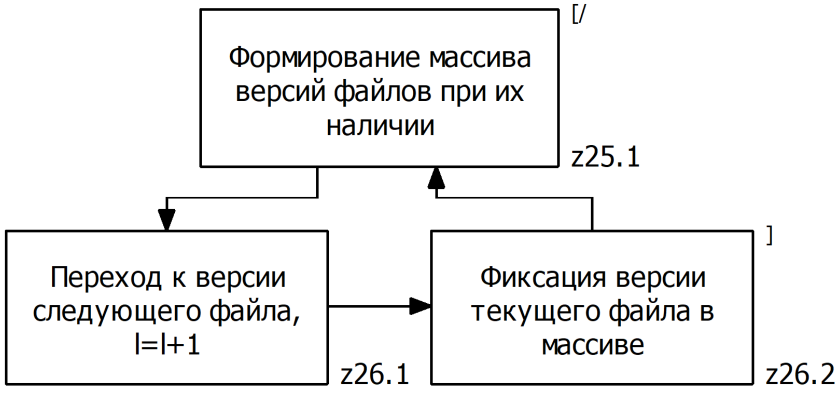


Рис.85 Декомпозиция блока z25.1

Приложение В

Инструкция по созданию модуля для ПК «ИС-2»





РУКОВОДСТВО ПРОГРАММИСТА

СОЗДАНИЕ МОДУЛЯ ДЛЯ ПРОГРАММНОГО КОМПЛЕКСА «ИС-2»

Требования:

1) Установленная среда разработки Microsoft Visual Studio 2015.

2) Установленный фреймворк Qt 5.9.1.

3) Установленное расширение на Visual Studio – Qt Visual Studio Addin.

4) Распакованные программные библиотеки ПК «ИС-2» по пути C:\IS_OUTPUT.

Разработка модуля выполняется в соответствии с методикой, представленной на Рис В.1.

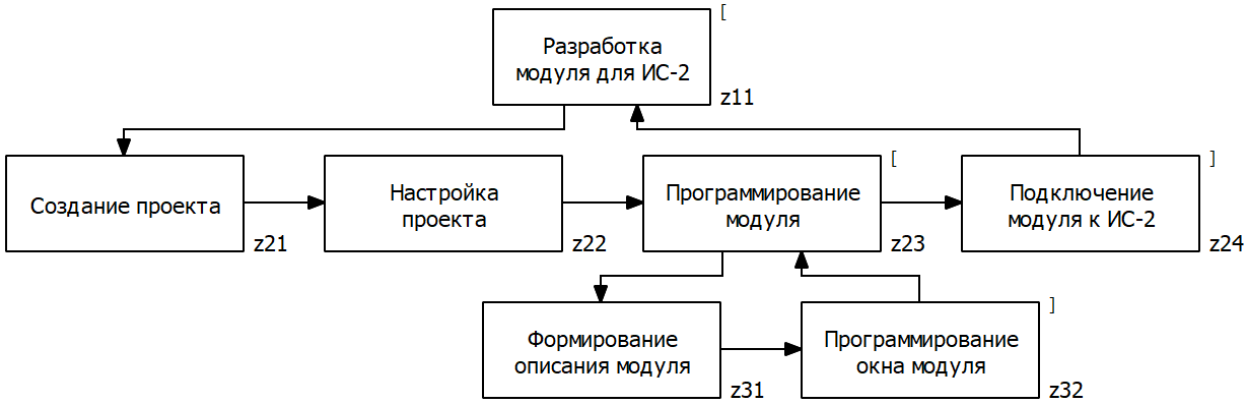


Рис В. 1. Методика создания модуля для ПК «ИС-2».

Описание и комментарии к действиям методики

Z11 Создание проекта

В среде Visual Studio 2015 создать новый проект. В качестве типа проекта указать Qt Class Library (рис В.2).

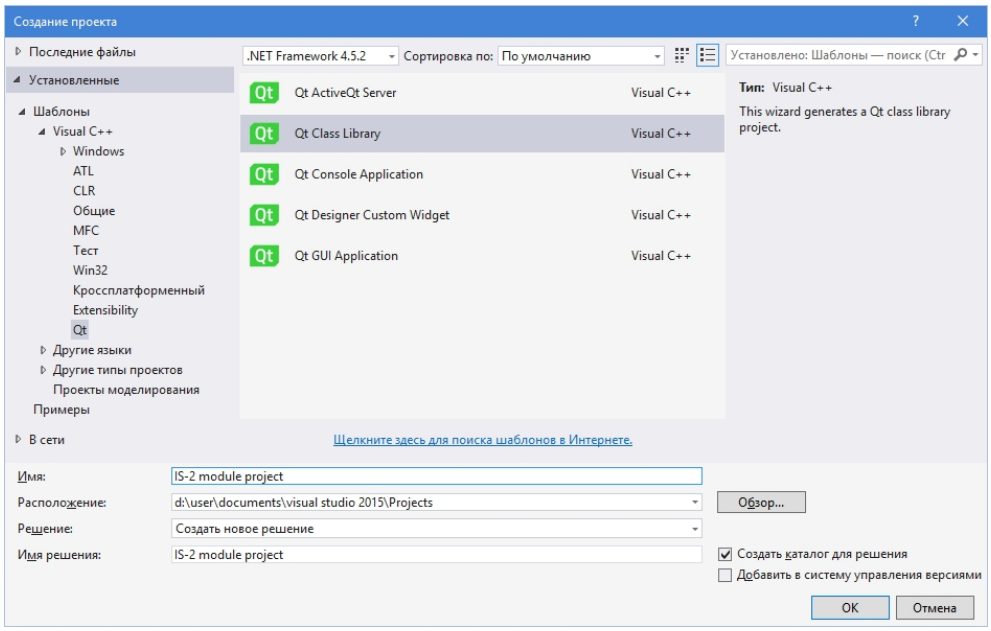


Рис В. 2. Создание проекта для нового модуля ИС-2

Установить имя проекта и его расположение по необходимости.

Подключаемые Qt-модули выбирать по необходимости, но как минимум должны быть модули, позволяющие реализовывать графический интерфейс GUI и Widgets. (рис В.3). Остальные модули можно использовать при необходимости.

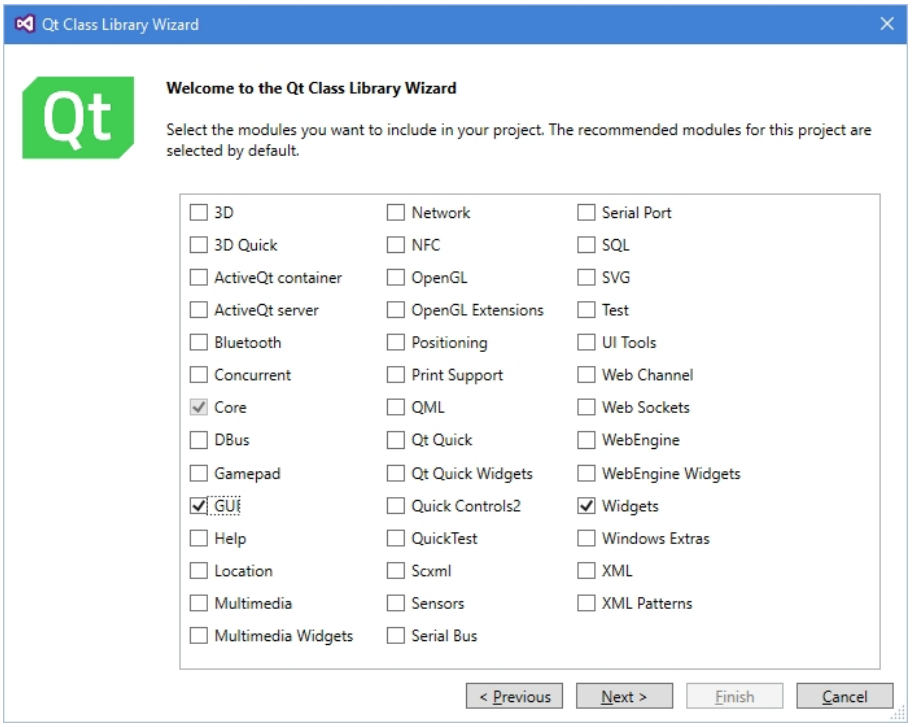


Рис В.3. Подключение модулей Qt к проекту

Z22 Настройка проекта

В свойствах проекта выставить следующие параметры:

1. Вкладка **«общее»**, для **всех** конфигураций:

Тип конфигурации	Динамическая библиотека (.dll)
Версия целевой платформы:	8.1
Выходной каталог:	C:\IS_OUTPUT\$(Configuration)\modules

2. Вкладка **«отладка»:**

В поле «команда» указать путь до исполняемого файла среды, в зависимости от необходимой версии. (IS2_PM.exe для начального моделирования, IS2_CM.exe для концептуального). Для конфигураций Debug и Release нужные файлы находятся в соответствующих папках. Данный шаг не является необходимым, но облегчит отладку модуля.

3. Вкладка «**C/C++**» -> «**Общее**»

Для всех конфигураций добавить в список «Дополнительные каталоги включаемых

файлов» следующие пути (рис В.4)::

C:\IS_OUTPUT\MAIWmodels\include

C:\IS_OUTPUT\ISsdk\include

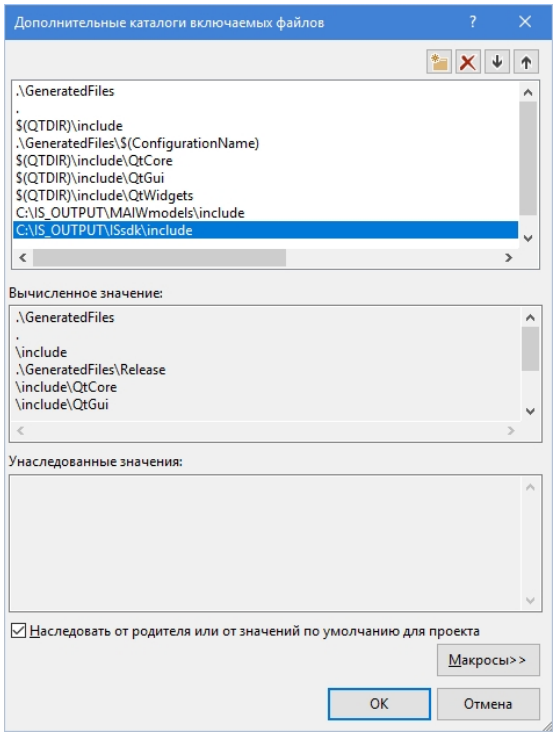


Рис В.4. Дополнительные каталоги включаемых файлов

4. Вкладка «**C/C++**» **->** «**Язык**»

Установить следующие параметры для всех конфигураций:

Считать WChar_t встроенным типом	Да

5. Вкладка «**Компоновщик**» -> «**Общее**»

Для **всех** конфигураций.

Убедится, что выходной файл имеет расширение ***.dll**

6. Вкладка «**Компоновщик**» -> «**Ввод**»

Добавить в список «дополнительные зависимости» следующие строки:

Для конфигурации **Debug:**

C:\IS_OUTPUT\ISsdk\ISsdk_d.lib

C:\IS_OUTPUT\MAIWmodels\MAIWmodels_d.lib

Для конфигурации **Release** (Рис В.5):

C:\IS_OUTPUT\ISsdk\ISsdk.lib

C:\IS_OUTPUT\MAIWmodels\MAIWmodels.lib

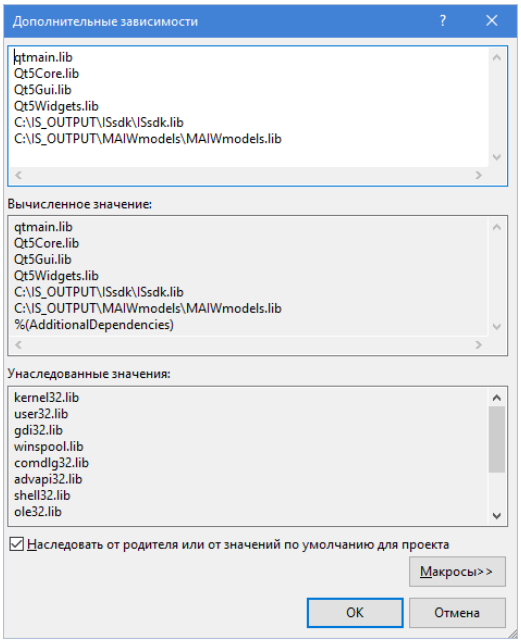


Рис В.5. Дополнительные зависимости

Проверить внесенные изменения, скомпилировав проект.

Z31. Формирование описание модуля

Перейти в файл ИМЯ_ПРОЕКТА.h (далее – h-файл). Отчистить его и создать заготовку для класса модуля (Листинг В.) (также необходимо удалить остатки класса, что был в h-файле из cpp-файла.) Имя класса может быть установлено любое. Необходимо, чтобы созданный класс являлся наследником базового для ИС-2 класса IsModule.

Листинг В.1. Заготовка класса модуля

```
#include <IS>

class SomeIS2Module : public IsModule
{
public:
  SomeIS2Module() {};
  ~SomeIS2Module() {};

  void loadModule() {}

  void showModuleWidget()
  {
  }

  ModuleInfo* setModuleInfo()
  {
  }

};
```

В функции setModuleInfo() можно указать необходимую информацию о разрабатываемом модуле (Листинг В.2).

Листинг В.2. Заполнение структуры с информацией о модуле

```
ModuleInfo* setModuleInfo()
{
  ModuleInfo *mi = new ModuleInfo;
  mi->description = "MyIS2Module is the best module in the world";
  mi->version = 0.1;
  mi->name = "MyIS2Module";
  return mi;
}
```

В файле ИМЯ_ПРОЕКТА.cpp добавить ф-цию, которая создает экземпляр класса модуля (Листинг В.В.3). Заголовок функции должен быть в точности, как приведен в листинге. Так как ИС-2 будет вызывать из файла модуля именно ее, и с нее начнётся процесс функционирования модуля.

Листинг В.В.3. Функция создания экземпляра класса модуля

```
extern "C" __declspec(dllexport) IsModule * loadModule()
{
        return ( new SomeIS2Module() );
}
```

Проверить проект, скомпилировав его.

Z33 Программирование окна модуля

В QtDesigner разработать (В панели меню Visual Studio пункт **«QtVsTools»** -> **«QtDesigner»**) интерфейс окна модуля, пример которого приведен на Рис В.В.6 и сохранить форму в директории проекта. Сохраненную форму необходимо добавить в студии к дереву проекта и скомпилировать его.

В h-файле создать заготовку для класса окна (Листинг В.В.4). Класс должен являться наследником WorkWidget – базового класса для любого окна интегрированной среды. Желательно, это сделать ПЕРЕД классом, описывающим модуль.

Листинг В.4. Класс, описывающий форму

```
#include <ui_module_form.h> //ui_имя_файла_формы!!!!!!!

class MyModuleForm : public WorkWidget
{
        Q_OBJECT

};
```

Сделать полную пересборку проекта! При этом в окне вывода должны появиться строки:

```
1> Moc'ing IS2moduleproject.h...
1>  Uic'ing module_form.ui...
```

говорящие, что студия «подцепила» файл интерфейса и его описание и обрабатывает их как файлы Qt перед компиляцией. Если строк не появилось, необходимо удалить описание класса, исключить файл интерфейса из проекта, перезапустить студию и попировать еще раз.

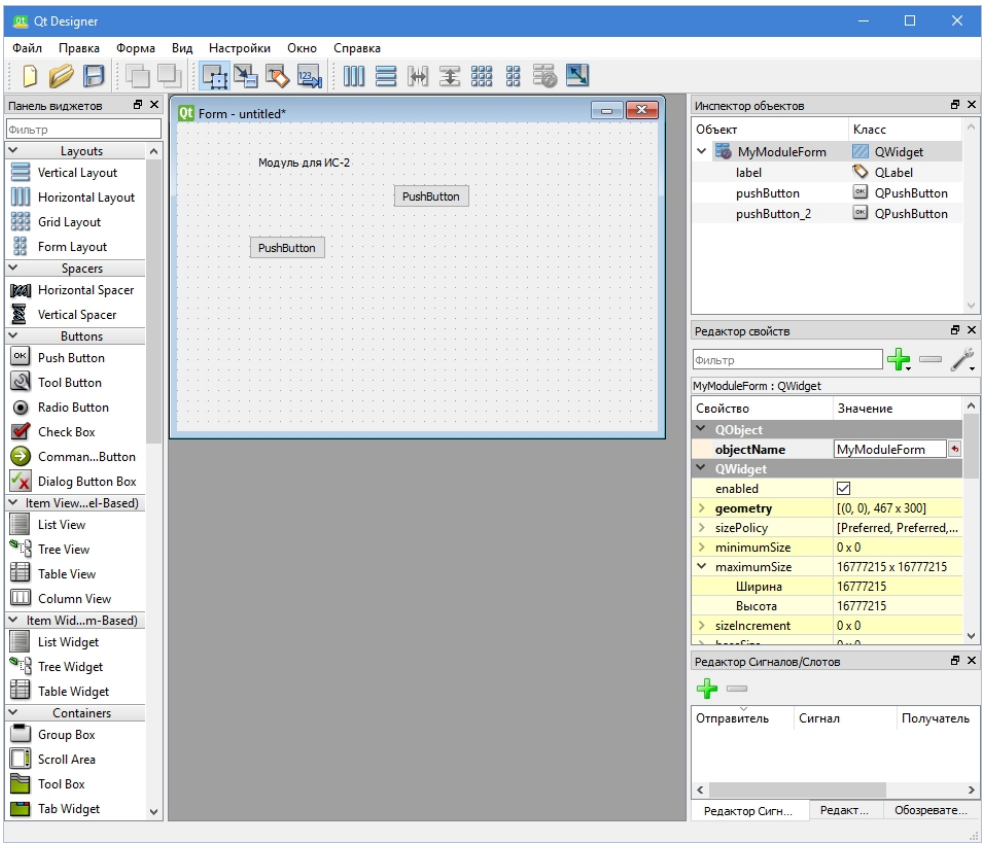


Рис В. 6. Пример создания нового окна в QtDesigner

Если все прошло успешно, можно в этом убедится, найдя в директории проекта не пустую папку GeneradedFiles (рис В.7а, В.7б).

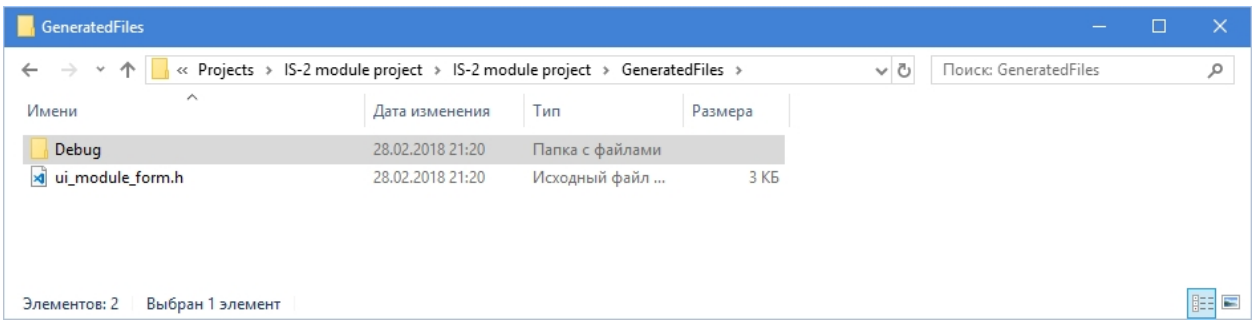


Рис В.7а. Папка GeneratedFiles

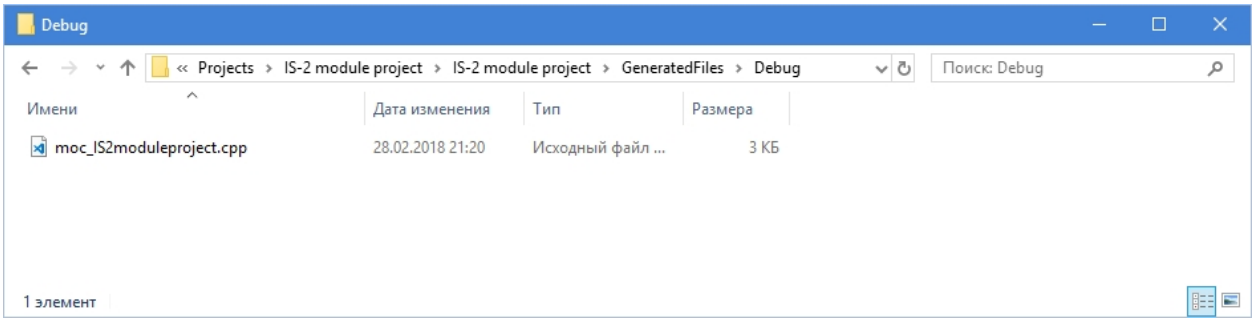


Рис В.7б. Папка GeneratedFiles

Для завершения создания окна модуля, необходимо привести код к виду как показано на листингах В.5, В.6.

Листинг В. 5. Класс, описывающий окно модуля. h-файл

```
#include <ui_module_form.h> //ui_имя_файла_формы.h !!!!!!!

class MyModuleForm : public WorkWidget
{
        Q_OBJECT

        public:
                MyModuleForm(QWidget *parent = 0);
                ~MyModuleForm();

        protected:

                Ui_MyModuleForm ui;
//класс Ui_MyModuleForm генерируется в процессе компиляции.
//Его имя будет сформировано из
//префекса Ui_ и значения свойства окна objectName, задаваемого в QtDesigner
//В этом примере название окна в дизайнере - MyModuleForm.
};
```

Листинг В.6. Класс, описывающий окно модуля. cpp-файл.

```
#include "IS2moduleproject.h"

. . .

MyModuleForm::MyModuleForm(QWidget *parent) :WorkWidget(parent)
{
        ui.setupUi(this);

        widgetName = QStringLiteral("Мой модуль для ИС-2");
}

MyModuleForm::~MyModuleForm()
{
}
```

После того, как окно сформировано, необходимо создать функцию, создающую экземпляр этого кона в классе модуля (Листинг В.7).

Листинг В.7. Функция регистрации модуля в среде

```
void showModuleWidget()
{
        RegisterModuleWidget(new MyModuleForm());
}
```

В результате, полученные файлы кода должны иметь вид, как показано в листингах В.8, В.9.

Листинг В.8. Заголовочный файл с описанием классов модуля и окна

```
#include <IS>


#include <ui_module_form.h> //ui_имя_файла_формы.h !!!!!!!

class MyModuleForm : public WorkWidget
{
        Q_OBJECT

        public:
                MyModuleForm(QWidget *parent = 0);
                ~MyModuleForm();

        protected:

                Ui_MyModuleForm ui;
//класс Ui_MyModuleForm генерируется в процессе компиляции.
//Его имя будет сформировано из
//префекса Ui_ и значения свойства окна objectName, задаваемого в QtDesigner
//В этом примере название окна в дизайнере - MyModuleForm.
```

```
};

class SomeIS2Module : public IsModule
{
public:
        SomeIS2Module() {};
        ~SomeIS2Module() {};

        void loadModule() {}

        void showModuleWidget()
        {
                RegisterModuleWidget(new MyModuleForm());
        }

        ModuleInfo* setModuleInfo()
        {
                ModuleInfo *mi = new ModuleInfo;
                mi->description = "MyIS2Module is the best module in the world";
                mi->version = 0.1;
                mi->name = "MyIS2Module";
                return mi;
        }

};
```

Листинг В.9. Файл с кодом реализации функций класса окна

```
#include "IS2moduleproject.h"

extern "C" __declspec(dllexport) IsModule * loadModule()
{
        return ( new SomeIS2Module() );
}

MyModuleForm::MyModuleForm(QWidget *parent) :WorkWidget(parent)
{
        ui.setupUi(this);
        widgetName = QStringLiteral("Мой модуль для ИС-2");
}

MyModuleForm::~MyModuleForm()
{
}
```

Z24 Подключение модуля к ИС-2

В директории ИС-2 (в версии Debug или Release, в зависимости от кого, какая версия модуля собиралась) в папке modules должен появится скомпилированный файл модуля. В примере это «IS-2 module project.dll».

В директории ИС-2 имеются файлы с описанием конфигураций среды default_PM.is2conf для версии начального моделирования default_CM.is2conf для версии концептуального

моделирования.

Необходимо добавить созданный модуль в ПК «ИС-2», например, в версию для концептуального моделирования. Для этого требуется открыть файл default_CM.is2conf с помощью текстового редактора. По умолчанию, содержание файла определено следующим образом:

```
<configuration description="">
      <module filename="htttp11.dll"/>
      <module filename="CS_visualisation.dll"/>
      <module filename="CM_auto.dll"/>
      <module filename="Model_Visualiser.dll"/>
      <module filename="CM_spec_WEB.dll"/>
      <module filename="CM_matrix_web.dll"/>
 </configuration>
```

Каждый тег <module> содержит имя файла модуля, загружаемого из папки modules в директории ИС-2.

Для добавления своего модуля необходимо добавить запись о своем модуле:

```
<module filename="IS-2 module project.dll"/>
```

После запуска среды, в ней появится новая вкладка с интерфейсом разработанной заготовки модуля (Рис В.8).

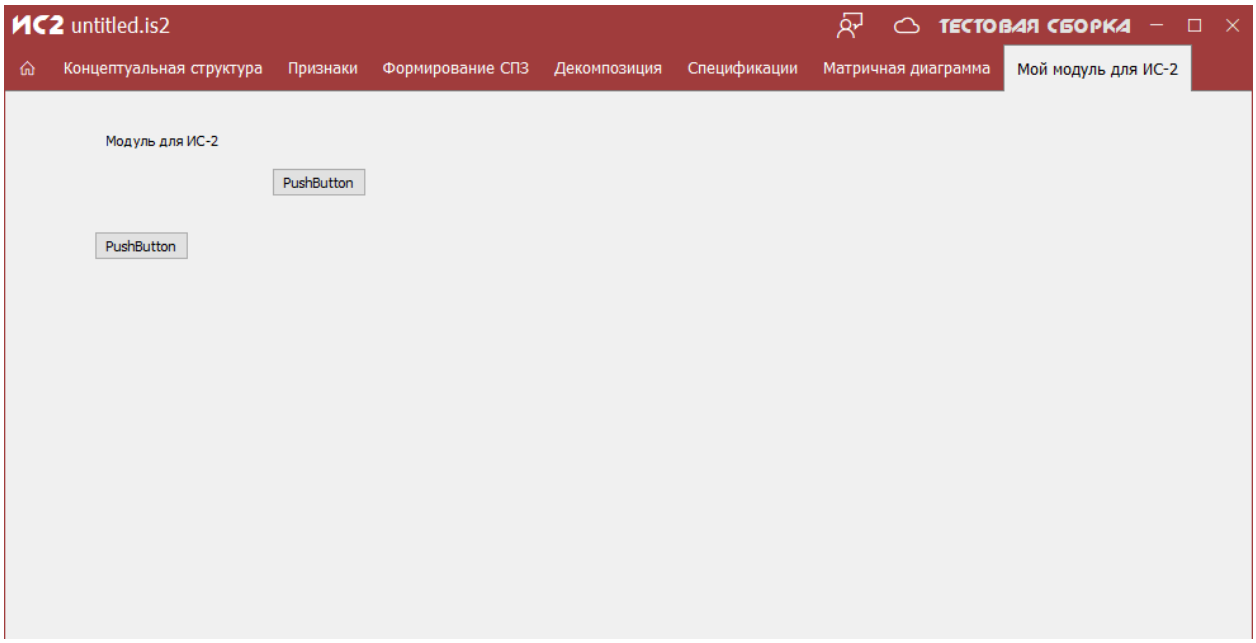


Рис В.8. Подключение к среде нового модуля

Приложение Г

Структуры данных ПК «ИС-2»

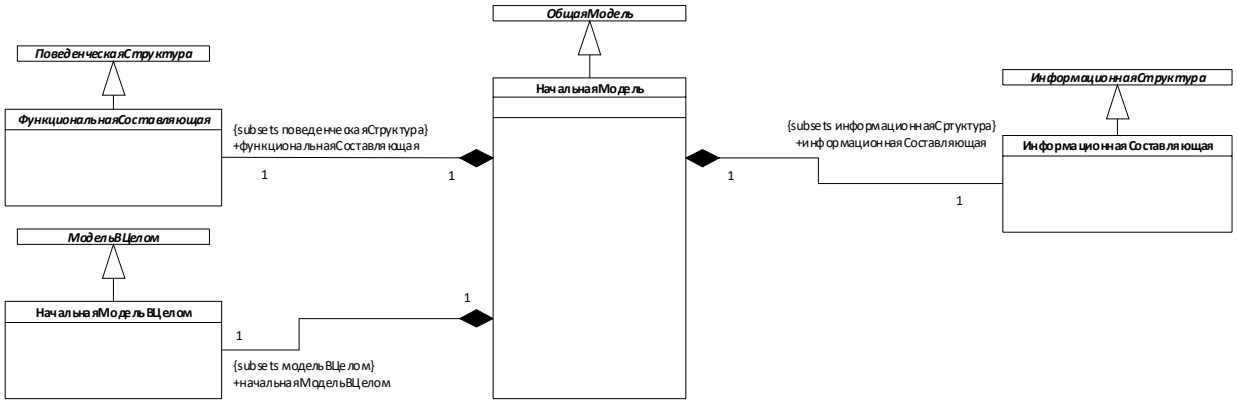


Рис. Г.1 Диаграмма классов начальной модели

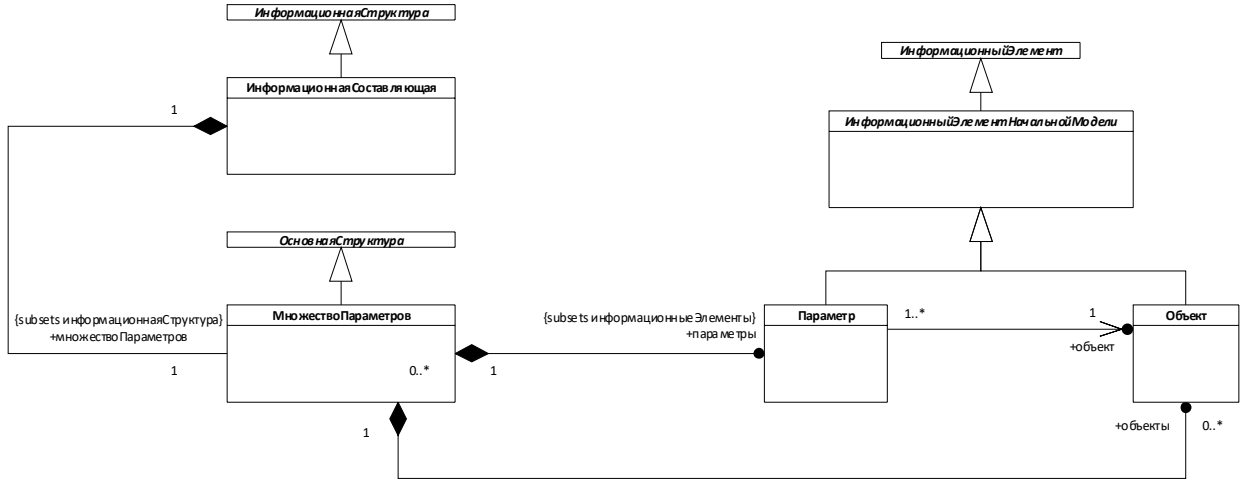


Рис. Г.2 Диаграмма классов информационной составляющей начальной модели

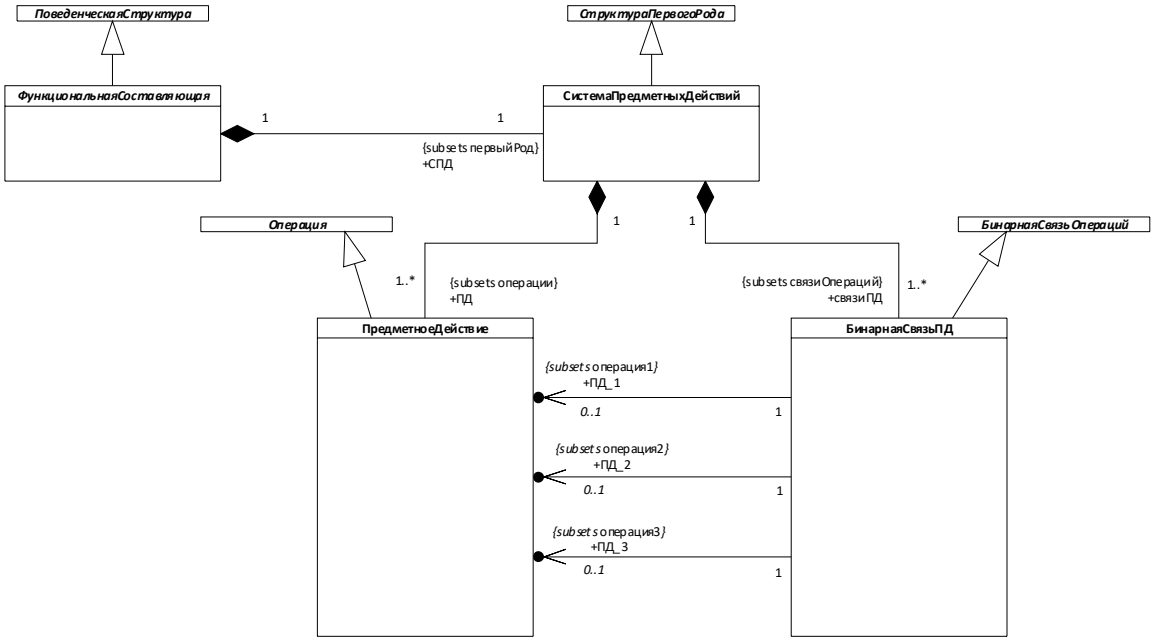


Рис. Г.3 Диаграмма классов функциональной составляющей начальной модели

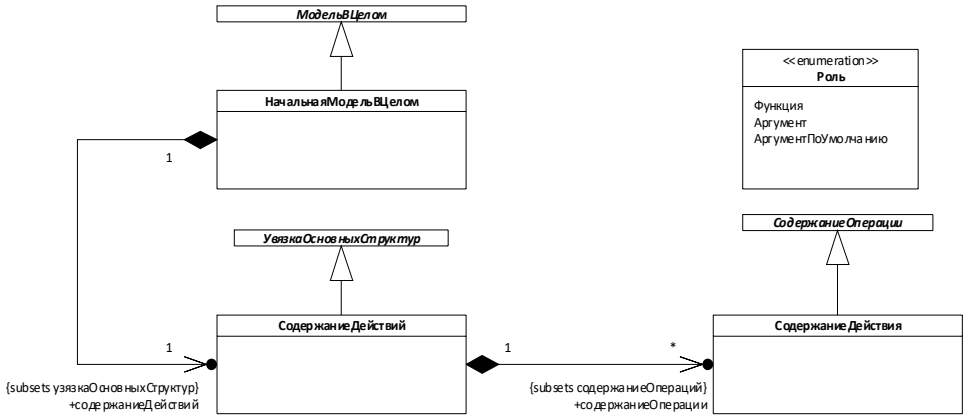


Рис. Г.4 Диаграмма классов содержательной составляющей начальной модели

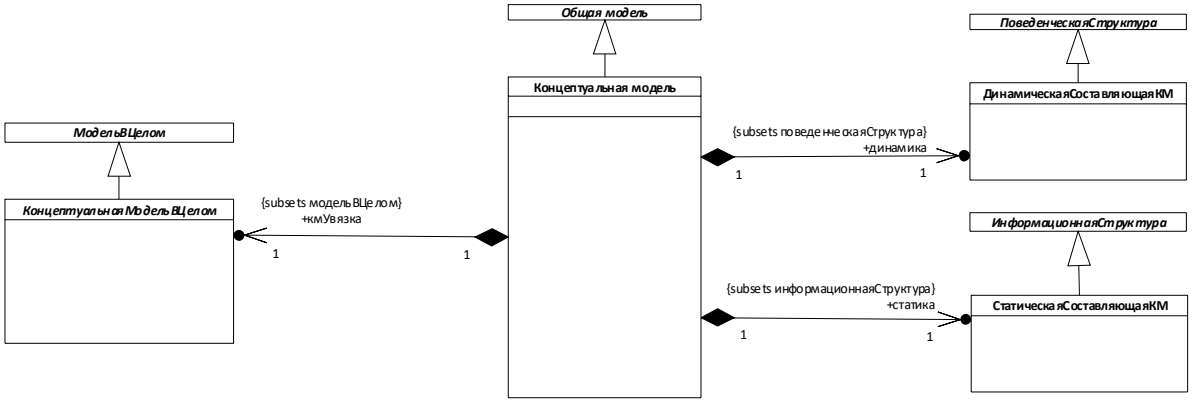


Рис. Г.5 Диаграмма классов концептульной модели

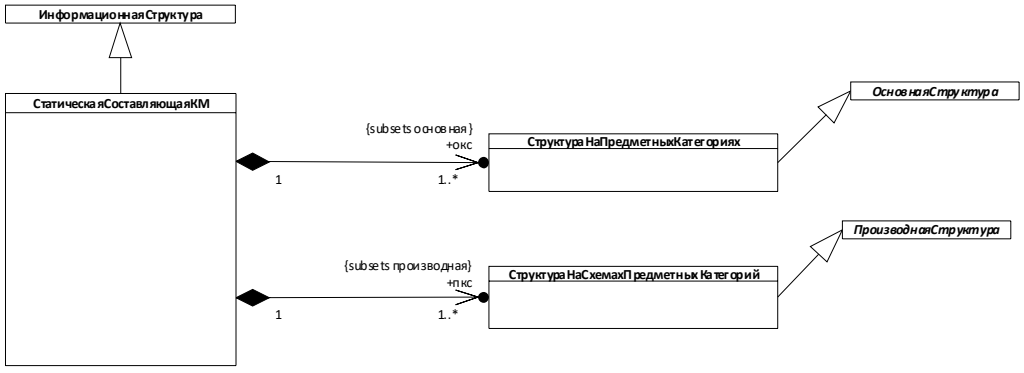


Рис. Г.6 Диаграмма классов статической составляющей концептульной модели

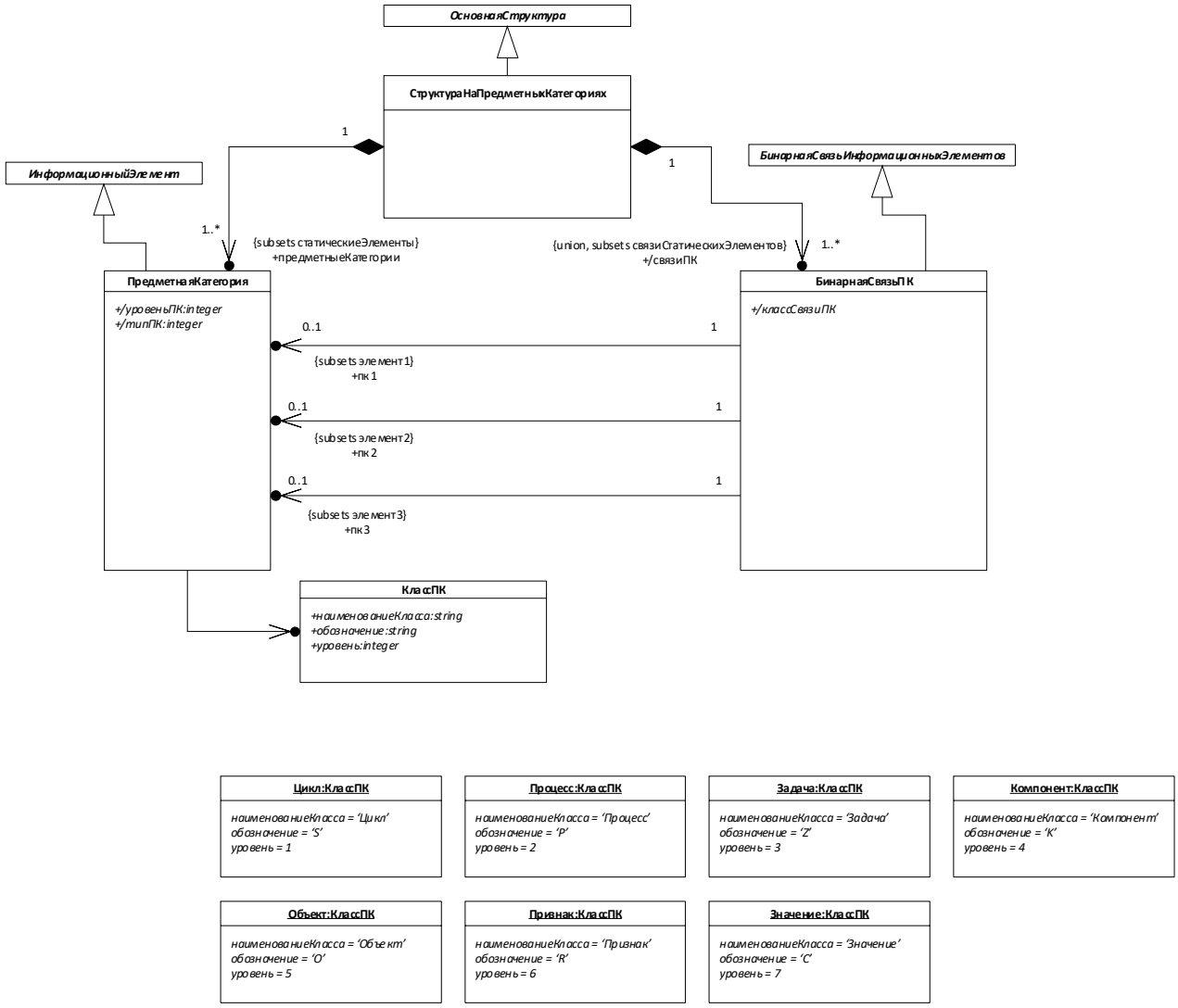


Рис. Г.7 Диаграмма классов основной концептуальной структуры

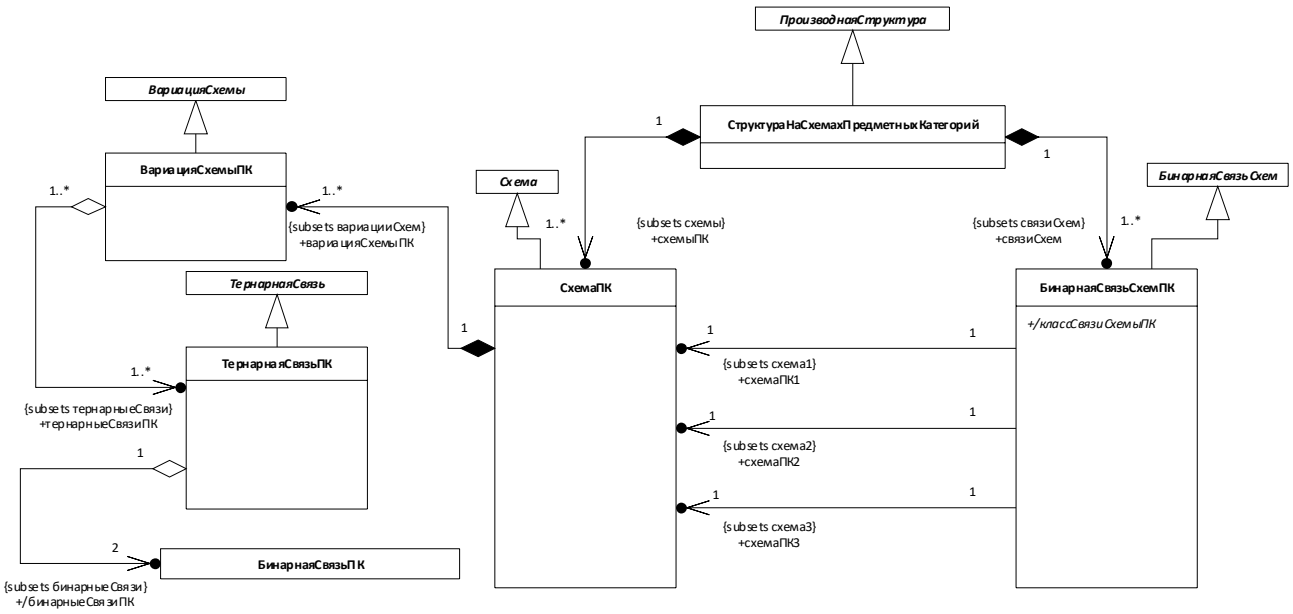


Рис. Г.8 Диаграмма классов производной концептуальной структуры

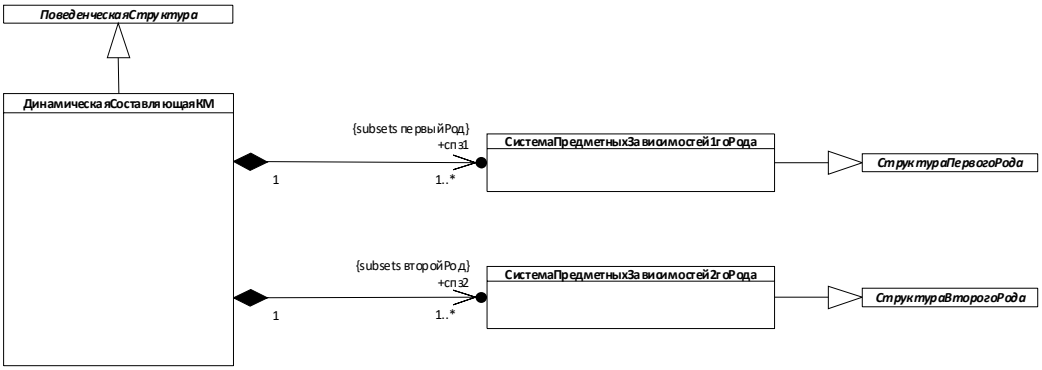


Рис. Г.9 Диаграмма классов динамической составляющей конецепуальной модели

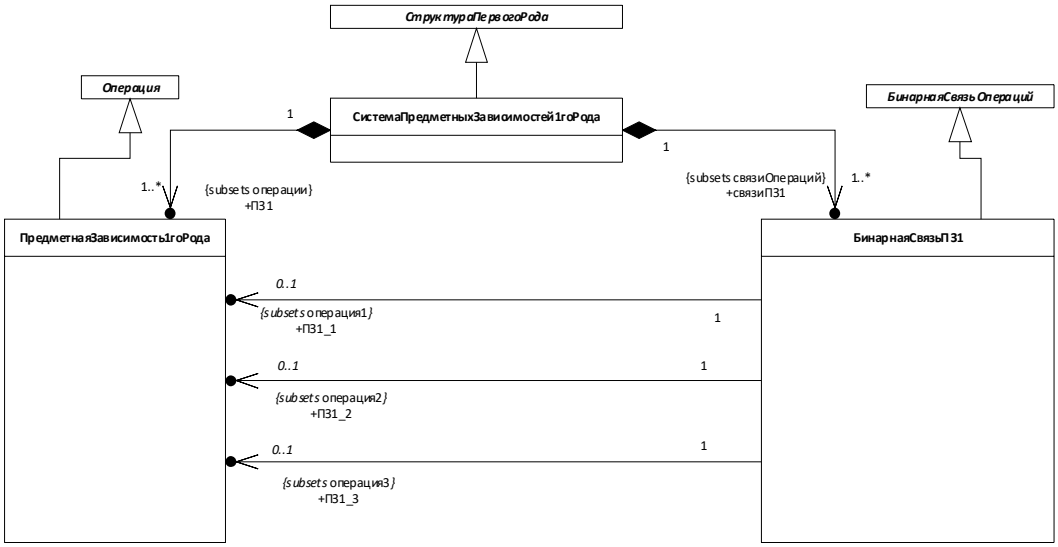


Рис. Г.10 Диаграмма классов системы предметных зависимостей певрого рода

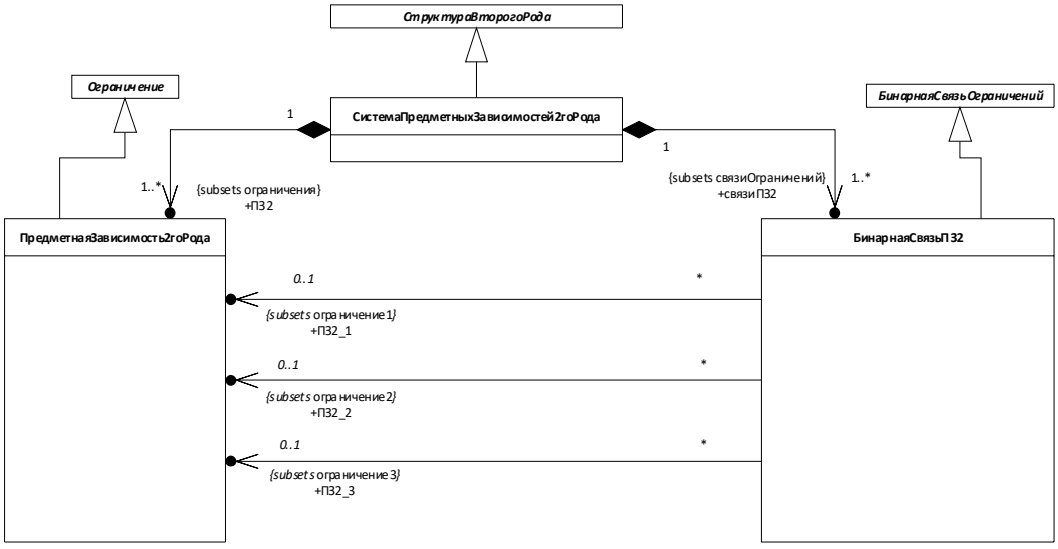


Рис. Г.11 Диаграмма классов системы предметных зависимостей второго рода

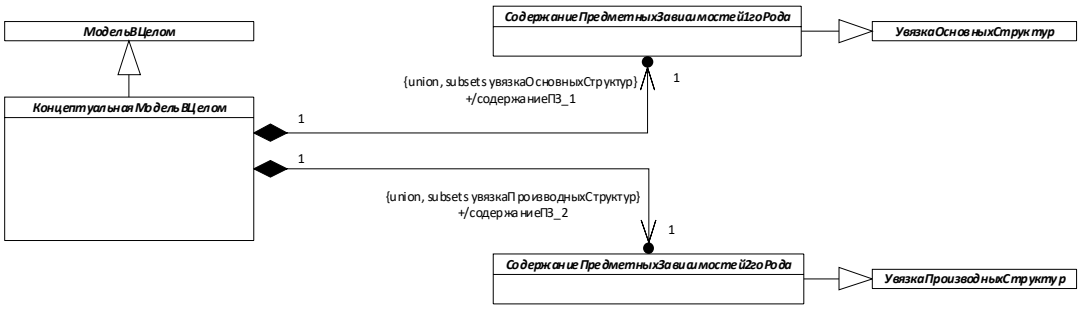


Рис. Г.12 Диаграмма классов составляющих модели в целом для концептуальной модели

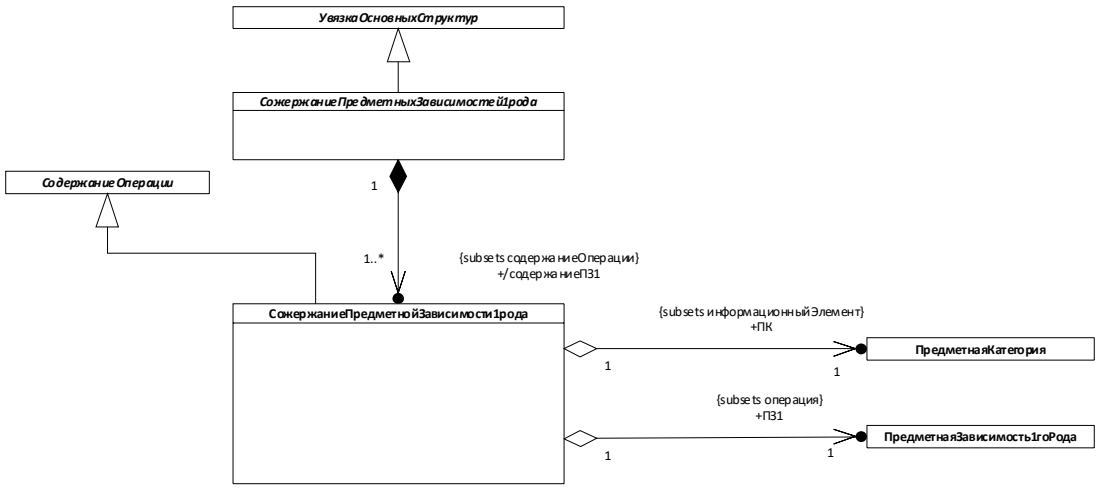


Рис. Г.13 Диаграмма классов содержательной структуры зависимостей первого рода

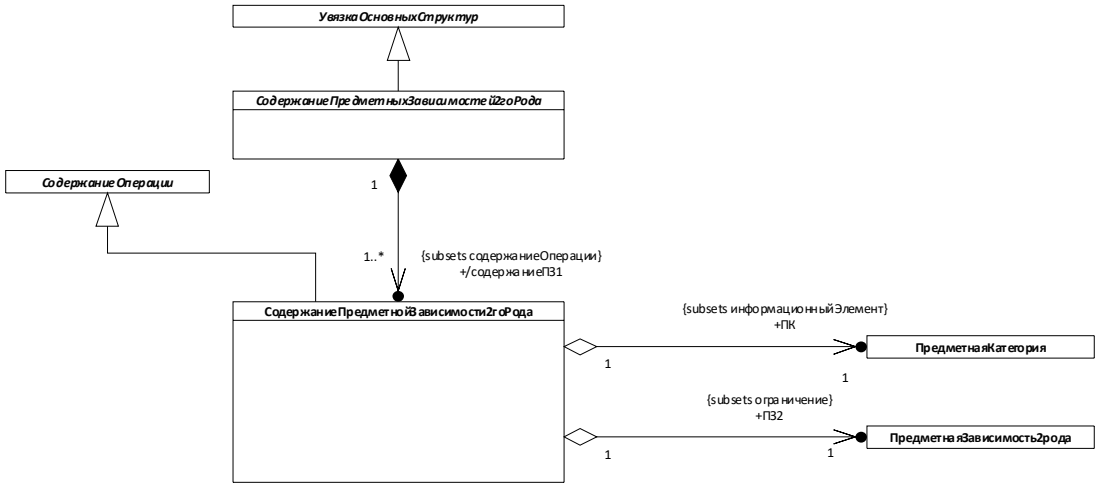


Рис. Г.14 Диаграмма классов содержательной структуры зависимостей второго рода

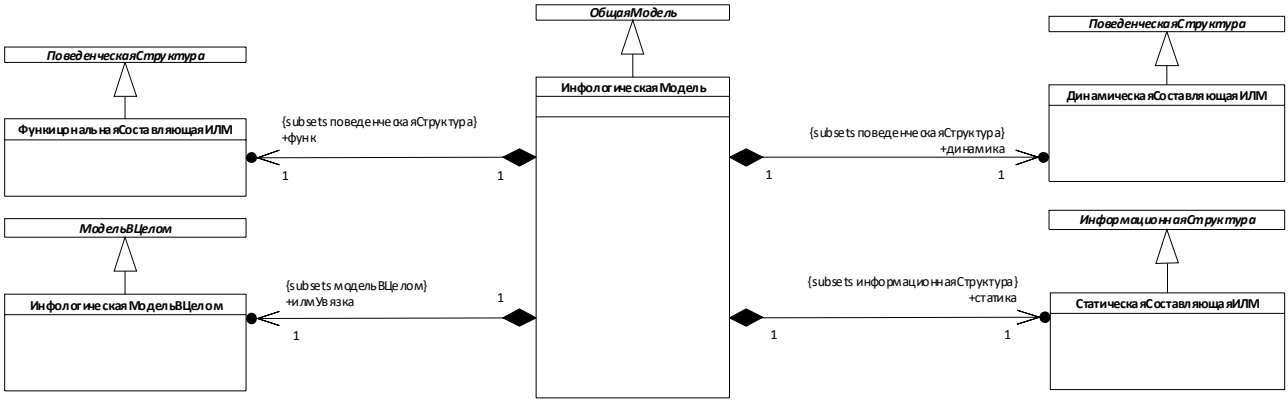


Рис. Г.15 Диаграмма классов инфологической модели

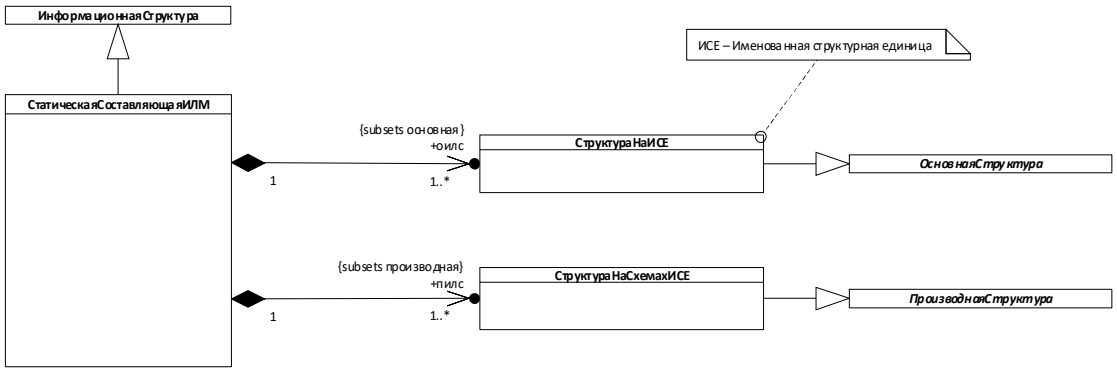


Рис. Г.16 Диаграмма классов статической составляющей инфологической модели

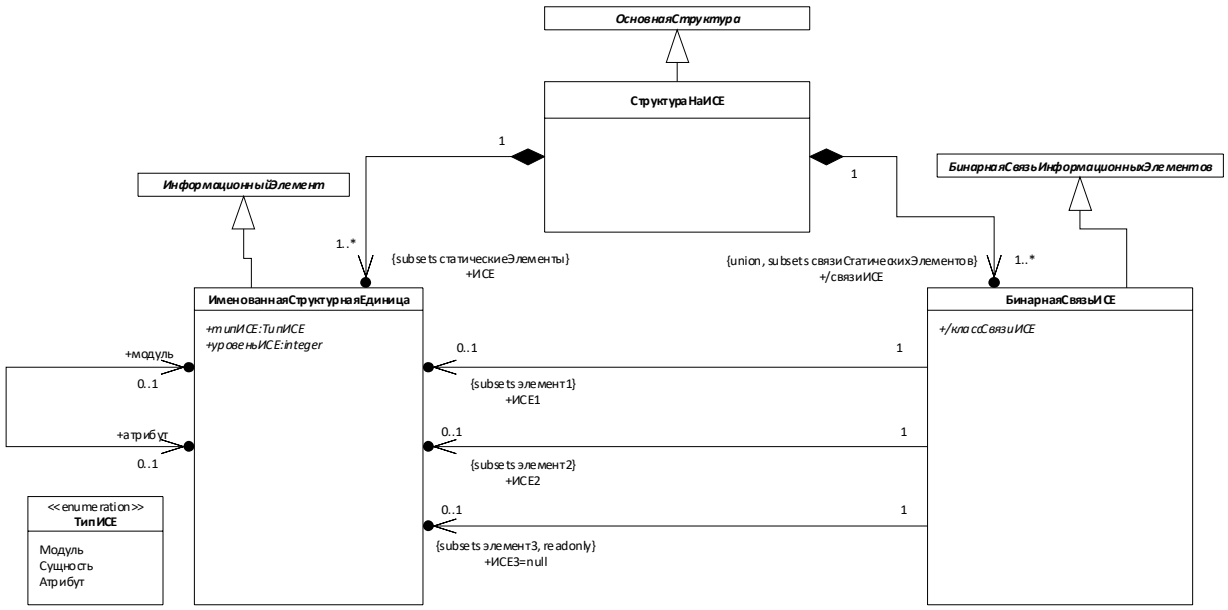


Рис. Г.17 Диаграмма классов основной инфологической структуры

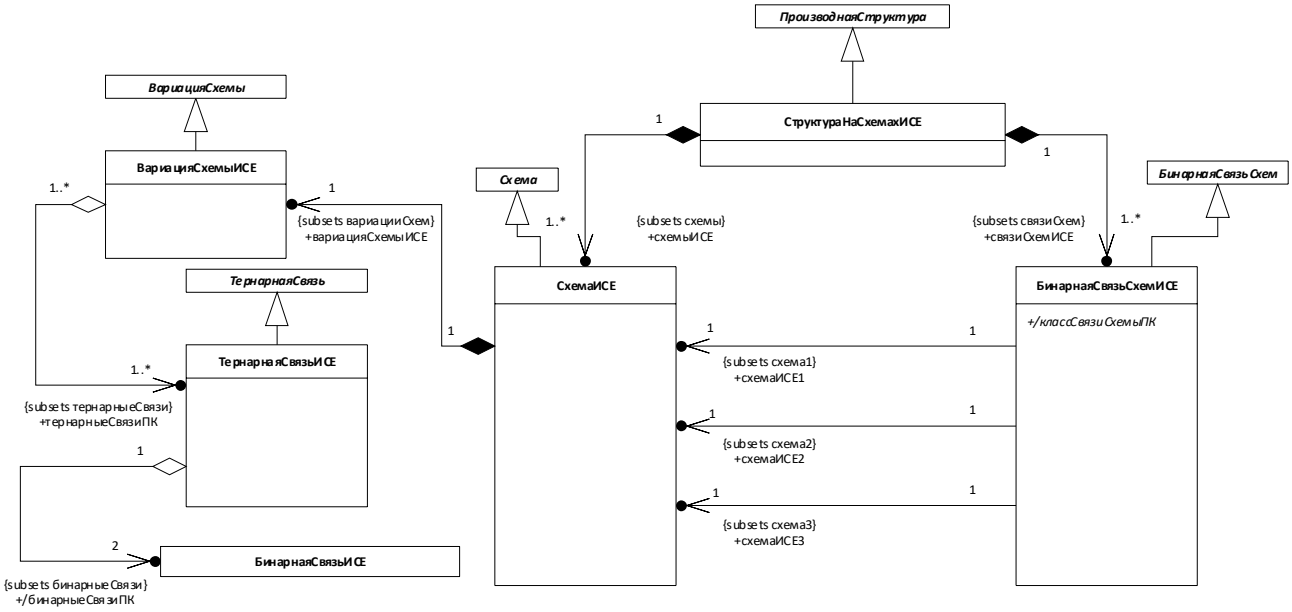


Рис. Г.18 Диаграмма классов производной инфологической структуры

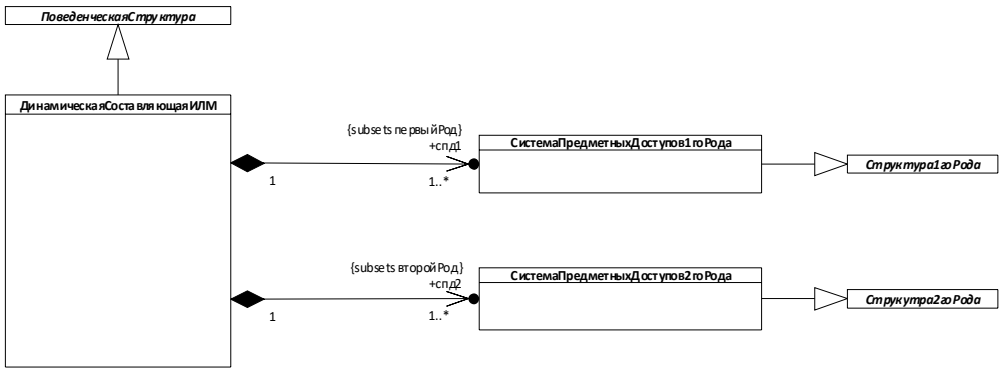


Рис. Г.19 Диаграмма классов динамической составляющей инфологичесокй модели

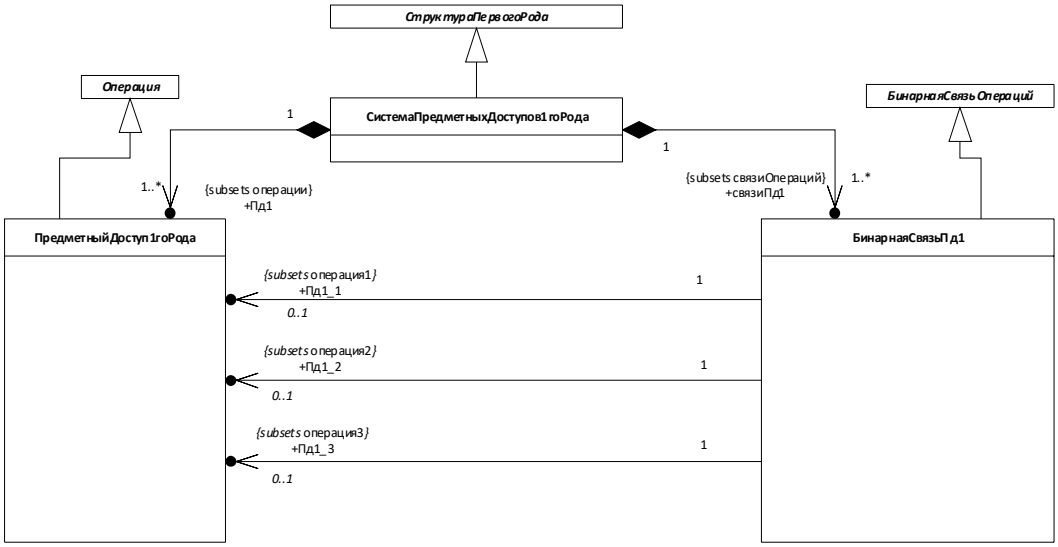


Рис. Г.20 Диаграмма классов системы предметных доступов первого рода

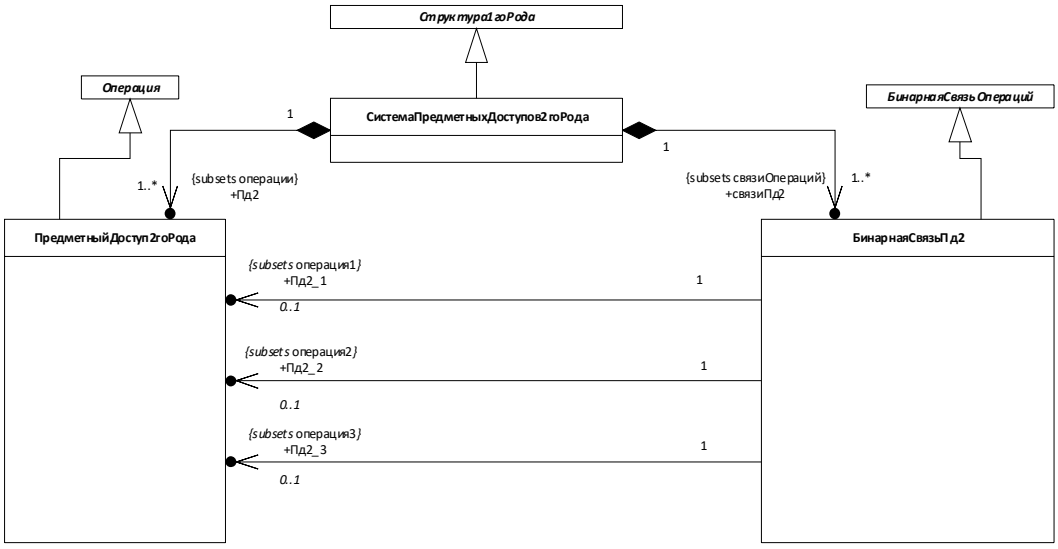


Рис. Г.21 Диаграмма классов системы предметных доступов второго рода

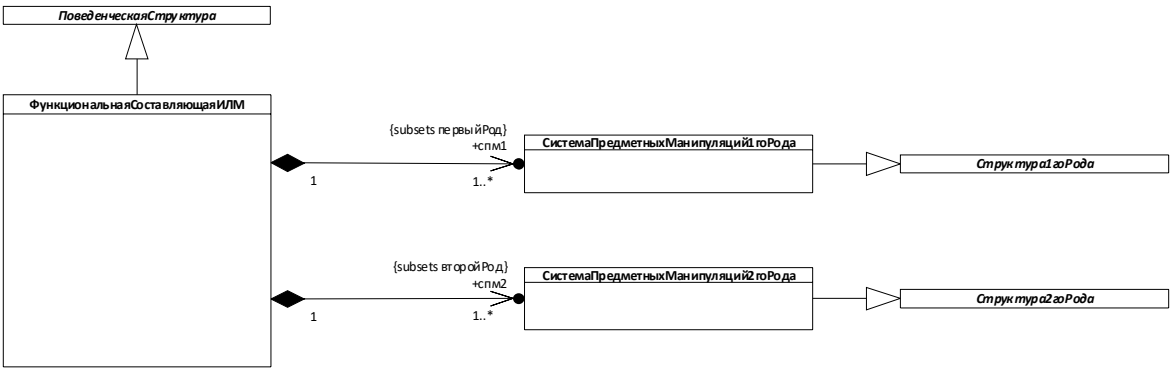


Рис. Г.22 Диаграмма классов функциональной составляющей инфологичсекой модели

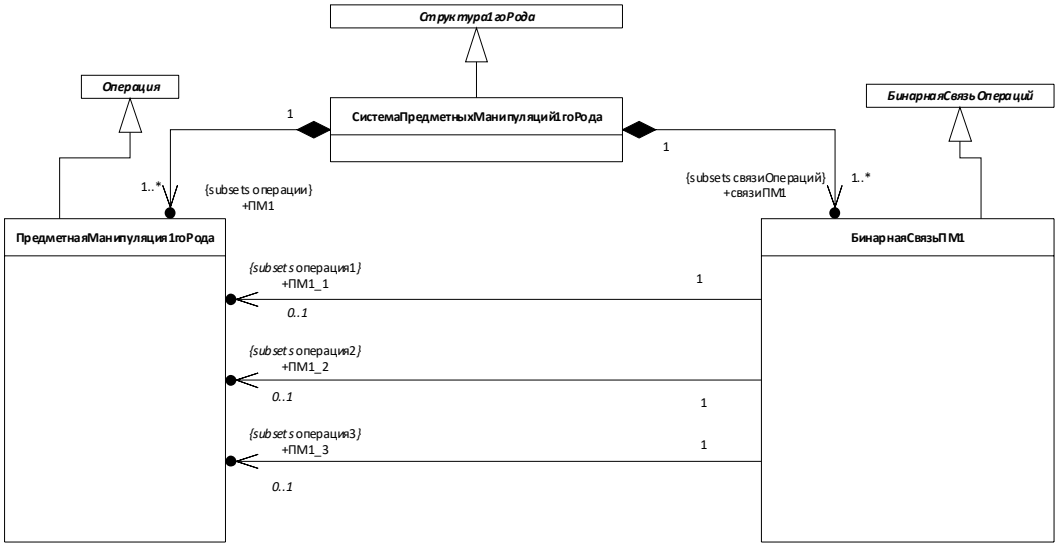


Рис. Г.23 Диаграмма классов системы предметных манипуляций первого рода

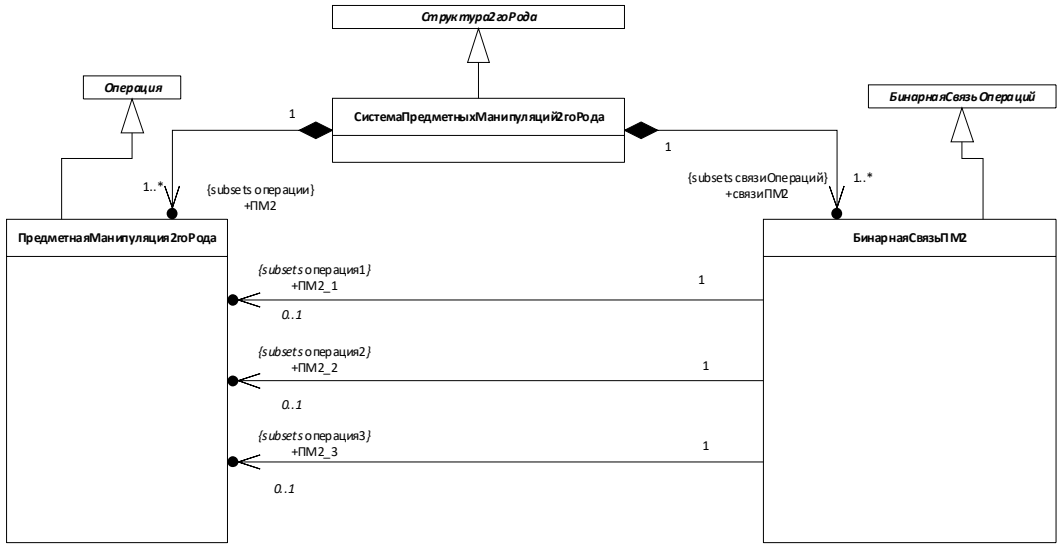


Рис. Г.24 Диаграмма классов системы предметных манипулдяций второго рода

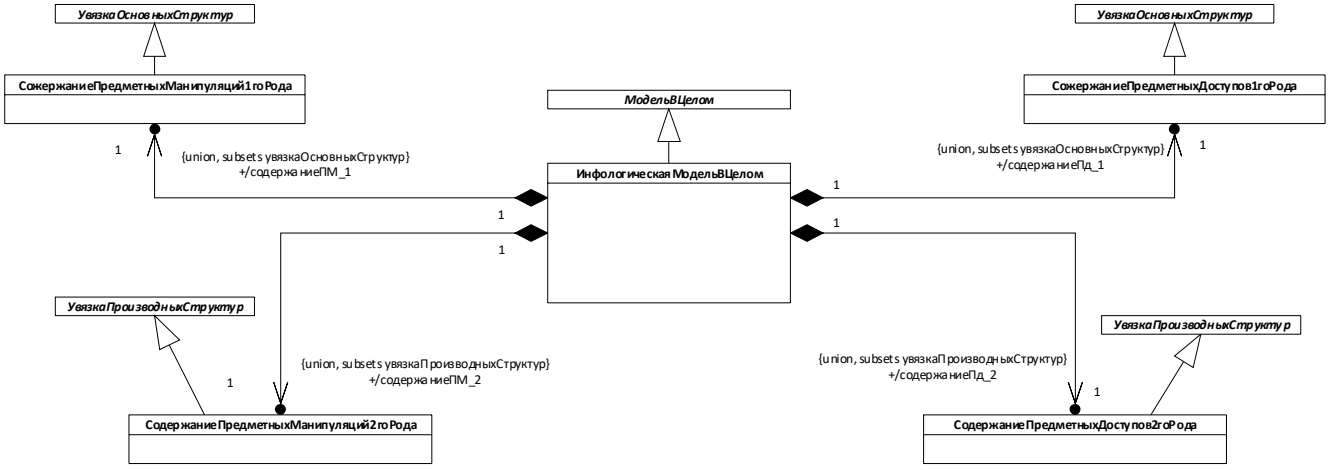


Рис. Г.25 Диаграмма классов модели в целом для инфологической модели

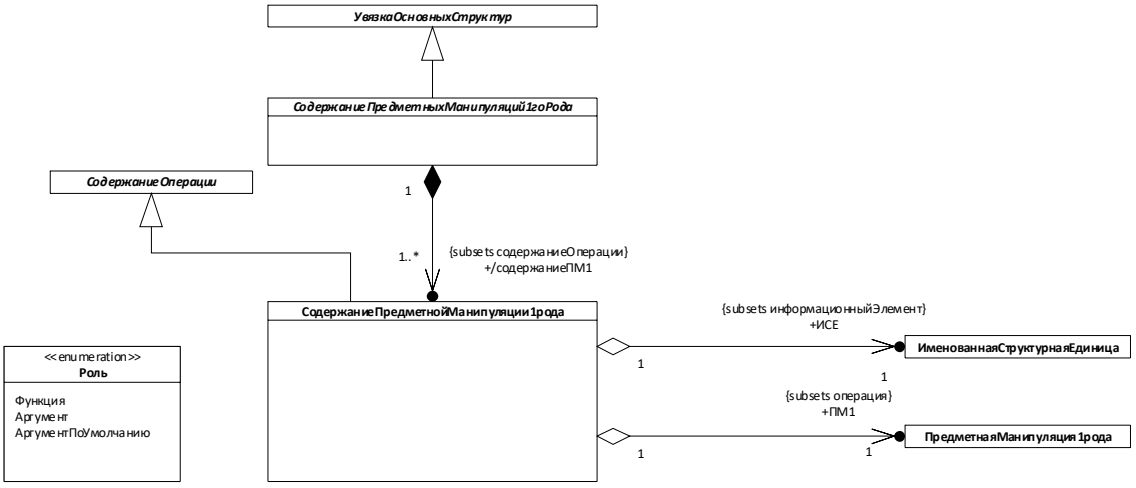


Рис. Г.26 Диаграмма классов содержательной структуры для манипуляций первого рода

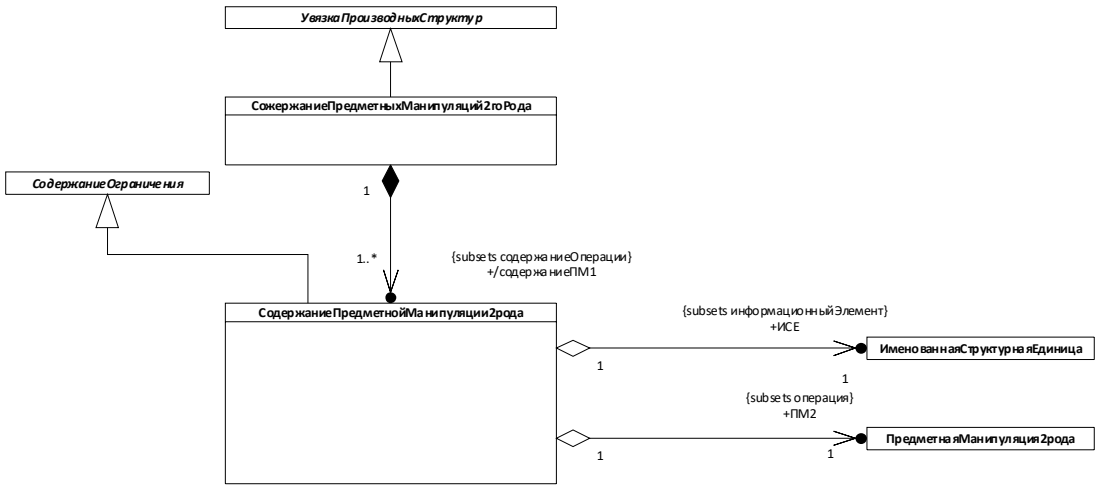


Рис. Г.27 Диаграмма классов содержательной структуры для манипуляций второго рода

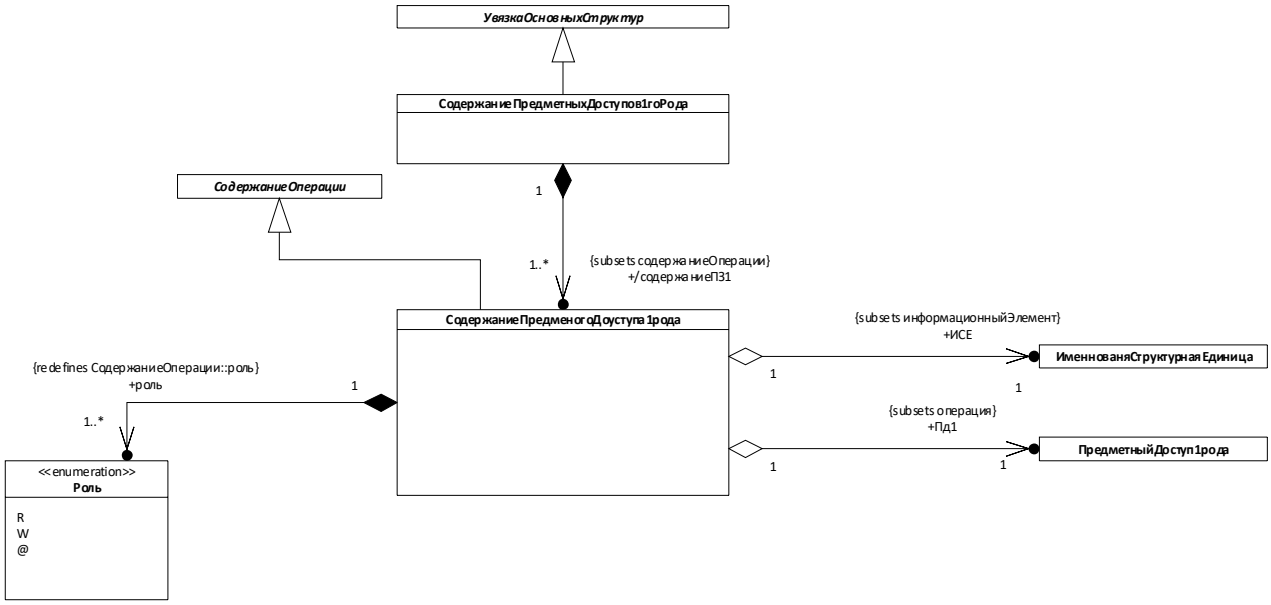


Рис. Г.28 Диаграмма классов содержательной структуры для доступов первого рода

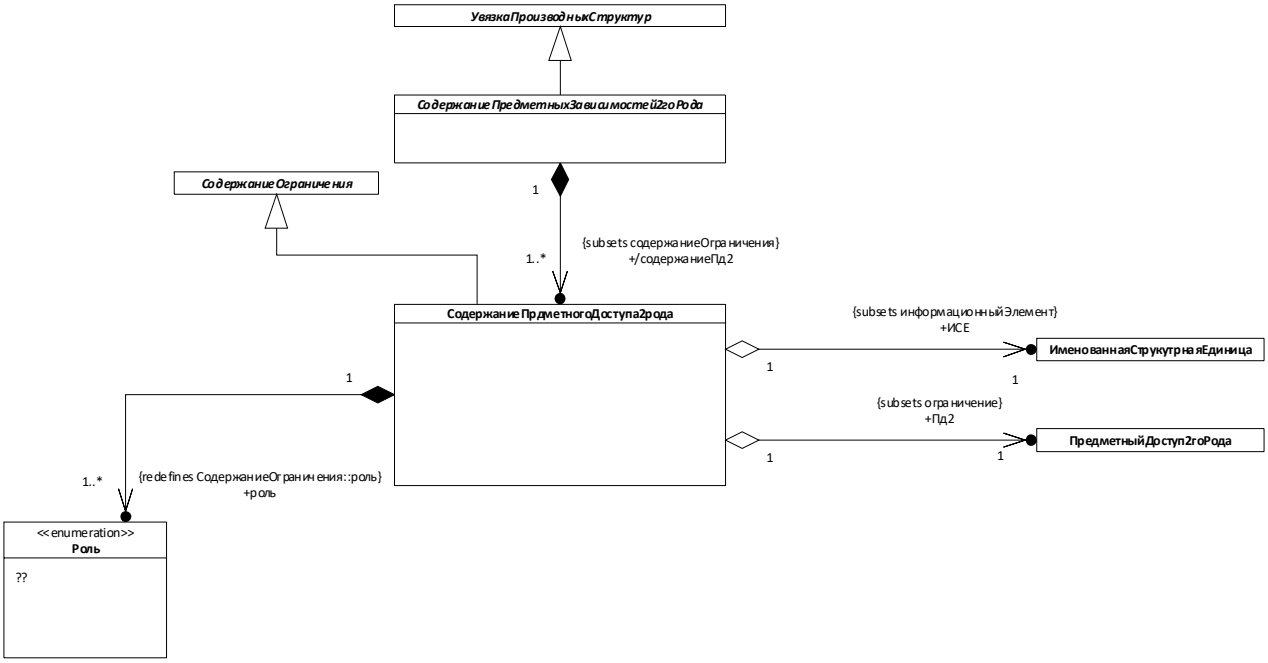


Рис. Г.29 Диаграмма классов содержательной структуры для доступов второго рода

Приложение Д

Свидетельства о государственной регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2017618975

Программный комплекс ИС-2-У

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (RU)***

Автор: ***Гаврилов Андрей Геннадьевич (RU)***

Заявка № **2017616041**
Дата поступления **23 июня 2017 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***14 августа 2017 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021618780

Модуль реструктуризации функциональной составляющей инфологической модели

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (RU)***

Авторы: ***Ружейников Антон Андреевич (RU), Гаврилов Андрей Геннадьевич (RU), Новоселова Ольга Вячеславовна (RU), Волкова Галина Дмитриевна (RU)***

Заявка № **2021615081**
Дата поступления **09 апреля 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***01 июня 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021616067

Модуль интеграции функциональной составляющей инфологической модели

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (RU)***

Авторы: ***Сидоров Антон Сергеевич (RU), Гаврилов Андрей Геннадьевич (RU), Новоселова Ольга Вячеславовна (RU), Волкова Галина Дмитриевна (RU)***

Заявка № **2021615061**
Дата поступления **09 апреля 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***15 апреля 2021 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021616013

Модуль реструктуризации динамической составляющей концептуальной модели

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (RU)***

Авторы: ***Ружейников Антон Андреевич (RU), Гаврилов Андрей Геннадьевич (RU), Новоселова Ольга Вячеславовна (RU), Волкова Галина Дмитриевна (RU)***

Заявка № **2021615065**
Дата поступления **09 апреля 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***15 апреля 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021616009

Модуль интеграции динамической составляющей концептуальной модели

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (RU)***

Авторы: ***Сидоров Антон Сергеевич (RU), Гаврилов Андрей Геннадьевич (RU), Новоселова Ольга Вячеславовна (RU), Волкова Галина Дмитриевна (RU)***

Заявка № **2021615079**
Дата поступления **09 апреля 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***15 апреля 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

Приложение Е

Справки об использовании результатов диссертационного исследования

ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ «СПЕЦИАЛЬНАЯ ИНТЕГРАЦИЯ»
125040, г. Москва, ул. Нижняя, д.14 стр. 1, т/ф +7(495) 215-54-35, info_si@specint.ru,
ОКПО 29101911, ОГРН 1147746196510, ИНН 7714928855, КПП 771401001

СПРАВКА
об использовании результатов диссертационного исследования
Гаврилова Андрея Геннадьевича

Результаты диссертационного исследования Гаврилова Андрея Геннадьевича на тему: «Разработка метода моделирования и средств поддержки управления развитием визуальной интегрированной среды проектирования автоматизированных систем», представленного на соискание ученой степени кандидата технических наук, использованы в практике деятельности ООО «Специальная интеграция» в виде:

1) практического применения программного комплекса «ИС-2», разработанного в процессе выполнения диссертационной работы, в деятельности компании, связанной с разработкой автоматизированных систем;
2) практического применения методик и средств развития программного комплекса «ИС-2» для разработки средств интеграции среды с программным обеспечением организации.

Формализация и систематизация предметной области на начальных этапах разработки информационных систем является трудоёмкой задачей из-за отсутствия универсального подхода к ее решению. Применение методик начального и концептуального моделирования, реализованных в программном комплексе «ИС-2», позволило повысить эффективность процесса формирования модели автоматизации задач. Применение визуальной технологии формирования моделей в интегрированной среде позволило сократить время, затрачиваемое на разработку моделей и снизить количество ошибок, возникающих в процессе моделирования. Использование методик, разработанных в результате диссертационного исследования и реализованных в средствах развития и конфигурирования программного комплекса, созданного в рамках диссертационного исследования, позволило выполнить разработку модуля интеграции с сервером баз данных. Этот модуль обеспечил автоматизированное отображение информационной структуры сформированной модели предметной задачи в структуру базы данных, уменьшив тем самым время разработки информационных систем.

Результаты диссертационного исследования имеют научную ценность и практическую направленность, дальнейшее развитие которых способствует превращению программного комплекса «ИС-2» в независимую среду разработки программного обеспечения и информационных систем, в частности.

Заместитель генерального директора А.В. Козлов

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО «МГТУ «СТАНКИН»)

СПРАВКА
об использовании результатов диссертационного исследования

Результаты диссертационной работы Гаврилова Андрея Геннадьевича на тему «Разработка метода моделирования и средств поддержки управления развитием визуальной интегрированной среды проектирования автоматизированных систем» – программный комплекс «ИС-2», программные модули его расширения и методическая документация были использованы в учебной и научной деятельности Федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технологический университет «СТАНКИН». В частности:

1. При выполнении инициативного проекта при поддержке Российского Фонда Фундаментальных Исследований № 13–07–00752 «Создание интегрированной среды поддержки проектирования и реализации прикладных автоматизированных систем на основе методологии автоматизации интеллектуального труда».
2. При выполнении инициативного проекта при поддержке Российского Фонда Фундаментальных Исследований № 17–29–07056 «Разработка моделей и методов представления и обработки проблемно-ориентированных знаний, извлекаемых из научно-технических текстов и конструкторско-технологической документации».
3. При выполнении инициативного проекта при поддержке Российского Фонда Фундаментальных Исследований № 17–29–07057 «Разработка методов реструктуризации и интеграции для семантических и синтаксических представлений при создании систем автоматизации процессов проектирования и управления».
4. При подготовке обучающихся, в качестве учебно-проектного средства поддержки дисциплин:
   - бакалавров по направлению 09.03.01 «Информатика и вычислительная техника», профили подготовки «Модели, методы и программное обеспечение анализа проектных решений» и «Программное обеспечение средств вычислительной техники и автоматизированных систем»;
   - магистров по направлению 09.04.01 «Информатика и вычислительная техника», профиль подготовки «Управление программными продуктами и проектами»;
   - аспирантов по направлению 09.06.01«Информатика и вычислительная техника», профиль подготовки «Системный анализ, управление и обработка информации».

Проректор по образовательной деятельности и молодежной политике ФГБОУ ВО «МГТУ «СТАНКИН», канд. техн. наук, доцент _____ /Бильчук Мария Викторовна/

«11» 04 2022 г.