Министерство науки и высшего образования Российской Федерации

Томский государственный университет
систем управления и радиоэлектроники

В.Г. Резник

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

Учебное пособие

Тема 1. Введение в предметную область дисциплины

Томск
2023

УДК 004.8 (004.9)
ББК 65.2-5-05

Резник, Виталий Григорьевич

Проектирование информационных систем. Тема 1. Введение в предметную область дисциплины. Учебное пособие / В.Г. Резник. – Томск : Томск. гос. ун-т систем упр. и радиоэлектроники, 2023. – 64 с.

Учебное пособие предназначено для обучения дисциплине «Проектирование информационных систем» для студентов направлений подготовки бакалавриата: 09.03.01 — «Информатика и вычислительная техника» и 09.03.03 — «Прикладная информатика».

Одобрено на заседании каф. АСУ протокол №____ от _____

УДК 004.8 (004.9)
ББК 65.2-5-05

Оглавление

1 ВВЕДЕНИЕ В ПРЕДМЕТНУЮ ОБЛАСТЬ ДИСЦИПЛИНЫ

В процессе обучения по выбранному направлению подготовки и её специализации (09.03.01 или 09.03.03) студент изучал множество дисциплин, каждая из которых имеет свою предметную область, целевые установки, методы и методики применения теории в будущей практической деятельности. На заключительном этапе обучения, студент должен успешно пройти производственную практику, написать и защитить выпускную квалификационную работу (ВКР). В указанном целевом аспекте дисциплина «Проектирование информационных систем» (ПИС) суммирует в себе знания, практическое применение которых будет характеризовать и оценивать степень профессиональной подготовки бакалавра.

В заявленном целевом аспекте дисциплины, первый раздел данного учебного пособия содержит не только наиболее важные термины и определения, касающиеся изучаемой предметной области, но также обсуждает различные их толкования, обусловленные соответствующими целевыми представлениями о предмете проектирования.

В идеале считается, что студент, проходя производственную практику, выбирает тему будущей ВКР, изучает предметную область объекта проектирования и проводит проектные работы, опираясь на знания, изложенные в данном учебном пособии. Свои проектные решения студент согласовывает с требованиями руководителя практики, а итоговые результаты производственной практики использует как исходный материал для будущей ВКР. Такой подход считается оптимальной целевой установкой студента в условиях строгого ограничения по времени периода написания и защиты ВКР.

В методическом плане, данный раздел разбит на шесть частей, последовательно излагающих заявленный выше учебный материал.

Первые четыре подраздела с различных точек зрения тематики раздела раскрывают ключевое базовое понятие «Информационная система» (ИС):

а) *как начальную парадигму* развития вычислительных систем;

б) *как новую парадигму* проектирования сложных программных систем;

в) *как часть общего системного понятия* «Автоматизированная система управления» (АСУ);

г) *как парадигму* CALS-технологий.

Пятый подраздел обсуждает непосредственные вопросы проектирования ИС, привязывая её предметную область к объектному пространству АСУ.

Шестой подраздел описывает конкретную учебную инфраструктуру проектировщика ИС, привязывая её к требованиям изучаемой дисциплины.

1.1 ИС как результат развития вычислительных систем

Словарь по информационным технологиям, оформленный как документ **ГОСТ 33707-2016** «Информационные технологии (ИТ). Словарь» на базе стандартах ISO/IEC 2382:2015, введённый в действие с 01 сентября 2017 года, содержит следующий набор определений [1]: « ... 4.451 **информационная база**: Набор экземпляров типов, соответствующих друг другу и информационной модели, принадлежащей экземпляру рассматриваемой предметной области.

П р и м е ч а н и е — Информационная база может либо не может быть пригодной для компьютерной обработки. Например, её не следует считать пригодной для компьютерной обработки, если она имеет форму рукописного документа. С другой стороны, если она задана в виде базы данных или компьютерного файла, то её следует считать пригодной для компьютерной обработки и, следовательно, её можно также называть объектной базой.

4.452 **информационная система**: Система, организующая обработку информации о предметной области и её хранение.

4.453 **информационный анализ**: Изучение документов и определение объёма формируемой и используемой информации, а также разработка схемы документооборота и модели информационных связей.

4.454 **информационный бит**: Бит, используемый для представления данных пользователя, отличных от целей управления.

4.455 **информационный объект**; ИО: Совокупность данных и программного кода, обладающая свойствами (атрибутами) и методами, позволяющими определённым образом обрабатывать данные. Самостоятельная единица применения и хранения в ИИС.

4.456 **информационный поиск**: Процесс нахождения и выбора (выдачи) требуемой (т. е. определённой заранее заданными признаками) информации из отдельного текста, документа, совокупности документов или вообще из запоминающего устройства любой физической природы.

4.457 **информация (в области обработки информации)**: Любые данные, представленные в электронной форме, написанные на бумаге, высказанные на совещании или находящиеся на любом другом носителе, используемые финансовым учреждением для принятия решений, перемещения денежных средств, установления ставок, предоставления ссуд, обработки операций и т. п., включая компоненты программного обеспечения системы обработки».

В дальнейшем мы будем использовать эти определения как базовые, дополняя их более конкретным семантическим содержанием. Но главное, что должен помнить студент, приведённые определения являются нормативными требованиями для специалистов его направления подготовки и могут быть необязательными или непонятными для специалистов других направлений и профилей обучения.

В процессе прохождения производственной практики студенту будет необходимо общаться с сотрудниками предприятия, которые проходили обучение в разные периоды времени, а их мышление и терминология тесно связаны с их непосредственной производственной деятельностью. Студенту будет необходимо не только собирать нужные сведения о своём объекте проектирования, но и объяснять свою позицию и точку зрения по этому вопросу. А чтобы успешно осуществлять указанную деятельность, студент должен уверенно разбираться в истории вопроса, который мы далее рассмотрим в двух аспектах:

а) *проблемы становления* информационных технологий;

б) *проблемы концепции* «*Программа-массив*».

1.1.1 Становление информационных технологий

На ранних этапах развития вычислительной техники, она использовалась исключительно для организации вычислений [2]. Вместо термина «*Информация*» повсеместно использовался термин «*Данные*», которые с точки зрения разработчика и пользователя программного обеспечения интерпретировались как входные (исходные) и выходные (результат вычислений) данные программы.

Сами данные типизировались (форматировались) как в аспектах аппаратного обеспечения вычислительных машин (ВМ), так и в аспектах языков программирования.

С появлением операционных систем (ОС) стали формироваться объекты, называемые «*Файловые системы*» (ФС). Первоначально они были сильно привязаны к конкретной реализации ОС, но со временем стали стандартизироваться и превратились в «очевидное» простейшее хранилище данных. ОС UNIX даже имеет свою базовую парадигму - «*Все есть файл*», подчёркивая фундаментальное значение этих систем.

С появлением цифровых средств связи, стало возможным как прямое соединение компьютеров, так и объединение их в компьютерные сети. Появилась возможность распределённой обработки данных, включая сетевые файловые системы. Появилось обобщённое понятие «*Системы обработки данных*» (СОД), которое в словаре [1] определяется как:

«4.1270 **система обработки данных:** Система, выполняющая автоматизированную обработку данных и включающая аппаратные средства, программное обеспечение и соответствующий персонал».

В учебном пособии [3] анализируется классификация СОД, предложенная А.М. Ларионовым, С.А. Майоровым и Г.И. Новиковым. Она делит СОД на две группы: *сосредоточенные* и *распределенные* системы.

В пределах изучаемой дисциплины, студент должен хорошо представлять себе классификацию *сосредоточенных систем*, например, в следующей интерпретации:

а) **ЭВМ** — согласованный в функциональном назначении аппаратный конструктив, дополненный базовым ПО ОС до конкретной виртуальной машины; в последующем, такая машина рассматривается как элемент более сложных систем;

б) **Вычислительные системы** — все прикладные аспекты программного обеспечения, реализованные поверх виртуальной машины ОС; в последующем, такие системы также могут рассматривается как элементы более сложных систем;

в) **Вычислительные комплексы** — системы аппаратных средств, ориентированные на повышение вычислительной мощности и надёжности всего конструктива, а также — дополненные соответствующей виртуальной машиной ОС; в последующем, такой комплекс также может рассматривается как отдельная ЭВМ или составляющая часть вычислительной системы.

Как альтернатива, дополняющая группу сосредоточенных систем, в СОД имеется *класс распределенных систем*, включающий в себя «*Системы телеобработки*» и «*Вычислительные сети*».

Система телеобработки — *вычислительная система* дополненная аппаратными и программными *средствами независимого удалённого доступа* к ней многими пользователями по отдельным каналам связи.

Вычислительная сеть — *удалённые друг от друга вычислительные системы*, которые объединены между собой программными и аппаратными средствами сетевой связи. В словаре [1, п. 223] этот термин определяется как: «Сеть, узлы которой состоят из компьютеров и аппаратуры передачи данных, а ветви которой являются линиями передачи данных».

Примечание — Обратим внимание, что на этапе становления информационных технологий, термин «*Информация*» понимался как некоторое свойство документов, объектов искусства и литературы доступное для восприятия и обработки только человеком. Поэтому, студенту в общении с сотрудниками предприятий, не имеющими должной подготовки в области информационных технологий, не следует использовать термин «Информация» в силу неверной трактовки его собеседниками. Следует использовать общедоступные термины дисциплин «Вычислительная математика», «Программирование», «Операционные системы», «Сети и телекоммуникации».

1.1.2 Парадигма «Программа-массив»

Начальный этап становления информационных технологий характеризуется:

а) *централизованной организацией* вычислительных процессов на ВМ мейнфреймов, которая ограничивает доступ разработчиков ПО к необходимым им вычислительным ресурсам;

б) *слабыми вычислительными ресурсами* вычислительных систем, включая быстродействие процессоров, размеры оперативной и внешней памяти, состава системного и инструментального программного обеспечения.

В условиях указанных ограничений, программное обеспечение разрабатывалось и отлаживалось достаточно долго, а насущная потребность в таких вычислительных приложениях вынуждало программистов использовать парадигму «Программа-массив».

Суть парадигмы «Программа-массив» — *сосредоточение основного внимания* алгоритму решения задачи, *оставляя второстепенное внимание* исходным данным. Более того, формат исходных данных и вывод результата выбирался под реализацию самого алгоритма, что делало их форматы непригодными для автоматической обработки другими программами.

Последствием такого подхода стал кризис в индустрии разработки ПО вызванный выбросом на рынок множества несовместимых между собой программ, решающих достаточно много узкоспециализированных задач, попытки объединения которых порождали потребность создания ещё большего количества программ, преобразующих форматы входных и выходных данных других программ.

С целью более качественного обоснования изложенных выше утверждений, проведём наглядную демонстрацию указанной ситуации двумя примерами.

Пример 1.1. На рисунке 1.1 показаны две программы **A** и **B**, которые реализуют два совместимых алгоритма общей задачи, но не являются совместимыми между собой по структуре и форматам входных и выходных данных. Такая ситуация требует разработки ещё двух программ **AB** и **BA**, обеспечивающих совместимость входных и выходных данных.

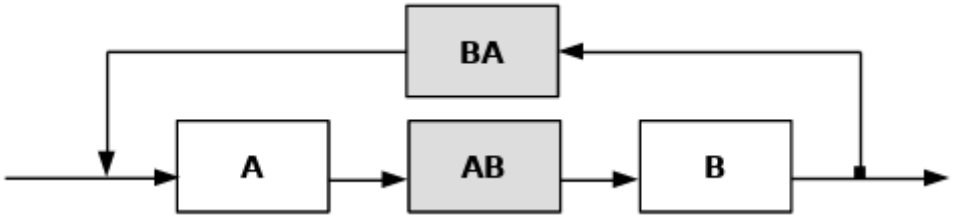

Рисунок 1.1 — Связь программ для Примера 1.1

Пример 1.2. На рисунке 1.2 показаны пять программ, имеющих между собой двунаправленные (дуплексные) связи с попарно несовместимыми интерфейсами.

Вопрос: Сколько дуплексных связей будет, если таких программ — **N**, и сколько нужно написать новых программ, чтобы эта система была программно совместима?

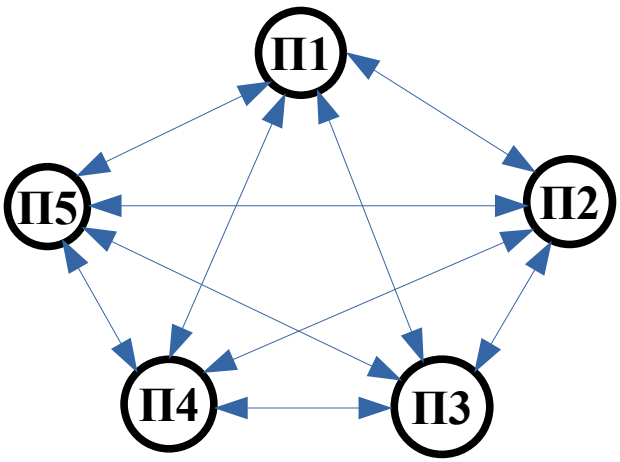

Рисунок 1.2 — Связь программ для Примера 1.2

Решение данного примера является достаточно простым. Действительно, если обозначить через $N_{св}$ и $N_{пр}$ *количество связей* и *новых программ*, то получаем расчётные формулы:

$$N_{св} = N*(N-1)/2; \quad (1.1)$$

$$N_{пр} = N*(N-1) = 2*N_{св}. \quad (1.2)$$

Таким образом, историческое развитие вычислительных систем, порождает проблему парадигмы «Программа-массив», что приводит к «взрывному» росту количества отдельных программных элементов и требует новых подходов, обеспечивающих более широкое использование средств вычислительной техники.

Как альтернатива чисто вычислительного подхода, выдвигающего на первое место реализацию алгоритмов и порождающего проблему парадигмы «Программа-массив», выступает ***информационный подход,*** породивший парадигмы информационного проектирования.

Цель информационного подхода — устранение недостатков парадигмы «Программа-массив» за счёт проектирования модели предметной области, что должно устранить необходимость разработки алгоритмов и реализацию дополнительных программ, занимающихся исключительно индивидуальным преобразованием данных для базовых программ, обеспечивающих общую целевую функциональность создаваемых систем.

Весь последующий учебный материал данного раздела фактически и посвящён различным аспектам указанной альтернативе — *информационному подходу проектирования ИС*.

1.2 ИС как парадигма информационного подхода

Суммируя итоги предыдущего подраздела, можно смело утверждать, что, с увеличением вычислительных ресурсов ВМ и сложности решаемых задач, парадигма «*Программа-массив*» породила три проблемы:

а) ***сложное взаимодействие*** большого количества программ;

б) ***сложность подготовки и ввода*** большого количества исходных данных;

в) ***сложность сохранения*** результатов расчётов и последующую их обработку.

Указанные три проблемы стимулировали бурное развитие парадигмы информационного подхода, которая стала распространяться как «*Технология проектирования предметной области*», явно выделяя фактор преобладания значимости типизации и хранения данных над значимостью алгоритма.

Парадигма информационного подхода активно пропагандировала централизованное хранение данных в универсальных форматах, что обеспечивалось ресурсами научных, университетских и производственных вычислительных центров, обещая разработчикам ПО единые интерфейсы предоставления исходных данных и результатов вычислений.

Новая парадигма стимулировала развитие нового научного направления — «*Информатика*» (*Informatique* или *Computer science*). В университетах открываются соответствующие направления подготовки студентов, появляются специалисты ИТ-технологий, термин «Информация» получает указанный в предыдущем подразделе смысл.

Обработка информации (данных) — [1]: «Совокупность операций, связанных с хранением, поиском, анализом, оценкой, воспроизведением информации с целью представления её в виде данных, удобных для использования потребителями».

Система обработки информации (СОИ) — [1]: «Совокупность технических средств и программного обеспечения, а также методов обработки информации и действий персонала, обеспечивающая выполнение *автоматизированной обработки информации*».

Примечание — Студент, изучающий данную дисциплину, должен понимать, что сам термин «*Информация*» или его производные не должен заменять или подменять терминологию конкретных предметных областей, иначе это нанесёт серьёзный ущерб его проектной и производственной деятельности.

Со временем, парадигма информационного подхода, сформулированная как «*Технология проектирования предметной области*», утратила свой первоначальный смысл и разделилась на два направления: «*Технология описания предметной области*» и «*Технология универсального управления данными*».

1.2.1 Технологии описания предметной области

Технологии описания предметной области предполагают теоретическое изучение сложных социальных или производственных систем и последующее документирование этих описаний. Подобную деятельность и стали называть «*Проектированием*».

Очевидно, что такое проектирование сильно зависит от свойств самой предметной области, поэтому, рассматривая проект как систему, можно выделить только обобщённые его части, показанные на рисунке 1.3.

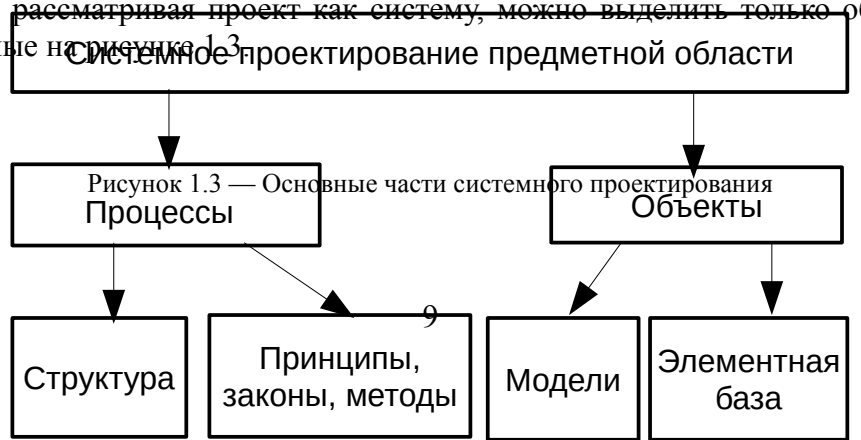

Рисунок 1.3 — Основные части системного проектирования

В общем случае, системное проектирование достаточно чётко выделяет *функциональный* и *объектный* аспекты описания предметной области.

В начале 1970-х годов, Дуглас Т. Росс стал пропагандировать методику «*Структурный анализ и проектирование*» (SADT, Structured Analysis and Design Technique), в которой предлагалось проводить функциональное описание систем в виде набора графических диаграмм.

В конце 1970-х годов, департамент военно-воздушных сил США реализовал идею SADT посредством стандартов **IDEF** (*ICAM Definitions*):

1) **IDEF0** — Function Modeling (Функциональное моделирование);
2) **IDEF1** — Information Modeling (Моделирование информационных потоков внутри системы);
3) **IDEF1X** — Data Modeling (Моделирование баз данных на основе модели «Сущность-связь»);
4) **IDEF2** — Simulation Model Design (Динамическое моделирование развития систем);
5) **IDEF3** — Process Description Capture (Документирование технологических процессов);
6) **IDEF4** — Object-Oriented Design (Проектирование объектных систем);
7) **IDEF5** — Ontology Description Captue (Проектирование онтологии систем);
8) **IDEF6** — Design Rational Capture (Проектирование обоснования проектных действий);
9) **IDEF7** — Information Auditing (Аудит информационных систем);
10) **IDEF8** — User Interface Modeling (Разработка интерфейсов пользователя);
11) **IDEF9** — Business Constraint Discovery method (Метод исследования бизнес-ограничений);
12) **IDEF10** — Implementation Architecture Modeling (Моделирование архитектуры выполнения);
13) **IDEF11** — Information Artifact Modeling (Информационное моделирование артефактов);
14) **IDEF12** — Organization Modeling (Организационное моделирование);
15) **IDEF13** — Three Schema Mapping Design (Трёхсхемное проектирование преобразования данных);
16) **IDEF14** — Network Design (Проектирование компьютерных сетей).

Не все из перечисленных стандартов получили должное признание и применение. Наиболее популярными из них являются: IDEF0, IDEF1X и IDEF3. Дополнительно, широкое распространение получила модель ***DFD*** или ***Data Flow Diagrams***.

> Студент должен хорошо осознавать, что *функциональный* и *объектный* аспекты описания предметной области понимаются в обобщённом смысле, а не в терминах языков программирования.

В середине 1990-х годов, группа сотрудников компании Rational Software (Гради Буч, Джеймс Рамбо и Ивар Якобсон) стали разрабатывать структурные методы описания предметной области программного обеспечения для языков ООП. В дальнейшем общее описание этих методов получило название **UML** (*Unified Modelling Language*) или «*Унифицированный язык моделирования*».

В 1996 году, прототип языка UML, был передан консорциуму **OMG** (*Object Management Group*), который ***в 1997 году*** выпустил спецификацию языка **UML 1.0**.

Далее:

а) *в июле 2005 года* была опубликована спецификация **UML 2.0**;

б) *в апреле 2012 года* были опубликованы стандарты ISO/IEC 19505-1:2012 и ISO/IEC 19505-2:2012.

Отметим, что все диаграммы языка UML сгруппированы в трёх частях, отражающих их следующее прикладное назначение.

Структурные диаграммы (*Structure diagrams*):

а) Диаграммы классов — *Class Diagrams*.

б) Диаграммы компонентов — *Component Diagrams*.

в) Диаграммы составной структуры — *Composite structure Diagrams*.

г) Диаграммы развёртывания — *Deployment Diagrams*.

д) Диаграммы объектов — *Object Diagrams*.

е) Диаграммы пакетов — *Package Diagrams*.

ж) Диаграммы профилей — *Profile Diagrams*.

Диаграммы поведения (*Behavior Diagrams*):

а) Диаграммы деятельности — *Activity Diagrams*.

б) Диаграммы состояний — *State Machine Diagrams*.

в) Диаграммы вариантов использования — *Use case Diagrams*.

Диаграммы взаимодействия (*Interaction Diagrams*):

а) Диаграммы коммуникаций — *Communication Diagrams*.

б) Диаграммы обзора взаимодействия — *Interaction Diagrams*.

в) Диаграммы последовательности — *Sequence Diagrams*.

г) Диаграммы синхронизации — *Timing Diagrams*.

> Примечание — **Важное свойство языка UML** — его диаграммы ориентированы на поддержку методологии «*Разработка, управляемая моделью*» (*Model Driven Development* или *MDD*). Суть этой методологии — генерация программного кода на основе построенных диаграмм. В частности, имеется свободный фреймворк ***Eclipse Modeling Framework***, который генерирует такой код, инструменты и прочие приложения на основе структурированной метамодели данных.

Таким образом, в настоящее время имеется достаточно большой набор средств для полного и качественного описания предметной области ИС.

1.2.2 Технологии универсального представления данных

Технологии универсального представления данных предполагают разработку универсальных средств хранения и манипулирования этими данными, а также разработку универсальных языков, обеспечивающих доступ и управление этими данными. Со временем, эти технологии сформировали своё собственное технологическое направление и стали называться **системами управления базами данных** (СУБД).

Сами данные размещались в специальных хранилищах, называемых **базами данных** (БД), а доступ к ним осуществлялся согласно *базовой модели структурного представления данных*:

а) **иерархическая модель** — модель, аналогичная архитектуре файловых систем ОС и удобная для отображения классов языков ООП;

б) **сетевая модель** — модель, ориентированная на отображение множественных семантических связей объектов;

в) **реляционная модель** — модель, хорошо ассоциируемая с таблицами представления данных.

Наибольшую популярность получила реляционная модель, поскольку она имеет достаточно простую интерпретацию в виде таблиц данных, достаточно развитый математический аппарат обработки данных и достаточно простой язык манипулирования данными — **SQL** (*Structured Query Language* — «Язык Структурированных Запросов»).

В случае централизованного хранения всех данных, проектирование простейших ИС сводится к следующим основным проектным решениям:

а) с помощью **ER-модели** (*ERM, Entity-Relationship Model*, *модель «сущность — связь»*) *описывается концептуальная схема предметной области*;

б) *проводится логическое проектирование*, которое заканчивается построением **даталогической модели**;

в) *выбирается СУБД* и *проводится* **физическое проектирование**, которое заканчивается построением схемы базы данных (*схемы БД*).

Примечание — Студент достаточно хорошо знаком со всеми аспектами работы с реляционными СУБД, пройдя обучение по дисциплине «Базы данных», поэтому более подробно эти вопросы в данной дисциплине не рассматриваются.

Уже отмеченный успех реляционной модели вполне обоснован, например, благодаря мощным возможностям языка SQL, практически отпала необходимость создания таких систем как «*Генераторы отчётов*», которые необходимы для ИС построенных на других архитектурах данных. Тем не менее, реляционные модели обладают рядом существенных недостатков, необходимых для понимания проектировщику ИС. И основным её недостатком является ***чувствительность к стабильности схемы БД***.

Формально, схема БД описывается *сценарием на DDL* (*Data Definition Language*) — языке описания данных конкретной СУБД. После запуска сценария или взаимодействия с СУБД в интерактивном режиме, схема БД создаётся и не может меняться в процессе эксплуатации БД.

Поскольку даталогическая модель БД проектируется на основе ERM-описания предметной области, то любые изменения в отношениях сущностей (*Unity*) предметной области приводят и к изменению схемы БД. Вариантами такого изменения отношений предметной области, например, могут быть:

а) *замена централизованной схемы хранения* и обработки данных на распределённую, предполагающую использование нескольких СУБД;

б) *внедрение готовых ИС* или обрабатывающих центров, имеющих свои оригинальные форматы представления данных.

Обратим внимание, что технологии универсального представления данных разрабатываются не только в рамках технологического направления СУБД. Все достаточно сложные программные системы создают свои специализированные представления данных, многие из которых становятся достаточно популярными и используются в ИС. В качестве примера приведём наборы офисных приложений, ставшие привычным дополнением к ПО ОС.

Типовой набор офисных приложений включает: *текстовый процессор*, *электронную таблицу*, *программу подготовки презентаций*, *систему управления базами данных*, *графическую программу* и *редактор формул*.

Первоначально, все эти программы имели оригинальные способы представления данных, сохраняемых в корпоративных форматах файлов. Последние версии этих программ используют стандартизированные способы представления данных, основанные на языке **XML** (*e**X**tensible **M**arkup **L**anguage*). Так:

а) корпорация Microsoft использует представления «*Microsoft Office Open XML*», соответствующие проекту ISO/IEC IS 29500:2008;

б) LibreOffice ориентируется на международный формат «*OpenDocument Format*», утверждённый стандартом ISO/IEC 26300:2006.

> Примечание — Приступая к прохождению производственной практики, студент должен хорошо понимать, что проектируемая им ИС во многом будет определяться ***структурой предметной области***, а важнейшим свойством этой структуры является «***Управление***».

Таким образом, парадигма информационного подхода ориентирована на создание инструментальных средств для тематики проектирования информационных систем.

1.3 ИС как подсистема АСУ

С конца 70-х годов (ещё во времена СССР), с целью внедрения передовых технологий в управление производством, создаётся серия стандартов **ГОСТ 24.xxx** — Система технической документации на АСУ (Единая система стандартов автоматизированных систем управления). Эти стандарты охватывали различные аспекты управления производством, а также определяли различные требования к самой системе управления:

а) **ГОСТ 24.101-80** — Виды и комплектность документов;

б) **ГОСТ 24.102-80** — Обозначение документов;

в) **ГОСТ 24.103-84** — Автоматизированные системы управления. Общие положения;

г) **ГОСТ 24.104-85** — Автоматизированные системы управления;

д) **ГОСТ 24.202-80** — Требования к содержанию документа «Технико-экономическое обоснование»;

е) **ГОСТ 24.203-80** — Требования к содержанию общесистемных документов;

ж) **ГОСТ 24.204-80** — Требования к содержанию документа «Описание постановки задачи»;

з) **ГОСТ 24.205-80** — Требования к содержанию документов по информационному обеспечению;

и) **ГОСТ 24.206-80** — Требования к содержанию документов по техническому обеспечению;

к) **ГОСТ 24.207-80** — Требования к содержанию документов по программному обеспечению;

л) **ГОСТ 24.208-80** — Требования к содержанию документов стадии «Ввод в эксплуатацию»;

м) **ГОСТ 24.209-80** — Требования к содержанию документов по организационному обеспечению;

н) **ГОСТ 24.210-82** — Требования к содержанию документов по функциональной части;

о) **ГОСТ 24.211-82** — Требования к содержанию документа «Описание алгоритма»;

п) **ГОСТ 24.301-80** — Общие требования к выполнению текстовых документов;

р) **ГОСТ 24.302-80** — Общие требования к выполнению схем;

с) **ГОСТ 24.304-82** — Требования к выполнению чертежей;

т) **ГОСТ 24.602-86** — Состав и содержание работ по стадиям;

у) **ГОСТ 24.703-85** — Типовые проектные решения. Основные положения.

В 90-е годы (после смены экономической модели государства), многие аспекты стандартов серии **24.xxx** стали неприемлемы для новых условий. Прежде всего расширилось само понятие термина «*Управление*», большую популярность стали приобретать стандарты серии **34.xxx** - «*Автоматизированные системы*» (АС). Многие авторы учебных пособий, например, Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Основы проектирования информационных систем. Учебное пособие. – СПб: Университет ИТМО, 2015. – 206 с. [4] стали интерпретировать ИС как АС, необоснованно ссылаясь на жизненные циклы (ЖЦ) АС и этапы его проектирования.

Сложившаяся ситуация не только запутывает и без того сложные вопросы проектирования ИС, но и явно искажает концептуальную суть создаваемых систем. Поэтому, важнейшая задача данного и следующего подразделов — формирование у студента правильных теоретических представлений о *предметной области изучаемой дисциплины*.

> Примечание — Студентам, изучающим автоматизированное проектирование, настоятельно рекомендуются использовать учебник [5]: Норенков, И.П. Основы автоматизированного проектирования: Учеб. для вузов. 4-е изд., перераб. и доп. - М.: Изд-во МГТУ им. Н.Э. Баумана, 2009. - 430 с.: ил. - (Сер. «Информатика в техническом университете»).

Чтобы более подробно раскрыть заявленную выше тему и устранить концептуальные противоречия во взглядах различных специалистов ИТ-технологий, рассмотрим понятия терминов АСУ и АС в трёх аспектах:

а) ***трёхуровневая архитектура АСУ*** — как базовая управляющая концепция автоматизации деятельности предприятия;

б) ***стандарты и компоненты АС*** — как нормативная база для проектирования автоматизированных систем;

в) ***стадии и этапы проектирования АС*** — как шаблон, в пределах которого осуществляется проектирование ИС.

Остальные уточняющие аспекты предметной области проектирования, касающиеся *жизненного цикла изделий* (ЖЦИ), будут рассмотрены в следующем подразделе.

1.3.1 Трёхуровневая архитектура АСУ

Плановая парадигма хозяйственной деятельности СССР (до 1991 года) естественным образом выводила *аспект управления* на главное место. В условиях такой парадигмы, результат внедрения средств вычислительной техники в промышленное или другое производство (организацию, предприятие) рассматривался как «*Автоматизированная система управления*» (АСУ), имеющая трёхуровневую архитектуру показанную ниже таблицей 1.1.

В плане развития такой архитектуры, уровни *стратегического управления* и *оперативного исполнения* считаются главными, постепенно поглощающими уровень *тактического управления*, поэтому сначала дадим характеристику именно им.

Уровень стратегического управления осуществляет административно-финансовое планирование, называемое АСУ производством или АСУП. Само управление рассматривается как разработка и последующая актуализация соответствующих планов, которые распространяются на нижние уровни АСУ.

В свою очередь, эти планы ограничиваются (управляются) *внешними требованиями государственного управления для юридических лиц*. Поскольку ограничения (условия) на административно-финансовую деятельность юридических лиц — достаточно формализованы, то разрабатываются и внедряются программные продукты известные как ERP, MRP и MRP2.

MRP (*Material Requirements Planning*) — система планирования потребностей в материалах, являющаяся наиболее популярной логистической концепцией.

MRP2 (*Manufacturing Resource Planning*) — система планирования производственных ресурсов, обеспечивающая как операционное, так и финансовое планирование производства.

ERP (*Enterprise Resource Planning*) — система планирования ресурсов предприятия, обеспечивающая организационную стратегию интеграции производства и операций, управление трудовыми ресурсами, финансового менеджмента и управления активами.

Таблица 1.1 — Трёхуровневая архитектура АСУ

Уровень управления	***Уровень АСУ***
Уровень стратегического управления	**АСУП** — «*АСУ производством*» или **ERP**-системы. Стратегическое административно-финансовое планирование и управление, решающее задачи: *что произвести*, *в каких объемах*, *к каким срокам*, *из чего* и прочее. **ERP** (*Enterprise Resource Planning*) — планирование ресурсов предприятия.
Уровень тактического управления	**АСУПП** — «*АСУ производственными процессами*» или «*АСУ подготовки производства*» или **MES**-системы**.** **Уровень** начальников производств, цеховых технологов, диспетчеров, мастеров, решающих задачу: *как произвести заданное*, *по каким технологиям*, *на каких станках*, *в каком порядке выполнять заказы*, чтобы минимизировать издержки и максимально эффективно использовать ресурсы. **MES** (*Manufacturing Execution System*) — система управления производственными процессами.
Уровень оперативного исполнения	**АСУТП** — «*АСУ технологическими процессами».* **Уровень** контроллерного управления, HMI с человеком исполнителем, **SCADA**-системы — решает задачи *поддержания заданных технологических параметров производственных процессов*. **SCADA** (*Supervisory Control And Data Acquisition*) — диспетчерское управление и сбор данных.

В настоящее время считается, что системы ERP полностью покрывают задачи, решаемые системами MRP и MRP2. Особо отметим, что к классу ERP-систем относится и продукция известной российской компании «**1С**».

Уровень оперативного исполнения – система технологических процессов или АСУТП, которая реализует планы, поступившие сверху иерархической цепочки управления. Считается, что поступающие планы преобразуются в программы для станков с ЧПУ (станков с числовым программным управлением) или в программы для роботизированных участков производства.

Высокая степень формализации среды данного уровня, обусловленная отсутствием человеческого фактора, создаёт условия и простор для реализации различных алгоритмов обработки информации. В перспективе — это системы SCADA.

SCADA (*Supervisory Control And Data Acquisition*) — системы диспетчерского управления и сбора данных, которые обеспечивают также в реальном масштабе времени *обработку*, *отображение* и *архивацию информации* об объекте мониторинга или управления.

В настоящее время диспетчерское управление характерно для *воздушного* и *железнодорожного транспорта*, в *химической*, *атомной* и других видах промышленности, где требуется критическое по времени и согласованное по взаимодействиям вмешательство, контроль и ответственность человека. В будущем предполагается, что SCADA-системы будут обычным явлением всех АСУТП, а человеку будет отведена только роль пассивного наблюдателя.

Уровень тактического управления включает в себя задачи, которые по тем или иным причинам оказались *недостаточно автоматизированы*, чтобы их можно было перенести на уровень АСУП или АСУТП.

Для этого уровня используются сокращения **АСУПП** (АСУ производственными процессами) или **MES** (Manufacturing Execution System), а некоторые авторы дают интерпретацию как «АСУ подготовки производства» или используют сокращение **АСТПП** (Автоматизированная система технологической подготовки производства).

Как отмечено выше, уровень АСУПП был намечен к последующему поглощению его другими уровнями, но действительность оказалась другой — задачи тактического управления плохо формализуются по причине большого многообразия производственных процессов, различных критериев их качественной оценки и экономической целесообразности их реализации.

Чтобы преодолеть указанные трудности, ***в 1992 году*** была создана международная организация **MESA** (*Manufacturing Execution System Association*) как ассоциация поставщиков и пользователей MES-систем. ***Цель ассоциации*** — информирование производителей о системах отслеживания выполнения заказов на производстве.

В 1997 году, MESA опубликовала «*Функциональную модель MES*», включающую одиннадцать функций:

1) Контроль состояния и распределение ресурсов (*RAS*, *Resource Allocation and Status*);
2) *Оперативное/Детальное планирование* (*ODS*, *Operations/Detail Scheduling*) — *удалена в модели 2004 года*;
3) Диспетчеризация производства (*DPU*, *Dispatching Production Units*) ;
4) *Управление документами* (*DOC, Document Control*) — *удалена в модели 2004 года*;
5) Сбор и хранение данных (*DCA*, *Data Collection/Acquisition*);
6) Управление персоналом (*LM*, *Labor Management*);
7) Управление качеством продукции (*QM*, *Quality Management*);
8) Управление производственными процессами (*PM*, *Process Management*);
9) *Управление производственными фондами (техобслуживание)* (*MM, Maintenance Management*) — *удалена в модели 2004 года*;
10) Отслеживание истории продукта (*PTG*, *Product Tracking and Genealogy*);
11) Анализ производительности (*PA*, *Performance Analysis*).

В 2001 году, ассоциация сменила своё название на ***Manufacturing Enterprise Solutions Association***, чтобы показать, что область интересов ассоциации включает всё программное обеспечение, используемое на производстве, обмен передовым опытом и инновационными идеями для распространения знаний о решениях в области оперативного управления производственными предприятиями.

В 2004 году, MESA представила новую модель из восьми функций — ***Collaborative Manufacturing Execution System*** (*c-MES*), удалив функции ODS, DOC и MM. Тем не менее и до настоящего времени, уровень MES-систем продолжает существовать и содержать задачи, требующие своей автоматизации.

> Примечание — Завершая теоретические рассуждения по данному пункту подраздела, отметим, что трёхуровневая модель АСУ является одной из важнейших основ классификации предметной области проектирования для всех организаций. На её основе производится и классификация создаваемых и уже используемых систем автоматизации предприятий.

Студенту следует хорошо усвоить назначение каждого из уровней управления АСУ, а также хорошо ориентироваться в используемой терминологии, сокращениях и типах задач, решаемых на каждом из них.

1.3.2 Стандарты и виды обеспечения АС

После 1991 года, смена политического курса России привела и к смене её экономического курса, в смысле отказа от планового управления хозяйственной деятельностью страны. Жёсткие плановые требования к предприятиям сменились конкурентной инициативой и внутреннее управление производственной деятельностью потеряло свой прежний смысл. На первое место стали выходить международные стандарты, а в российских ГОСТ стала широко применяться серия стандартов ***34.xxx***, относящаяся к научно-техническому направлению «Информационные технологии», где термин «Управление» удалён из названия типа системы, подразумевая достаточно широкое его толкование.

Познавательная цель данного пункта — дать максимально точное определение понятию ***АС*** («Автоматизированная система») и уточнить отличие этого термина от термина ***ИС*** («Информационная система»).

Достижение указанной цели осуществим ссылками на стандарты серии ГОСТ 34.xxx, понимая их наследственную связь с ГОСТ 24.xxx, что прямо указано в тексте многих из этих документов:

а) **ГОСТ 34.003-90** — Автоматизированные системы. Термины и определения [6];

б) **ГОСТ 34.201-89** — Виды, комплектность и обозначения документов, при создании автоматизированных систем;

в) **ГОСТ 34.320-96** — Концепции и терминология для концептуальной схемы и информационной базы;

г) **ГОСТ 34.321-96** — Система стандартов по базам данных. Эталонная модель управления [7];

д) **ГОСТ 34.601-90** — Автоматизированные системы. Стадии создания [8];

е) **ГОСТ 34.602-89** — Техническое задание на создание автоматизированной системы [9];

ж) **ГОСТ 34.603-92** — Виды испытаний автоматизированных систем.

Начнём с ГОСТ 34.003-90 [6], где в разделе «1. АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ. ОБЩИЕ ПОНЯТИЯ» находим два определения.

Автоматизированная система (АС) — [6]: «Система, состоящая из персонала и комплекса средств автоматизации его деятельности, реализующая информационную технологию выполнения установленных функций.

Примечания:

1. В зависимости от вида деятельности выделяют, например следующие виды АС: автоматизированные системы управления (АСУ), системы автоматизированного проектирования (САПР), автоматизированные системы научных исследований (АСНИ) и др.

2. В зависимости от вида управляемого объекта (процесса) АСУ делят, например, на АСУ технологическими процессами (АСУТП). АСУ предприятиями (АСУП) и т. д.».

Интегрированная автоматизированная система (ИАС) — [6]: «Совокупность двух или более взаимоувязанных АС, в которой функционирование одной из них зависит от результатов функционирования другой (других) так, что эту совокупность можно рассматривать как единую АС».

Примечание — Приведённые выдержки из источника [6] наглядно показывают наследственную преемственность ГОСТ 34.xxx от трёхуровневой архитектуры АСУ.

Примечание — Обратите внимание, что в явном виде упоминаются только два уровня управления: АСУП и АСУТП. Более того, для понятия АСУТП выделен даже целый раздел: «*7. АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ. ОСНОВНЫЕ ПОНЯТИЯ*».

К сожалению источник [6] не даёт прямое определение ИС, но в межгосударственном стандарте ГОСТ 34.321-96 [7], в разделе 4 «Требования к управлению данными», находим такое толкование.

Информационная система — [7]: «... это система, которая организует *процессы сбора*, *хранения* и *обработки информации о проблемной области*. Она может быть размещена на одной или нескольких компьютерных системах. Если информационная система размещена на нескольких компьютерных системах, то она будет рассматриваться как *распределенная информационная система*.

Данные поступают в информационную систему и исключаются из неё, и эти взаимодействия могут осуществляться или людьми, или процессами.

Управление данными в настоящем стандарте будет касаться организации и управления постоянными данными. Постоянные данные — это данные, которые хранятся в информационной системе в течение определённого периода времени. Система, которая выполняет функцию организации и управления постоянными данными, называется *системой управления данными*».

По мнению автора данного пособия, источник [7] несколько сужает понятие ИС, не раскрывая взаимодействие ИС с другими подсистемами АС. Чтобы устранить этот недостаток, обратимся к разделу «*2. ОСНОВНЫЕ КОМПОНЕНТЫ АВТОМАТИЗИРОВАННЫХ СИСТЕМ*» источника [6] (ГОСТ 34.003-90).

Пользователь АС — [6]: «Лицо, участвующее в функционировании АС или использующее результаты её функционирования».

Эксплуатационный персонал АС — Лица, поддерживающие функционирование АС.

Организационное обеспечение АС (**ОО** АС) — [6]: «Совокупность документов, устанавливающих организационную структуру, права и обязанности пользователей и эксплуатационного персонала АС в условиях функционирования, проверки и обеспечения работоспособности АС».

Методическое обеспечение АС (**МетО** АС) — [6]: «Совокупность документов, описывающих технологию функционирования АС, методы выбора и применения пользователями технологических приёмов для получения конкретных результатов при функционировании АС».

Техническое обеспечение АС (**ТО** АС) — [6]: «Совокупность всех технических средств, используемых при функционировании АС».

Математическое обеспечение АС (**МО** АС) — [6]: «Совокупность математических методов, моделей и алгоритмов, применённых в АС».

Программное обеспечение АС (**ПО** АС) — [6]: «Совокупность программ на носителях данных и программных документов, предназначенная для отладки, функционирования и проверки работоспособности АС».

Информационное обеспечение АС (**ИО** АС) — [6]: «Совокупность форм документов, классификаторов, нормативной базы и реализованных решений по объёмам, размещению и формам существования информации, применяемой в АС при её функционировании».

Лингвистическое обеспечение АС (**ЛО** АС) — [6]: «Совокупность средств и правил для формализации естественного языка, используемых при общении пользователей и эксплу-

атационного персонала АС с комплексом средств автоматизации при функционировании АС».

Правовое обеспечение АС (**ПравО** АС) — [6]: «Совокупность правовых норм, регламентирующих правовые отношения при функционировании АС и юридический статус результатов ее функционирования.

П р и м е ч а н и е. Правовое обеспечение реализуют в организационном обеспечении АС».

Эргономическое обеспечение АС (**ЭО** АС) — [6]: «Совокупность реализованных решений в АС по согласованию психологических, психофизиологических, антропометрических, физиологических характеристик и возможностей пользователей АС с техническими характеристиками комплекса средств автоматизации АС и параметрами рабочей среды на рабочих местах персонала АС».

Таким образом, мы с уверенностью можем утверждать, что **ИС** — это **ИО** АС, представляющая *одну из девяти обеспечивающих подсистем АС*, и в таком аспекте она и должна проектироваться.

Студент должен хорошо знать состав всех обеспечивающих подсистем АС, поскольку их перечень и требования к ним обязательно отражаются в документе «*Техническое задание*» (ТЗ) на проектируемую АС, а значит они должны быть отражены и в описании предметной области ИС.

Если учесть, что **ПравО** АС реализуется в **ОО** АС, то взаимодействие между этими подсистемами можно представить показанным ниже рисунком 1.4.

Если учесть, что все обеспечивающие подсистемы АС взаимодействуют между собой, то проектировщик ИС, учитывая каждое ограничение, которое АС накладывает на **ИО** АС, должен учитывать ещё семь ограничений, накладываемых другими обеспечивающими подсистемами.

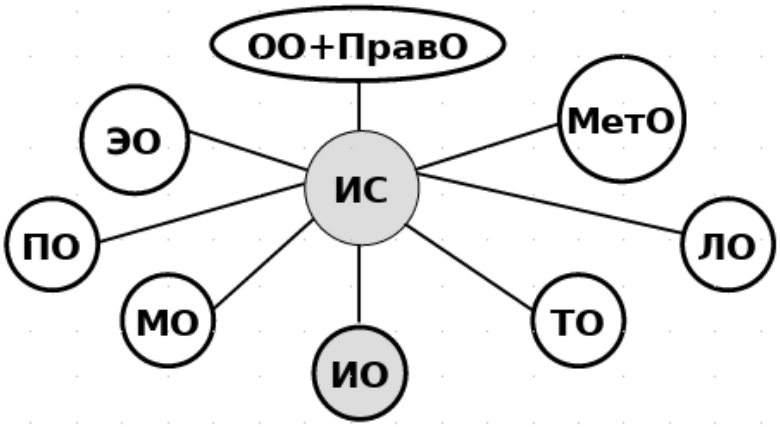

Рисунок 1.4 — Взаимодействие подсистем АС, отображаемых ИС

Примечание — Согласно стандартам АС, студент должен рассматривать и проектировать **ИС** как обеспечивающую подсистему АС (**ИО** АС), которая должна предоставлять всю необходимую информацию остальным обслуживающим подсистема АС.

С такой точки зрения, проектирование ИС напрямую связано со стадиями и этапами создания АС. И здесь уместно сделать три важных замечания, касающиеся обеспечивающих подсистем, проявляющихся в плане их приоритетных отношений:

а) ***в вычислительных системах***, основанных на парадигме «Программа-массив», *подсистема ИО* фактически не существовала как самостоятельное понятие, являясь вспомогательной частью подсистемы ПО;

б) ***в парадигме информационного подхода***, *подсистема ИО* приобретает своё самостоятельное смысловое значение и системные свойства; в совокупности с инструментальными средствами СУБД эти системные свойства формируют понятие информационной системы (ИС);

в) ***в парадигме АСУ***, *существенная часть подсистемы ИО* начинает обслуживать компоненту управления, которая сосредоточена в подсистемах *ОО АС* и *ПравО АС*.

1.3.3 Стадии и этапы создания АС

Познавательная цель данного пункта — дать общее представление о проектировании АС.

> Примечание — Поскольку ИС является обеспечивающей подсистемой АС, то она проектируется на основании требований, предъявляемых к ней АС, а значит — привязана к стадиям и этапам создания АС.

Все стадии и этапы создания АС определены ГОСТ 34.601-90 [8] и в обобщённом виде перечисляют все необходимые для выполнения работы и результирующий перечень документов. Вся эта информация о стадиях и этапах приведена ниже в таблице 1.2.

Стадия создания АС — [6]: «Одна из частей процесса создания АС, установленная нормативными документами и заканчивающаяся выпуском документации на АС, содержащей описание полной, в рамках заданных требований, модели АС на заданном для данной стадии уровне, или изготовлением несерийных компонентов АС, или приёмкой АС в промышленную эксплуатацию».

Этап создания АС — [6]: «Часть стадии создания АС, выделенная по соображениям единства характера работ и (или) завершающего результата или специализации исполнителей».

Очередь АС — [6]: «Часть АС, для которой в техническом задании на создание АС в целом установлены отдельные сроки ввода и набор реализуемых функций».

Безусловно, проектировщик ИС должен досконально знать все стадии и этапы создания АС, но его профессиональный интерес сосредоточен не на всех стадиях одинаково. Поэтому в таблице 1.2 выделим группы стадий, как это показано в таблице 1.3, а затем дадим качественную характеристику каждой группе.

Группа стадий «До ТЗ» — комплекс работ, по завершению которых принимается проектное решение — «*Будет создаваться АС или нет*».

Весь комплекс работ выполняется двумя группами заинтересованных лиц:

а) ***Заказчик*** — организация (предприятие), для которой предполагается создание АС;

б) ***Потенциальные исполнители*** — проектные организации, которые конкурируют за право создания АС, предлагая свои «Технико-коммерческие предложения» (ТКП).

На основании конкурса ТКП и принимается основное проектное решение о создании АС. Если такое решение является положительным, то определяются:

а) головной исполнитель;

б) список частных исполнителей (субподрядчиков);

в) общая стоимость создания АС.

Таблица 1.2 — Стадии и этапы создания АС [8]

Стадии	***Этапы работ***
1. Формирование требований к АС.	1.1. Обследование объекта и обоснование необходимости создания АС. 1.2. Формирование требований пользователя к АС. 1.3. Оформление отчёта о выполненной работе и заявки на разработку АС (тактико-технического задания).
2. Разработка концепции АС.	2.1. Изучение объекта. 2.2. Проведение необходимых научно-исследовательских работ. 2.3. Разработка вариантов концепции АС, удовлетворяющего требованиям пользователя. 2.4. Оформление отчёта о выполненной работе.
3. Техническое задание.	Разработка и утверждение технического задания на создание АС.
4. Эскизный проект.	4.1. Разработка предварительных проектных решений по системе и ее частям. 4.2. Разработка документации на АС и её части.
5. Технический проект.	5.1. Разработка проектных решений по системе и её частям. 5.2. Разработка документации на АС и её части. 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и (или) технических требований (технических заданий) на их разработку. 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации.
6. Рабочая документация.	6.1. Разработка рабочей документации на систему и её части. 6.2. Разработка или адаптация программ.
7. Ввод в действие.	7.1. Подготовка объекта автоматизации к вводу АС в действие. 7.2. Подготовка персонала. 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями). 7.4. Строительно-монтажные работы. 7.5. Пусконаладочные работы. 7.6. Проведение предварительных испытаний. 7.7. Проведение опытной эксплуатации. 7.8. Проведение приёмочных испытаний.
8. Сопровождение АС.	8.1. Выполнение работ в соответствии с гарантийными обязательствами. 8.2. Послегарантийное обслуживание.

Таблица 1.3 — Группы стадий создания АС

Группа стадий	***Список стадий создания АС***
До ТЗ	1-Формирование требований к АС 2-Разработка концепции АС
Подписание ТЗ	3-Техническое задание
Исполнение системы	4-Эскизный проект 5-Технический проект 6-Рабочая документация (Рабочий проект)
Завершение работ	7-Ввод в действие 8-Сопровождение АС

Примечание — Для студента знание стадий и этапов работ, входящих в группу стадий «До ТЗ», является *важным в методологическом плане знанием*, поскольку ему придётся фактически повторять эти работы во время проектирования ИС.

Группа стадий «Подписание ТЗ» — соответствует стадии «*Техническое задание*» (**ТЗ**), в процессе которой создаются, согласовываются и утверждаются два документа:

а) ***ТЗ на АС*** согласно ГОСТ 34.602-89 [9], где описываются все требования ко всем обслуживающим подсистемам АС, а также перечисляются последующие стадии создания АС;

б) ***Договор на создание АС***, в котором перечислены стадии работ по созданию АС, их стоимость, условия оплаты выполненных работ и порядок разрешения конфликтов сторон.

На стадии «*Техническое задание*» разрешается:

а) ***исключить*** стадию «*Эскизный проект*»;

б) ***объединить*** стадии «*Технический проект*» и «*Рабочая документация*» в одну стадию «*Технорабочий проект*».

После завершения стадии «*Подписание ТЗ*», ТЗ является тем документом, на основании которого оценивается результат проектирования ИС, поэтому, для проектировщика ИС, — это основной источник информации.

Группа стадий «Исполнение системы» — комплекс работ, которые должен выполнить головной исполнитель (*Исполнитель*).

Как отмечено выше, ГОСТ 34.602.89 допускает различное количество стадий этой группы, но они закреплены в требованиях ТЗ и Договоре на создание АС.

Исполнитель, без согласования с Заказчиком, но на основании внутренних распоряжений, может:

а) ***делить*** стадии и этапы работ в пределах ограничений основного ТЗ;

б) ***формировать*** «*Частные технические задания*» (ЧТЗ) на подсистемы АС и программное обеспечение.

Проектировщик ИС должен:

а) определить все подсистемы АС, выделенные **Исполнителем**;

б) подготовить и утвердить у **Исполнителя** необходимые ЧТЗ на ИС;

в) разработать и передать **Исполнителю** Рабочую Документацию на ИС.

После завершения группы стадий «Исполнение системы», Исполнитель передаёт Заказчику:

а) технические и программные средства АС;

б) рабочую документацию на АС.

Группа стадий «Завершение работ» — комплекс работ, которые Исполнитель должен выполнить в соответствии с требованиями ТЗ и условиями Договора.

Примечание — Обычно, на группе стадий «*Завершение работ*», Исполнитель выступает в качестве «Наблюдателя» или «Консультанта».

Завершая тематику данного подраздела, отметим, что стандарты ГОСТ серий 24.ххх и 34.ххх описывают предприятия и организации, продукция которых не меняется или мало меняется со временем. В таких условиях, все проектные решения, касающиеся как АС, так и ИС, будут долгое время сохранять свою актуальность.

С другой стороны в данном подразделе не рассматривалась сама продукция, ради создания которой предприятие и функционирует. Продукция предприятия, даже если потреб-

ность в её производстве вполне обоснована, требует проектирования, чтобы удовлетворять современным условиям. К тому же современный динамичный мир предъявляет к проектным решениям дополнительные требования, которые усиливаются условиями конкурентной борьбы.

Все сказанное выше также требует рассмотрения и анализа деятельности предприятия с позиции создания самой продукции, подчинив этой цели управленческую составляющую его функционирования.

> Примечание — Следует заметить, что как АС, так и ИС, являются результатом соответствующей производственной деятельности, поэтому они могут рассматриваться как изделия субъекта, названного **Исполнителем**.

Таким образом, в современных представлениях, ***ИС*** должна рассматриваться как изделие, входящее составной частью в изделие ***АС***, на которые распространяются общие требования теории и практики создания изделий промышленности.

1.4 ИС как парадигма CALS-технологий

В предыдущем подразделе ИС рассматривалась как подсистема АСУ (АС), отражая концептуальные представления, главенствующую роль в которых играет свойство — «*Управление*».

Познавательная цель данного подраздела — описать место ИС в структурном представлении предприятия (организации), когда в идею его архитектуры положен принцип «Жизненного цикла изделия» (ЖЦИ).

Инициатором появления и развития CALS-технологий считается Министерство обороны США, когда оно в 1985 году объявило о создании глобальной АС для электронного описания всех этапов проектирования, производства и эксплуатации продуктов военного назначения. В процессе развития идея CALS-технологий распространились на структуры НАТО, а затем — и на другие отрасли промышленности, где используется большой номинал различной продукции (изделий).

CALS-технологии (*Continuous Acquisition and Life cycle Support*) — зонтичный термин, обозначающий непрерывную информационную поддержку поставок и жизненного цикла изделий.

ИПИ (*Информационная Поддержка Изделий*) — русскоязычный термин, эквивалентный понятию CALS, где информационная поддержка процессов ЖЦИ применяется к проектированию и производству высокотехнологичной и наукоемкой продукции.

Достаточно подробно CALS-технологии описаны в уже рекомендованном ранее учебнике Норенкова И.П. «*Основы автоматизированного проектирования*» [5], где этой теме посвящена отдельная глава «6. ИНФОРМАЦИОННАЯ ПОДДЕРЖКА ЭТАПОВ ЖИЗНЕННОГО ЦИКЛА ИЗДЕЛИЙ — CALS-ТЕХНОЛОГИИ». Мы будем также придерживаться другого достаточно авторитетного источника [10] «Концепция развития CALS-технологий в промышленности России. НИЦ CALS-технологий «Прикладная логистика», который более тесно связан со стандартами Технического комитета по стандартизации ТК 431 «*CALS-технологии*».

Как и архитектура АСУ, концепция CALS-технологий или ИПИ имеет свою вертикальную архитектуру:

а) Инвариантные понятия ИПИ;

б) Стадии ЖЦ изделия (ЖЦИ);

в) Интегрированная информационная среда (ИИС) или Единое информационное пространство (ЕИП);

г) Инструментарий: CAE/CAD/CAM, PDM и другие.

Рассмотрим эти уровни более подробно.

Инвариантные понятия ИПИ — *понятия, имеющие разные названия*, но один и тот же смысл, которые условно делятся на две группы [10]: «*Основные ИПИ принципы*» и «*Базовые ИПИ технологии*».

Основные ИПИ принципы — концептуальные намерения, которые ИПИ должна реализовывать. К ним относятся:

а) *анализ и реинжиниринг бизнес-процессов* (*Business-processes analysis and reengineering*) — фундаментальное переосмысление и радикальное перепроектирование бизнес-процессов;

б) *безбумажный обмен данными* (*Paperless data interchange*) — электронный цифровой обмен данными с использованием ЭЦП (электронной цифровой подписи);

в) *параллельный инжиниринг* (*Concurrent Engineering*) — параллельное выполнение различных стадий создания изделия;

г) *системная организация постпроизводственных процессов ЖЦИ* — интегрированная логистическая поддержка изделия.

Базовые ИПИ технологии — это конкретные технологии, по сути реализующие одни и те же решения. К ним относятся:

а) управление проектом (*Project Management*);

б) управление данными об изделии (*Product Data Management, PDM*);

в) управление конфигурацией изделия (*Configuration Management*);

г) управление ИИС, в том числе информационными потоками (*Information Management*);

д) управление качеством (*Quality Management*);

е) управление потоками работ (*Workflow Management*);

ж) управление изменениями производственных и организационных структур (*Change Management*).

Примечание — Важно отметить [10], что: «ИПИ-технологии реализуются силами многопрофильных рабочих групп, объединяющих в своём составе экспертов различных специальностей».

Стадии ЖЦ изделия (ЖЦИ) — качественные состояния, которые изделие проходит от момента концептуальной идеи его создания до полной утилизации. ***ГОСТ Р 50.1.031-2001*** «Терминологический словарь. ...» [11] выделяет следующие девять стадий ЖЦИ:

1) Маркетинговые исследования;
2) Составление технического задания;
3) Проектирование;
4) Технологическая подготовка производства;
5) Изготовление;
6) Поставка;
7) Эксплуатация;
8) Ремонт;
9) Утилизация.

Примечание — Более подробно стадии ЖЦИ обсуждаются в пункте 1.4.1 данного подраздела.

Интегрированная информационная среда (ИИС) или **ЕИП** — ***единое информационное пространство***) — *совокупность распределенных баз данных*, взаимодействующих по единой системе правил и доступных для всех участников стадий ЖЦИ.

Более подробно ИИС и ЕИП обсуждаются в пункте 1.4.2 данного подраздела.

Инструментарий — *набор программных средств* (программных систем), с помощью которых информация в ИИС *создаётся*, *преобразуется*, *хранится* и *передаётся* от одного участника ЖЦИ к другому. Источник [10] выделяет следующие шесть инструментальных средств:

а) автоматизированные системы конструкторского и технологического проектирования (CAE/CAD/CAM);

б) программные средства управления данными об изделии (изделиях) (PDM);

в) автоматизированные системы планирования и управления производством и предприятием (MRP/ERP);

г) программно-методические средства анализа логистический поддержки и ведения баз данных по результатам такого анализа (LSA/LSAR);

д) программные средства управления потоками работ (WF);

е) методология и программные средства моделирования и анализа бизнес-процессов (SADT) и другие.

Примечание — Более подробно инструментарий обсуждается в пунктах 1.4.3, 1.4.4.

На рисунке 1.5 наглядно показано иерархическое отношение указанных понятий.

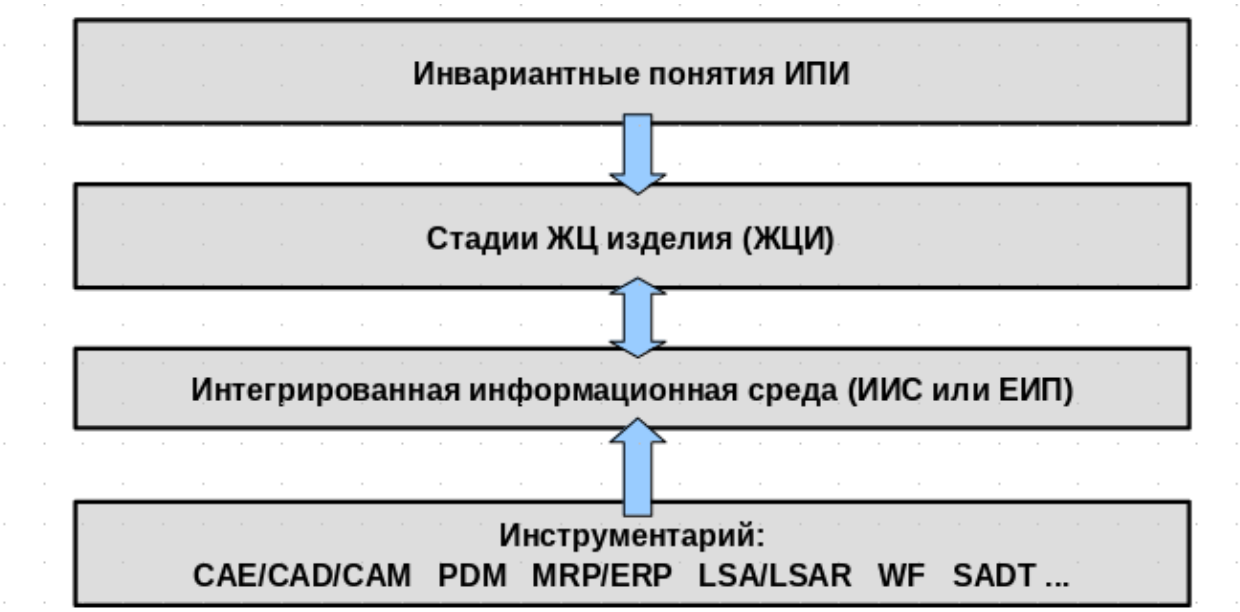

Рисунок 1.5 — Иерархическое представление концепции CALS-технологий

Что касается стандартизации CALS-технологий, то она достаточно запутана и основана на зарубежных источниках. Как отмечает источник [10]: «... приказом Госстандарта РФ №515 от 01.12.99 создан Технический комитет ТК431 «CALS-технологии», силами которого разработан ряд стандартов серии ГОСТ Р ИСО 10303, являющихся адекватными переводами соответствующих международных стандартов». Приведём их список, выделив группировку по общей тематике.

Стандарты группы «Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными»:

а) ГОСТ Р ИСО 10303-1-99 — Часть 1. Общие представления и основополагающие принципы;

б) ГОСТ Р ИСО 10303-11-2000 — Часть 11. Методы описания. Справочное руководство по языку EXPRESS;

в) ГОСТ Р ИСО 10303-12-2000 — Часть 12. Методы описания. Справочное руководство по языку EXPRESS-I;

г) ГОСТ Р ИСО 10303-21-99 — Часть 21. Методы реализации. Кодирование открытым текстом структуры обмена;

д) ГОСТ Р ИСО 10303-22-2001 — Часть 22. Методы реализации. Стандартный интерфейс доступа к данным;

е) ГОСТ Р ИСО 10303-41-99 — Часть 41. Интегрированные обобщённые ресурсы. Основы описания и поддержки изделий;

ж) ГОСТ Р ИСО 10303-45-2000 — Часть 45. Интегрированные обобщённые ресурсы. Материалы;

з) ГОСТ Р ИСО 10303-203-2003 — Часть 203. Протокол применения 203 – Проектирование изделия управляемой конфигурации.

В 2001 году Госстандарт РФ и выпустил ряд документов в статусе ***Рекомендаций по стандартизации*** (РС). Они относятся к стандартам группы «Информационные технологии поддержки жизненного цикла продукции»:

а) ГОСТ Р 5***.01-02 — Техническая документация в электронном виде. Основные положения и общие требования;

б) РС Р 50.1.027-2001 — Автоматизированный обмен технической информацией. Основные положения и общие требования;

в) РС Р 50.1.028-2001 — Методология функционального моделирования;

г) РС Р 50.1.029-2001 — Интерактивные электронные технические руководства. Общие требования к содержанию, стилю и оформлению;

д) РС Р 50.1.030-2001 — Интерактивные электронные технические руководства. Логическая структура базы данных;

е) РС Р 50.1.031-2001 — Терминологический словарь. Часть 1. Стадии жизненного цикла продукции [11];

ж) РС Р 50.1.032-2001 — Терминологический словарь. Часть 2. Применение стандартов серии ГОСТ Р ИСО 10303.

Примечание — Как отмечено в источнике [10]: «Термины, установленные в РС, обязательны для применения во всех видах документации и литературы по технологиям непрерывной информационной поддержки жизненного цикла продукции».

1.4.1 Концепция жизненного цикла изделия (ЖЦИ) применительно к ИС

Как отмечено ранее, концепция жизненного цикла изделия (ЖЦИ) зародилась в недрах министерства обороны США, а затем распространилась на различные отрасли производства и, прежде всего, — на отрасли машиностроения.

В данном пункте мы рассмотрим термин ЖЦИ в более широком смысле — как модель, в которой понятие изделия распространяется на все, что создаётся на предприятии. Например:

а) ***АСУ*** или ***АС*** — как изделие для управления предприятием;

б) ***ИС*** — как изделие для АС или ИИС.

Тогда любая модель ЖЦИ включает в себя следующие обобщённые понятия:

а) *Стадии* и *Этапы* работ;

б) *Результаты* выполнения работ на каждой стадии;

в) *Ключевые события* — точки завершения работ и принятия решений.

В пределах выделенных обобщённых понятий можно использовать одну из трёх разновидностей моделей ЖЦИ: *каскадную*, *итерационную* или *спиралевидную*.

Каскадная модель ЖЦИ — *описывает классический подход* к разработке систем в любых предметных областях. Она предусматривает последовательную организацию работ, которые разбиты на стадии и этапы. Следующая стадия начинается только после завершения предыдущей стадии. Каждая стадия завершается выпуском полного комплекта документации.

Такую модель ЖЦИ для создания АС требует ГОСТ 34.601-90 [8], выделяя наличие восьми последовательно выполняющихся стадий (см. таблицу 1.2, пункта 1.3.1).

Итерационная модель ЖЦИ — *не предполагает чёткого разбиения на стадии и этапы*: система проектируется сразу на нескольких уровнях и состоит из серии коротких циклов (шагов) по ***планированию***, ***реализации***, ***изучению*** и ***действию***. Здесь имеется множество проектных решений *«снизу-вверх»*, когда проектные решения по отдельным задачам объединяются в общие системные решения.

Такую модель ЖЦИ предлагают нам CALS-технологии в плане непрерывной информационной поддержки изделий.

Спиралевидная модель ЖЦИ — *предполагает наличие стадий (этапов)*: ***разработка требований, проектирование, реализация, тестирование, ввод в действие***. Но, в отличие от каскадной модели, эта модель *предполагает итерационный процесс разработки изделия*. Каждая итерация представляет собой законченный цикл разработки, приводящий к выпуску внутренней или внешней версии изделия. На каждой итерации продукция совершенствуется, чтобы стать законченным и более совершенным изделием.

Такая модель ЖЦИ характерна для разработки программного обеспечения больших и достаточно автономных систем, например, ПО офисных систем, когда изделие постоянно совершенствуется, но поступает на рынок как продукт, который объявлен и сопровождается как конкретная версия.

Примечание — В пределах нашей дисциплины, проектирование ИС привязано к жизненного циклу АС, поэтому мы будем опираться на ГОСТ 34.601-90 [8].

1.4.2 Концепции ИИС и ЕИП

В формальном плане термин ИИС определяется в ***Рекомендациях по стандартизации*** (***РС***) ГОСТ Р 50.1.031-2001 [11] в подразделе «*3.2 Интегрированная информационная среда и информационное взаимодействие*». Одновременно там даются понятия, раскрывающие эти термины.

Интегрированная информационная среда (***ИИС****, Integrated Information Environment*, *IIE*) — [11]: «Совокупность распределенных баз данных, содержащих сведения об изделиях, производственной среде, ресурсах и процессах предприятия, обеспечивающая корректность, актуальность, сохранность и доступность данных тем субъектам производственно-хозяйственной деятельности (ПХД), участвующим в осуществлении ЖЦИ (далее — субъекты ПХД), кому это необходимо и разрешено. Все сведения (данные) в ИИС хранятся в виде информационных объектов».

В тех случаях, когда речь идёт о «виртуальном предприятии», используется термин «Единое информационное пространство» или ЕИП.

Информационный объект (***ИО***, *Information Object*) — [11]: «Совокупность данных и программного кода, обладающая свойствами (атрибутами) и методами, позволяющими определенным образом обрабатывать данные. Самостоятельная единица применения и хранения в ИИС».

Класс информационных объектов (***КИО***, *Entity, Class of Information Objects*) — [11]: «ИО, свойства которого объявлены, но им не присвоены конкретные значения. Класс может порождать экземпляры, наследующие его свойства и методы».

Экземпляр класса (***Instans***) — [11]: «ИО, получающийся из КИО присвоением свойствам конкретных значений».

Коллекция объектов (***КО***, *Information Objects Collection*) — [11]: «Совокупность ИО, относящихся к одному или нескольким классам. КО позволяет добавлять и удалять ИО и обращаться к любому из них. КО может использоваться для создания нового ИО».

Информационное взаимодействие (*Information Interaction*) — [11]: «Совместное использование данных, находящихся в ИИС, и обмен данными, осуществляемые субъектами ПХД, в соответствии с установленными правилами».

Совместное использование данных (*Join data using*) — [11]: «Независимое обращение субъектов ПХД к ИО, находящимся в ИИС, с целью их использования в приложениях или модификации в соответствии с установленными правилами».

Правила информационного взаимодействия и обмена данными — [11]: «Правила, регламентирующие для субъектов ПХД:

а) доступ к ИО;

б) право модификации ИО;

в) право на помещение нового ИО в ИИС;

г) протоколы передачи данных по каналам связи;

д) условия защиты информации в ИИС;

е) структуру и форму обменного файла и т. д.».

Общая база данных об изделиях (***ОБДИ***, *Common product data base, CPDB*) — [11]: «Часть ИИС — хранилище ИО, содержащих в произвольном формате информацию, требуемую для выпуска и поддержки технической документации, необходимой на всех стадиях ЖЦИ, для всех изделий, выпускаемых предприятием. Каждый ИО в ОБДИ идентифицируется уникальным кодом и может быть извлечён из ОБДИ для выполнения действий с ним. ОБДИ обеспечивает информационное обслуживание и поддержку деятельности:

а) *заказчиков* (владельцев) изделия;

б) *разработчиков* (конструкторов), технологов, управленческого и производственного персонала предприятия-изготовителя;

в) *эксплуатационного и ремонтного персонала заказчика*. ОБДИ может состоять из нескольких разделов:

1) нормативно-справочного;

2) долговременного;

3) актуального».

Общая база данных о предприятии (***ОБДП***, *Common enterprise data base*) — [11]: «Часть ИИС — хранилище ИО, содержащих в произвольном формате данные о финансово-

экономическом состоянии предприятия, его внешних связях, производственно-технологической среде, действующей на предприятии системе качества и т. д.».

Анализируя уже приведённые определения, становится ясно, что ИИС представляет из себя систему, интегрирующую другие системы, элементами которых являются *информационные объекты* (**ИО**).

Примечание — Учитывая этот факт, сокращение **ИО** (в смысле **ИО АС** — «*Информационное Обеспечение*» по ГОСТ 34.003-90 [6]) далее использоваться не будет.

Следует также заметить, что, в рамках общей БД предприятия, в ИИС выделяются:

а) *база данных по экономике и финансам*;

б) *база данных о внешних связях предприятия*;

в) *база данных о производственно-технологической среде предприятия*;

г) *база данных о системе качества*.

Дополнительно, в том же подразделе, ГОСТ Р 50.1.031-2001 [11] раскрывает понятия *электронного хранилища* и *электронного документа*.

Электронное хранилище (*Vault*) — [11]: «Область хранения ИИС. В хранилище находятся либо ИО, либо информация о путях доступа к ним. Информация в электронных хранилищах контролируется на основе специальных правил и порождаемых ими процессов».

Документ электронный (***ДЭ***, *Electronic Document*) — [11]: «Информационный объект, состоящий из двух частей:

а) реквизитной, содержащей идентифицирующие атрибуты (имя, время и место создания, данные об авторе и т. д.) и электронную цифровую подпись ...;

б) содержательной, включающей текстовую, числовую и/или графическую информацию, которая обрабатывается в качестве единого целого.

При необходимости ДЭ может приобретать различные формы визуального отображения: на экране или бумаге (см. ГОСТ Р 51141)».

Документ технический электронный (***ДТЭ***, *Technical Electronic Document*) — [11]: «Электронный документ, содержательная часть которого включает технические данные».

По отношению к ДЭ и ДТЕ применяются варианты: ***оригинал***, ***подлинник*** и ***копия***.

Документация электронная (*Electronic Documentation, ED*) — [11]: «Комплект ДЭ и/или ДТЭ, объединённых по тематическому или предметному принципу, например техническая документация в электронной форме: чертежи, ведомости, спецификации, сборочные таблицы, сопроводительные документы, производственные требования и стандарты; прочая информация, подготовленная в процессе проектирования и относящаяся к конструкции, производству, контролю, управлению, снабжению и сбыту, испытаниям и контролю изделий. Документация электронная представляет собой коллекцию ИО ..., формируемую в ИИС».

Электронная цифровая подпись (***ЭЦП***, *Digital signature*) — [11]: «Специальное криптографическое средство обеспечения подлинности, целостности и авторства ДЭ или ДТЭ. ЭЦП связывает содержание документа и идентификатор подписывающего лица и делает невозможным изменение документа без нарушения подлинности подписи. Формирование ЭЦП электронного документа или пакета документов (файла или файлов) при их подготовке

и передаче, а также проверка наличия и неискажённости подписи обеспечиваются специальными программными средствами (см. ГОСТ Р 34.10)».

Завершим перечень определений, характеризующих ИИС, описанием понятия изделие, которое даётся в подразделе «*3.3 Изделие и предприятие*» рассматриваемого стандарта.

Изделие (Product, Item) — [11]: «По 3.2.26 ГОСТ Р ИСО 10303-1.

П р и м е ч а н и е — Изделие может представлять собой *материальный предмет*, *вещество*, *услугу*, *программный продукт*, *систему*, состоящую из материальных предметов и программных средств, взаимодействующих между собой, являющихся результатом деятельности предприятия».

Таким образом, **концепция ИИС** требует создания единой информационной проекции предприятия.

> Примечание — *Организационно*, эта проекция представляет собой единое корпоративное **Электронное хранилище** (*Vault*), а *логически* — распределенную неоднородную объектную систему, подсистемы которой сосредоточены в базах данных различного назначения, содержащих и предоставляющих доступ к **Информационным Объектам** (**ИО**).

Представленная интерпретация концепции ИИС — весьма привлекательна для базового представления тематики изучаемой дисциплины, поскольку позволяет интерпретировать **ИС** как **ИИС** или её достаточно самостоятельную часть. Тем более, что такая интерпретация ИС является более значимой по сравнению с её проекцией — как обеспечивающей подсистемы **АС**.

С другой стороны, кажущаяся простота концепции ИИС — обманчива. Она переносит проблематику создания **Электронного хранилища** в две плоскости:

а) в проблематику наличия или создания ***Инструментария*** нужного назначения, обеспечивающего создание и манипулирование информационными объектами (ИО);

б) в проблематику наличия или создания ***Распределенной вычислительной сети*** (***РВ-сети***), обеспечивающей взаимодействие различных частей ИИС.

Первый аспект проблем обсуждается в последующих двух пунктах данного подраздела, а тематика второго аспекта проблем — в других темах пособия (по мере необходимости). Здесь лишь отметим, что РВ-сети, в плане своего развития (см. учебное пособие [3]), стали предоставлять сервис ***Виртуальных предприятий***. Если предприятие или организация используют подобный сервис для развёртывания своего ***Инструментария***, то вместо термина ***ИИС*** принято использовать термин ***ЕИП***, который расшифровывается как ***Единое Информационное Пространство***.

1.4.3 САПР и их классификации

Самым нижним уровнем, в иерархии представления концепции CALS-технологий показанном на рисунке 1.5, является **Инструментарий**, который обеспечивает среду ИИС в плане создания, модификации и перемещения ИО или более сложных объектов ДЭ и ДТЭ. Этот инструментарий может быть достаточно разнообразным, но тот, который предназначен для автоматизации проектирования изделий, в России называется **САПР** или **Система автоматизированного проектирования**. Более точное определение САПР даётся в ГОСТ 22487 — 77 «ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЕ. Термины и определения» [12].

Система автоматизированного проектирования (***САПР***) — [12]: «Комплекс средств автоматизации проектирования, взаимосвязанных с необходимыми подразделениями проект-

ной организации или коллективом специалистов (пользователем системы), выполняющий автоматизированное проектирование».

В целом, САПР стали интенсивно развиваться ***с середины 1960-х годов***, когда появились ЭВМ достаточной мощности, чтобы осуществлять подобную деятельность. В частности, в это время американская корпорация IBM выпустила свою систему «*IBM Drafting System*».

В англоязычной литературе термин САПР переводится различными способами. Например на рисунке 1.5 ему соответствует многозначное обозначение **CAE/CAD/CAM**, но общепринятым обобщённым термином является **CAD** или **Computer-Aided Design**. Чтобы разобраться с указанной неопределённостью обозначений, учтём, что развитие САПР шло параллельно с развитием аппаратных средств вычислительной техники и его программного обеспечения. К тому же САПР — достаточно сложные и дорогие системы, которые классифицируются по двум признакам: *отраслевому* и *целевому* назначениям.

По отраслевому назначению, САПР делятся *на три большие группы* и обозначаются:

а) **MCAD** (*Mechanical Computer-Aided Design*) — автоматизированное проектирование механических устройств. Машиностроительные САПР, применяются в автомобилестроении, судостроении, авиакосмической промышленности, производстве товаров народного потребления. Они включают в себя разработку деталей и сборок (механизмов) с использованием параметрического проектирования на основе конструктивных элементов, технологий поверхностного и объёмного моделирования: SolidWorks, Autodesk Inventor, CATIA;

б) **EDA** (*Electronic Design Automation*) или **ECAD** (*Electronic Computer-Aided Design*) — САПР электронных устройств, радиоэлектронных средств, печатных плат и другие: Altium Designer, OrCAD;

в) **AEC CAD** (*Architecture, Engineering and Construction Computer-Aided Design*) или **CAAD** (*Computer-Aided Architectural Design*) — САПР в области архитектуры и строительства. Используются для проектирования зданий, промышленных объектов, дорог, мостов и другое: Autodesk Atchitectural Desktop, Piranesi, ArchCAD.

По целевому назначению, САПР делятся *на четыре группы* и обозначаются:

а) **CAD** (*Computer-Aided Design/Drafting*) — средства автоматизированного проектирования, предназначенные для автоматизации двумерное или трёхмерного геометрического проектирования, создания конструкторской или технологической документации, САПР общего назначения. Для обозначения данного класса средств САПР используется также термин **CADD** (*Computer-Aided Design and Drafting*) — автоматизированное проектирование и создание чертежей, а системы геометрического моделирования обозначают как **CAGD** (*Computer-Aided Geometric Design*);

б) **CAE** (*Computer-Aided Engineering*) — средства автоматизации инженерных расчётов, анализа и симуляции физических процессов, которые осуществляют динамическое моделирование, проверку и оптимизацию изделий. Подкласс средств CAE, используемый для компьютерного анализа, обозначается термином **CAA** (*Computer-Aided Analysis*);

в) **CAM** (*Computer-Aided Manufacturing*) — средства технологической подготовки производства изделий, которые обеспечивают автоматизацию программирования и управления оборудованием с ЧПУ или ГАПС (Гибких автоматизированных производственных систем). Русскоязычным аналогом термина является **АСТПП** — автоматизированная система технологической подготовки производства.

г) **CAPP** (*Computer-Aided Process Planning*) — средства автоматизации планирования технологических процессов, применяемые на стыке систем CAD и CAM.

Подводя итог анализу терминологии САПР, можно сделать следующие выводы:

а) отсутствуют САПР, которые бы предназначались для непосредственного проектирования ИС;

б) даётся прямое указание на то, что САПР сами являются автоматизированными системами (АС), которые уже были рассмотрены нами в предыдущем подразделе 1.3.

Чтобы более обосновано подтвердить сделанное заявление, снова обратимся к ГОСТ 22487— 77 [12], где перечисленны все обеспечивающие подсистемы САПР. Они полностью совпадают с перечисленными ранее обеспечивающими подсистемами АС. Единственное, что в САПР добавляется проектирующее программное обеспечение. В своей работе [13], автор учебника по автоматизированному проектированию И.П. Норенков представляет *общую архитектуру ПО САПР*, показанную на рисунке 1.6.

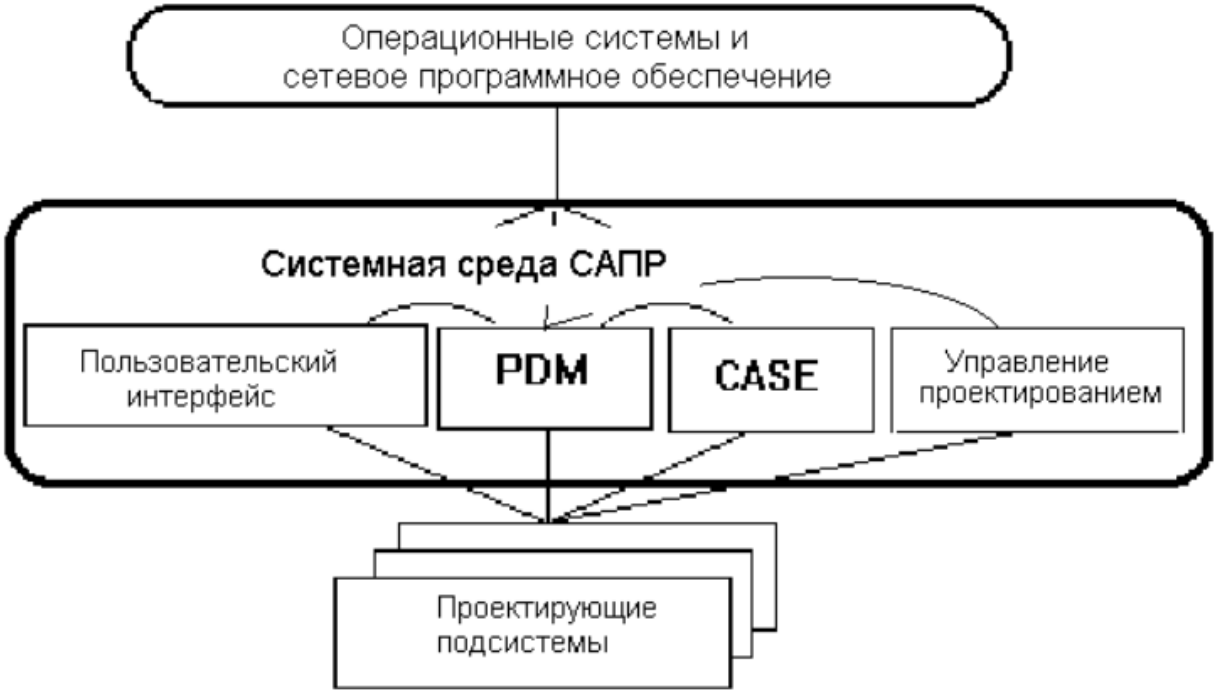

Рисунок 1.6 — Структура программного обеспечения САПР [13]

Хорошо видно, что структура программного обеспечения САПР состоит из трех частей: ОС и сетевого ПО, обеспечивающих подсистем, обозначенных на рисунке как «*Системная среда САПР*», и проектирующих подсистем.

Проектирующие подсистемы непосредственно выполняют проектные процедуры, примерами которых могут служить подсистемы геометрического трёхмерного моделирования механических объектов, изготовления конструкторской документации, схемотехнического анализа, трассировки соединений в печатных платах и другие процедуры.

Обслуживающие подсистемы предназначены для функционирования проектирующих подсистем САПР. Их обычно называют системной средой (или оболочкой) САПР. В качестве типичного состава обслуживающих подсистем выделяют:

а) **PDM** (*Product Data Management*) — подсистема управления проектными данными, являющаяся организационно-технической системой управления всей информацией о сложных изделиях, таких как корабли, самолёты, ракеты, компьютерные сети и дру-

гие;

б) **DesPM** (*Design Process Management*) — подсистема управления самим процессом проектирования;

в) подсистема управления пользовательскими интерфейсами для связи разработчиков с ЭВМ;

г) **CASE** (*Computer Aided Software Engineering*) — набор инструментальных средств для разработки и сопровождения программного обеспечения САПР.

Примечание — Практически все обслуживающие подсистемы САПР, кроме CASE-систем, являются достаточно специализированными инструментами, чтобы их можно было изучать в общем курсе по проектированию ИС.

1.4.4 CASE-средства CALS-технологий

Согласно своему названию CASE-средства стали развиваться как инструменты и методы программной инженерии для целей проектирования программного обеспечения. Собственно в таком качестве они и вошли в состав САПР. Но с момента появления стандарта ISO/IEC 14102:2008 «*Guide-line for thee valuation and selection of CASE tools*» — Руководство по оценке и выбору инструментов CASE, эти средства стали рассматриваться как поддержка жизненного цикла ПО, что вполне соответствует тематике данного подраздела.

Здесь совершенно нельзя обойти вниманием широкоизвестный учебник Вендрова А. М. «*Проектирование программного обеспечения экономических информационных систем*» [14], где автор систематически рассматривает ЖЦИ ПО, попутно интерпретируя все уже описанные нами технологии применительно к выбранной тематике. Дополнительно, изложенная в [14] теоретическая часть технологии проектирования ПО дополняется учебным пособием: Вендров А.М. «*Практикум по проектированию программного обеспечения экономических информационных систем*» [15], где автор, используя инструментарий компании ***Rational Software*** и её методологии ***RUP*** (***Rational Unified Process***), демонстрирует ряд проектных решений, включая применение языка UML.

Примечание — Следует обратить внимание, что прямое использование методологии и инструментария CASE-средств ещё не означает проектирование ИС, но представляет значительный интерес в плане её возможной реализации.

В плане тематики нашей дисциплины, CASE-средства интересны тем, что интенсивно используют структурный подход:

а) SADT (Structured Analysis and Design Technique);

б) DFD (Data Flow Diagrams);

в) ERD (Entity-Relationship Diagrams).

Кроме того, в общем случае и в плане функциональной ориентации, CASE-средства подразделяются:

а) средства анализа, предназначенные для построения и анализа модели предметной области;

б) средства проектирования баз данных;

в) средства разработки приложений;

г) средства реинжиниринга процессов;

д) средства планирования и управления проектами;

е) средства тестирования ПО систем;

ж) средства документирования проектов.

В указанных аспектах, кроме достаточно дорогих CASE-средств компании ***Rational Software***, которые ***с 2003 года*** принадлежат корпорации IBM, имеется большое количество свободного ПО, выполняющего аналогичные или подобные функции.

Начнём с того, что первоначально средства автоматизации разработки программного обеспечения стали формироваться как системы **IDE** (*Integrated Development Environment*). В частности, та же корпорация IBM, стремясь навести порядок в своих инструментальных средствах основанных на языке Java, сформировала ***к 2001 году*** инструментальную систему (IDE) под названием Eclipse («*Затмение*»). Затратив (по сообщениям самой же IBM) порядка 40 миллионов долларов, эта система была передана, ***в феврале 2004 года***, в общественный фонд под названием ***Eclipse Foundation***. ***В июне 2004 года года***, этот фонд выпустил версию Eclipse 3.0, которая полностью соответствовала спецификациям сервисной платформы **OSGi** (*Open Services Gateway initiative*) и приобрела модульную расширяемую структуру. ***В сентябре 2017 года***, правопреемница технологий языка Java корпорация Oracle передала Eclipse Foundation права на использование Java EE (*Java Platform Enterptise Edition*). ***В апреле 2018 года***, Eclipse Foundation переименовала Java EE в новую платформу **Jakarta EE**, добавив в неё поддержку облачных технологий.

Интерес к IDE Eclipse не является случайным. Дело в том, что эта инструментальная платформа хорошо интегрируется с другими инструментальными средствами, имеющими к ИС прямое отношение: СУБД и сервера web-сервисов. На момент написания данного текста (март 2020 года), дистрибутивы Eclipse представлены тринадцатью специализированными системами для ОС MS Windows, Mac Cocoa и Linux. Среди них можно выделить:

а) Eclipse IDE for **Enterprise** Java Developers — среда для разработки приложений уровня предприятий;

б) Eclipse IDE for **Web** and **JavaScript** Developers — среда для разработки современных web-приложений;

в) Eclipse **Modeling** Tools — среда для моделирования приложений, включающая средства построения различных диаграмм.

Разработанная на языке Java, IDE Eclipse имеет средства доступа ко всем известным СУБД посредством ПО **JDBC** (*Java DataBase Connectivity*), но наиболее интересна её тесная связь с ПО организации-фонда **Apache Software Foundation** (ASF).

Фонд ASF был ***основан в 1999 году***, а первым его наиболее известным брендом стал **Apache HTTP-сервер** — свободный web-сервер, написанный на языке C в 1995 году, но поддерживающий модульную структуру, он один из первых реализовал поддержку языка **PHP** (*Hypertext Preprocessor*), открыв дорогу другим решениям основанным на технологии клиент-сервер и протоколе HTTP.

Для нас, с практической точки зрения, более интересен проект **Apache Tomcat**, реализованный ASF в 1999 году на языке Java, он представляет контейнер сервлетов с поддержкой технологий **JSP** (*JavaServer Pages*) и **JSF** (*JavaServer Faces*). В совокупности это позволило разрабатывать web-приложения уровня предприятия.

В последующие годы ASF реализовала или обеспечивала поддержку нескольким десяткам различных проектов, многие из которых содержали различные технологические новинки. Среди них можно отметить:

а) **Apache Derby** — реляционная СУБД, написанная на Java в 1997 году и обеспечивающая как сетевой, так и встроенный варианты подключения приложений;

б) **Apache OpenOffice** — свободный пакет офисных приложений, реализованный в апреле 2002 года и одним из первых поддерживающий новый открытый формат ISO/IEC 26300 (*OpenDocument*);

в) **Apache TomEE** — свободный сервер приложений, начатый разрабатываться в апреле 2012 года и полностью реализованный в сентябре 2017 года.

Современный уровень развития свободного программного обеспечения вполне реализует замену многих CASE-средств устаревших технологий.

1.4.5 Выводы по подразделу

Таким образом, парадигма CALS-технологий является наиболее современной концепцией построения ИС, поскольку она включает в себя достижения всех предыдущих парадигм, а также вносит свои собственные концептуальные представления, которые отсутствуют у рассмотренных ранее.

Примечание — Прежде всего отметим, что парадигма технологий и ИС, как система, — это совершенно разные, хотя и связанные, понятия.

Парадигма технологий — это набор концептуальных представлений, которые нам дают качественные представления о средствах реализации системы, например, ИС.

ИС, **как система**, — это фактически проект, который описывает как концептуальные представления должны быть реализованы и что из этого получится.

Чтобы наглядно показать различие заявленных понятий, обратим внимание, что парадигмы вычислительных систем, информационного подхода, концепции АСУ и CALS-технологий упорядочены по мере развития средств вычислительной техники и ее программного обеспечения.

Парадигма вычислительных систем основана на развитии языков программирования и алгоритмов быстрого вычисления различных функций. Сами вычисления сначала проводились в пакетном режиме запуска отдельных программ, а затем — в мультипрограммном режиме вычислительной среды различных ОС. При этом информация как таковая не являлась элементом вычислительных технологий. Она присутствовала в головах разработчиков алгоритмов и программ, а также специалистов, интерпретировавших результаты вычислений. ИС, как системы, фактически не существовали, поскольку информация формировалась и накапливалась, по результатам научных или инженерных исследований, вне пространства вычислительных систем и в виде журнальных публикаций, теоретических концепций научных школ и архивов библиотечных учреждений.

Негативные последствия указанного подхода широко известны как проблематика парадигмы «*Программа-массив*», порождающая необходимость написания двух дополнительных программ на каждую пару связываемых прикладных программ (см. пункт 1.1.2).

Парадигма информационного подхода вводит в багаж технологий разработчиков ПО само понятие информации, переводя научные и инженерные знания в форматы данных, которые сохраняются во внешней (долговременной) памяти средств вычислительной техники.

Первоначально информация начала сохраняться в файловых системах ОС, подтверждая базовый тезис систем UNIX: «*Все есть файл*». Формируются первые прообразы ИС в виде комплектов документации на различные языки и системы программирования. Делаются

первые попытки автоматизации доступа к документации в виде специальных утилит, например, утилиты ***man*** и ***info***.

Развитие сетевых технологий и специальных средств управления данными (СУБД) выводят информационный подход на новый уровень, ставя задачи проектирования ИС и стимулируя различные мультимедийные разработки в области представления информации. В таких условиях ИС уже рассматривается как обеспечивающая подсистема некоторого набора приложений, реализованным в виде **АРМ** (*Автоматизированного Рабочего Места*), с простейшей распределенной архитектурой показанной на рисунке 1.7.

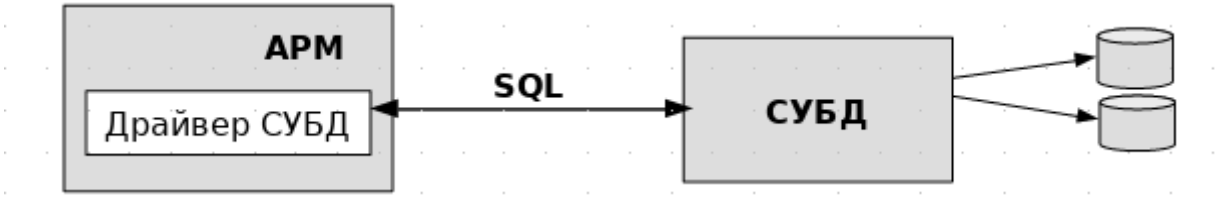

Рисунок 1.7 — ИС обслуживающая отдельное приложение

Примечание — Являясь обслуживающей подсистемой АРМ, ИС вполне удовлетворяется возможностями реляционных СУБД, предоставляя ПО АРМ информацию в виде простейших типов данных.

Дальнейшее публичное развитие парадигмы информационного подхода идёт в направлении разделения инструментальных средств ИС, согласно модели «*Клиент-сервер*», и унификации этих средств в плане отображения, обработки и хранения информации. Примеры такого развития предоставляют web-технологии, где сторона клиента реализуется в виде web-браузеров, а прикладная часть приложений переносится на сторону сервера (web-сервера). И в такой интерпретации ИС рассматриваются как web-сайты.

Парадигма АСУ ограничивает рамки рассматриваемых информационных технологий предметной областью нашей дисциплины, одновременно накладывая на целевое назначение ИС требование обеспечить приемлемый уровень управления производственной структурой уровня предприятия.

В такой проекции проектирование и создание ИС столкнулось с рядом **серьёзных препятствий**:

а) низкий уровень представления и обработки данных реляционными СУБД, ориентированными на информационное обеспечение расчётных прикладных программ;

б) большое количество разнообразных по постановкам и алгоритмам решения задач автоматизации производственно-хозяйственной деятельности предприятия;

в) непомерно большие объёмы данных, необходимые для ручного ввода в АС с целью начала ее работы и поддержки актуальности баз данных.

Результат воздействия указанных препятствий привёл к следующим **последствиям**:

а) достаточно успешной автоматизации задач уровня оперативного исполнения (АСУТП) в силу достаточно простого представления информации в устройствах контроллерного управления;

б) «*островной*» или «*кусочной*» автоматизации ряда общих задач производственно-хозяйственной деятельности: кадровое, финансовое и общее материальное и энергетическое обеспечение предприятия;

в) кризисному положению задач уровня тактического управления, которые во многом уникальны для каждого предприятия и не могут быть решены собственными силами

его сотрудников.

Примечание — В рамках парадигмы АСУ, ИС формируются как обслуживающие подсистемы отдельных прикладных систем, которые проектно не связаны между собой в силу их раздельного внешнего проектирования, реализации и сопровождения.

Парадигма CALS-технологий вводит в предметную область деятельности предприятия понятие ЖЦИ — жизненного цикла изделия, которое позволяет рассматривать ИС под новым углом зрения:

а) вводится понятие **ИИС** — Интегрированной Информационной Среды;

б) вводится понятие **ИО** — Информационного Объекта;

в) вводятся и стандартизируются **Правила** информационного взаимодействия и обмена данными между субъектами **ПХД** (субъектами Производственно-Хозяйственной Деятельности).

Примечание — В таких условиях значение ИС выводится на новый уровень значимости. Теперь **ИС** не просто обеспечивает информационную потребность отдельных прикладных программ на уровне простейших типов данных, а обслуживает документооборот предприятия, манипулируя сложными информационными объектами (**ИО**).

Концепция ЖЦИ позволяет рассматривать всю деятельность предприятия как изготовление изделий — *материальных* и/или *информационных*. Изделиями могут быть отдельные ИО, ИС или вся среда ИИС, которые находятся на разных стадиях своего жизненного цикла и по разному взаимодействуют между собой. Что касается самой ИИС, то она может быть в общем случае представлена некоторым множеством *функциональных центров* (***ФЦ***), преобразующих ***входные*** материальные и информационные объекты в ***выходные*** объекты. И, в случае максимального уровня неопределённости, её можно представить рисунком 1.8.

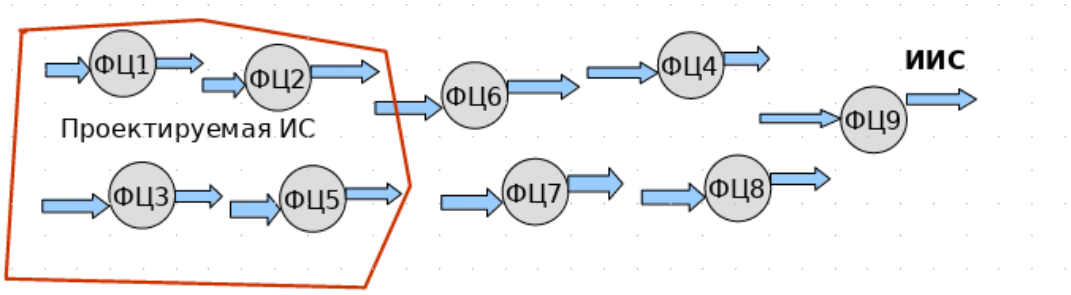

Рисунок 1.8 — Общее представление ИИС как множества функциональных центров предприятия

Таким образом, концепция ЖЦИ CALS-технологий снимает жёсткие требования к информационной архитектуре производственной деятельности предприятия, предоставляя варианты эволюционного развития этой системы в целом. Развитие ИИС может рассматриваться как выделение некоторой совокупности ***Функциональных Центров*** (***ФЦ***), описания их входов и выходов, как отдельных интерфейсов, и последующая модификация вычислительных средств и программного обеспечения для согласования этих интерфейсов.

Примечание — В условиях представленной модели ИИС, задача проектирования ИС может быть поставлена как *деятельность по дополнению выделенного набора ФС необходимым набором ПО*, обеспечивающим согласование всех их интерфейсов.

1.5 Проектирование ИС

ИС — это изделие предприятия или часть его технической производственной базы?

Рассмотрев все четыре исторически сложившихся парадигмы описания ИС, легко прийти к выводу, что наиболее современным взглядом на объект исследования является модель **ИИС** (*Интегрированной Информационной Среды*), которая согласно ГОСТ Р 50.1.031-2001 [11] обеспечивает все процессы ПХД (***Производственно-Хозяйственной Деятельности***) предприятия, сохраняя и предоставляя данные всем хозяйствующим субъектам посредством совокупности распределённых баз данных. Исходя из этой модели, ***предметная область ИС*** — выделенная, согласно целевому заданию на проектирование, совокупность **ФС** (***Функциональных Центров***) ИИС (см. рисунок 1.8), которые должны быть согласованы по входным и выходным интерфейсам.

Конкурирующей, но более адекватной, является парадигма АСУ, которая стандартами ГОСТ серий ***24.xxx***, ***34.xxx*** и ***19.xxx*** создаёт нормативную базу для проектирования ИС.

Познавательная цель данного подраздела — описать общую последовательность проектных действий и решений, необходимых для достижения студентом позитивного результата в процессе проектирования ИС.

Приступая к выполнению производственной практики или ВКР, студент сталкивается *с тремя основными вопросами*:

1) Какую ***тему работы*** выбрать и как правильно поставить задачу на проектирование?
2) Какие ***основные модели*** и технологии следует применить в процессе выполнения работы?
3) Какова ***правильная последовательность работ***, необходимая для выполнения проекта, и чем её обосновать или подтвердить?

Нужные ответы на эти вопросы зависят от специфики самой предметной области и задания полученного от руководителя практики или ВКР. На них студент должен научится отвечать самостоятельно, используя знания полученные во время учёбы в вузе. В данном подразделе изложены только общие рекомендации, которые студент должен использовать, чтобы не допускать грубых ошибок, а более подробный учебный материал по проектированию ИС изложен в последующих разделах учебного пособия.

1.5.1 Общая постановка задачи на проектирование

Студент должен отличать задачу на ***проектирование*** от ***исследовательской*** задачи.

Любая задача всегда содержит некоторую описательную часть, где присутствуют названия вещественных предметов, научных теорий или разделов знаний, а также выделяется целевая составляющая — что должно быть получено в результате решения задачи.

Целевая составляющая исследовательской задачи — изучение и описание чего-либо, что содержит большую неопределённость и заранее неизвестен результат этого изучения.

Например, нужно исследовать некоторый алгоритм или алгоритмы, найти оптимальное решение или возможность достижения чего-либо. Обычно такие задачи связаны с разработкой некоторого инструментария, прямое использование которого не является очевидным.

Целевая составляющая задачи ПИС — создание некоторой системы, включающей аппаратное и программное обеспечение при заданных ограничениях на аппаратную платформу, ОС и другие программные системы.

В общем случае постановок задачи, ПИС достаточно чётко выделяет следующие требования:

а) *автоматизировать* некоторый бизнес-процесс;

б) *обеспечить* сбор, сохранение некоторой информации (данных) и доступ к ней;

в) *осуществить* взаимодействие удалённых между собой систем.

> **Задачи ПИС** всегда имеют ограничивающие условия, которые можно рассматривать как ***внешнее управление***, а главное — они имеют одного или несколько ***субъектов*** (*исполнителей*), которые несут персональную ответственность за работу системы посредством непосредственного управления ею или посредством контроля её функционирования.

Студенту следует сразу научиться рассматривать задачу на ПИС как ***обобщённую функцию***, которую он в последующем может декомпозировать до нужного уровня подробности, при этом выделяя:

а) *входные* информационные объекты (ИО), которые ИС должна импортировать;

б) *выходные* ИО, которые ИС должна создать (экспортировать);

в) *документы*, приказы и распоряжения, ограничивающие выполнение функции ИС;

г) *список сотрудников предприятия*, которые участвуют в выполнении функции ИС или контролируют её выполнение.

> Примечание — Студент должен отличать задачу на проектирование ИС (***ПИС***) от задач на проектирование АС и других задач.

ИС является обеспечивающей подсистемой некоторой ***метасистемы***, поэтому задача ПИС формулируется в пределах некоторой АС (или её части) или в пределах САПР (или её части).

Классическим (эталонным) примером постановку задачи на ПИС можно считать формирование ***ТЗ*** на ***АС***, где:

1) в качестве ***базовой основы*** берётся конкретная подсистема АС и изучается её обеспечивающая подсистема ИС;
2) ***дополнительно***, выделяются и описываются расчётные подсистемы, которые не относятся к ИС, но требуются для функционирования подсистемы АС в целом;
3) относительно дополнительных подсистем принимаются ***отдельные решения*** о включении их в задание на проектирование или нет.

Если задача формулируется в пределах некоторой прикладной производственной системы, например, ***ERP*** некоторого производителя, то такие случаи выходят за рамки нашей дисциплины и здесь не рассматриваются.

> Студент должен уметь правильно оценить ***требуемый объем работ*** для поставленной задачи.

Обычно, стремясь создать полезную для производства систему и получить хорошую оценку, студент завышает свои возможности и, в результате, не успевает выполнить все необходимые работы. Нужно всегда помнить, что система всегда кажется проще, чем она есть на самом деле.

Чтобы избежать или уменьшить последствия неправильной оценки требуемого объёма работ на проектирование и реализацию ИС, следует найти и изучить один или несколько ***прототипов*** создаваемой системы.

Нужно помнить, что в проектировании полностью оригинальных систем практически не существует. Всегда существуют некоторые ***аналоги***, которые по ряду причин не могут быть использованы непосредственно. Поэтому, проводя рассуждения относительно существующего, а значит и уже описанного аналога (***прототипа***), студент имеет опорную базу, относительно которой гораздо легче описывать требования к проектируемой ИС и обосновывать доказательную составляющую проектных решений.

Знание и хорошее описание прототипов будущей ИС является показателем профессиональной подготовки студента и оценивается только с положительной стороны.

Примечание — Студент должен видеть конечную цель постановки задачи: **Частное Техническое Задание** (***ЧТЗ***) на проект ИС.

Следует правильно понимать, что ***Задание*** студента на производственную практику или ВКР, написанное максимум на двух страницах текста и содержащее название темы работы, перечень графического материала и других обязательных атрибутов отчёта или пояснительной записки, частью которых оно подразумевается, является организационным документом вуза, обосновывающим легитимность учёбы студента, но не является ЧТЗ на ПИС.

Для многих направлений обучения студентов существующая практика вузов обычно не требует разработки технических заданий на учебные работы, поскольку в большинстве случаев они имеют исследовательский характер, хотя и связаны с некоторыми элементами проектирования. Обычно такое проектирование необходимо для разработки программного обеспечения, но в силу небольшой сложности создаваемых систем или по иным причинам явное создание документа ТЗ (ЧТЗ) не производится.

Примечание — В рамках изучаемой дисциплины, создание документа ЧТЗ на ПИС является обязательным этапом процесса проектирования.

Поскольку ИС является обслуживающей подсистемой некоторой АС, то и написание ЧТЗ на ПИС следует создавать на основе ТЗ на АС.

Поскольку разработка ИС тесно связана с разработкой ПО, то обычно при создании ЧТЗ на ИС руководствуются требованиями ГОСТ серии 19, которые регламентируют разработку программного обеспечения.

Более подробно вопросы создания ЧТЗ на ПИС рассмотрены во втором разделе данного учебного пособия, озаглавленного как «*Концептуальное проектирование ИС*».

Студент должен правильно понимать, что написание ЧТЗ является не просто «данью традиций», требуемых формальными процедурами проектирования, а важнейшим документом, по которому всегда оценивается работа проектировщика.

Примечание — Правильное составление ЧТЗ должно организовать всю последующую работу студента по созданию ИС. Подписанный и утверждённый документ ЧТЗ выступает самым авторитетным аргументом при принятии последующих проектных решений.

1.5.2 Базовые модели проектирования ИС

В общем случае при проектировании ИС могут использоваться все модели, порождённые парадигмами и технологиями, описанными в первых четырёх подразделах данного раздела. Многие из таких моделей являются достаточно специализированными и применяются только в отдельных отраслях производства.

Направления подготовки бакалавров «*Информатика и вычислительная техника*» и «*Прикладная информатика*» традиционно ориентированы на автоматизацию производственных бизнес-процессов, что связано с задачами управления и информационного обеспечения в АС, а не с общими или специфическими задачами САПР. В таком аспекте для нас наиболее близкими по предметной области являются *архитектурные модели предметной области АСУ* и *распределенная архитектурная модель ИИС*, предложенная парадигмой CALS-технологий.

Что касается *базовых моделей проектирования ИС*, то они делятся на две большие группы:

а) **Функциональные модели проектирования**, которые отражают *концептуальную часть любой системы*, выражая её будущие возможности в виде выполняемых ею функций и давая основания для сравнения её с другими аналогичными системами или прототипами;

б) **Объектные модели проектирования**, которые отражают *проект реализации конкретной системы* через архитектуру и взаимосвязь объектов абстрактных или конкретных языков программирования.

Излишне напоминать, что указанные базовые модели находятся в диалектическом противоречии, хотя предназначены для описания одной и той же системы. Они борятся за «*главенство*» в праве влияния на проектные решения, которые принимаются в процессах проектирования ИС.

> Примечание — Согласно общим принципам проектирования первенство влияния на проектные решения принадлежит *функциональным моделям*.

Главенство функциональной модели над объектной основано на принципе ***функциональной целесообразности*** создания системы, которая должна функционировать согласно поставленной цели. Сама цель является по отношению к системе внешней (*декларативной*). Её достижению подчинены все проектные решения.

Само применение функциональных моделей начинается с постановки задачи на проектирование, что должно заканчиваться формированием и последующим утверждением *частного технического задания* (ЧТЗ) на *проектирование информационных систем* (ПИС).

Более подробно эти вопросы изложены во втором разделе данного пособия, но главное, что должен помнить студент — ЧТЗ на ПИС излагается в терминах функциональных моделей, а объектные модели присутствуют в виде ограничений на задачи проектирования.

Завершается построение функциональной модели ИС на *этапе концептуального проектирования*. Этим вопросам посвящён третий раздел данного пособия, где подробно описаны синтаксис и семантика всех трёх основных функциональных моделей: IDEF0, IDEF3 и DFD. Но основной здесь является модель IDEF0, которая полностью задана рекомендациями РС Р 50.1.028-2001 [16]. Эта модель позволяет проводить декомпозицию *организационно-технической структуры организации* (предприятия) в соответствии с уровнем используемых функций, что наглядно показано на рисунке 1.9.

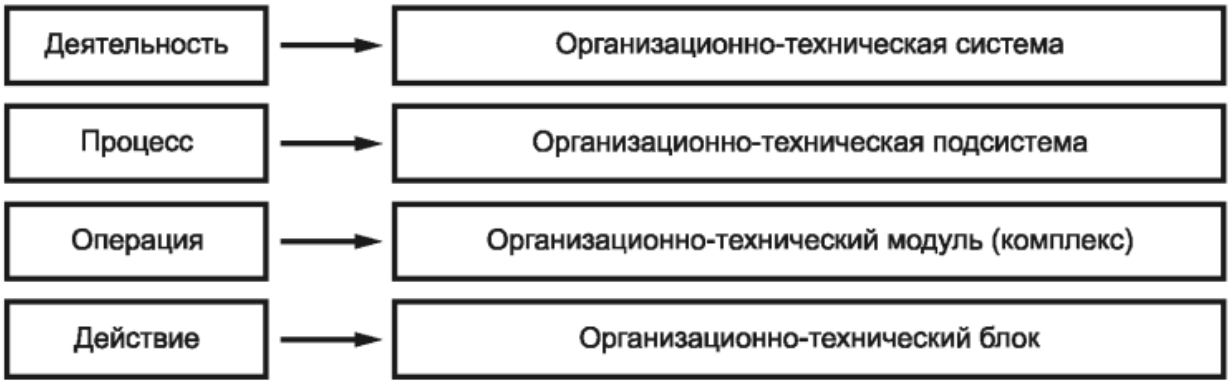

Рисунок 1.9 — Проекция функций модели IDEF0 на организационно-техническую структуру предприятия [16]

Формальная связь элементов рисунка 1.9 определяется стандартом [16], что подтверждается цитатой: «...

Организационно-техническая система — организационная структура, персонал и комплекс технических средств (оборудование), необходимые для осуществления ***деятельности***.

Организационно-техническая подсистема — часть организационно-технической системы, обеспечивающая протекание ***процесса*** (субдеятельности).

Организационно-технический комплекс (модуль) — часть организационно-технической подсистемы, предназначенная для выполнения ***операции.***

Организационно-технический блок — часть организационно-технического комплекса, обеспечивающая выполнение ***действия***».

Необходимо заучить и уверенно использовать семантику этих функций [16]: «...

Деятельность (синонимы: *дело, бизнес*) — *совокупность процессов*, выполняемых (протекающих) последовательно или/и параллельно, преобразующих множество материальных или/и информационных потоков во множество материальных или/и информационных потоков с другими свойствами. **Деятельность** осуществляется в соответствии с заранее определённой и постоянно корректируемой **целью**, с потреблением финансовых, энергетических, трудовых и материальных **ресурсов**, при выполнении **ограничений** со стороны внешней среды. ...

Процесс (синоним: *бизнес-процесс*) — *совокупность последовательно или/и параллельно выполняемых операций*, преобразующая материальный или/и информационный потоки в соответствующие потоки с другими свойствами. **Процесс** протекает в соответствии с управляющими директивами, вырабатываемыми на основе **целей деятельности**. В ходе процесса потребляются финансовые, энергетические, трудовые и материальные **ресурсы** и выполняются **ограничения** со стороны других процессов и внешней среды. ...

Операция — *совокупность последовательно или/и параллельно выполняемых действий*, преобразующих объекты, входящие в состав материального или/и информационного потока, в соответствующие объекты с другими свойствами. Операция выполняется: а) в соответствии с **директивами**, вырабатываемыми на основе директив, определяющих протекание процесса, в состав которого входит операция; б) с потреблением всех видов необходимых **ресурсов**; в) с соблюдением ограничений со стороны других операций и внешней среды. ...

Действие — *преобразование какого-либо свойства материального или информационного объекта в другое свойство*. Действие выполняется в соответствии с **командой,** яв-

ляющейся частью **директивы** на выполнение операции, с потреблением необходимых ресурсов и с соблюдением ограничений, налагаемых на осуществление операции ...».

Дополнительно, когда деятельность и процесс являются сложными функциями, допускается использование терминов [16]: «...

Субдеятельность — *совокупность нескольких процессов в составе деятельности*, объединённая некоторой частной целью (являющейся «подцелью» деятельности).

Подпроцесс — *группа операций в составе процесса*, объединённая технологически или организационно».

Теперь рассмотрим объектные модели.

Подчинённое положение объектных моделей не означает их малую значимость. Дело в том, что объектные модели связаны с реализацией ИС, а успешность (*качество*) реализации системы во многом зависит от уровня развития соответствующих технологий. В этом отношении сами объектные модели являются зависимыми от многих условий и могут существенно меняться по мере технологического развития средств вычислительной техники и инструментального обеспечения процессов создания систем. Этот факт наглядно показан в обзорах существующих парадигм (см. подразделы 1.1 — 1.4 настоящего пособия). Более подробно эта часть учебного материала изложена в четвёртом разделе, где наиболее перспективной архитектурой, в плане реализации ИС, следует считать модель распределённых сервис-ориентированных систем.

Не вдаваясь в детали реализации конкретных моделей, будем рассматривать *базовую объектную модель проектирования* в виде классической трёхзвенной архитектуры «*Клиент-сервер*», показанной на рисунке 1.10.

Рисунок 1.10 — Базовая объектная трёхзвенная архитектура «Клиент-сервер»

Бизнес-логика — *центральный метаобъект ИС*, уже реализованный как «Сервер приложений» и содержащий инструментальные средства для размещения «Сервисов» (Объектов-приложений), подразумеваемых как отдельные ***ФС*** (Функциональные центры) ***ИИС*** (Интегрированной Информационной Среды), что ранее уже показано на рисунке 1.8.

Представление — *метаобъект создания, модификации и потребления информации*, уже реализованный как «Клиент» (Набор клиентских приложений) и способный взаимодействовать с метаобъектом «Бизнес-логика» посредством сети.

Данные — *метаобъект объектной трёхзвенной архитектуры*, уже реализованный как СУБД и способный управлять по классической схеме необходимым набором баз данных.

> Примечание — **Главное требование** к базовой объектной модели — *обеспечить реализацию нисходящего процесса проектирования*.

Детали применения конкретной объектной трёхзвенной архитектуры «*Клиент-сервер*» рассмотрим позже в подразделе 1.6.

1.5.3 Этапы проектирования ИС

Проектирование ИС всегда связано с использованием уже готового программного обеспечения (ПО), а также с созданием, по мере необходимости, нового ПО.

Исходя из этого тезиса, студент должен иметь общее представление о ***Единой Системе Программной Документации*** (*ЕСПД*), содержащей [17]: «... комплекс государственных стандартов, устанавливающих взаимоувязанные правила разработки, оформления и обращения программ и программной документации»:

а) ГОСТ 19.001-77 — Общие положения [17];

б) ГОСТ 19.003-80 — Схемы алгоритмов и программ. Обозначение условные графические;

в) ГОСТ 19.101-77 — Виды программ и программных документов;

г) ГОСТ 19.102-77 — Стадии разработки [18];

д) ГОСТ 19.103-77 — Обозначение программ и программных документов;

е) ГОСТ 19.104-78 — Основные надписи;

ж) ГОСТ 19.105-78 — Общие требования к программным документам;

з) ГОСТ 19.106-78 — Требования к программным документам, выполненным печатным способом;

и) ГОСТ 19.201-78 — Техническое задание. Требования к содержанию и оформлению;

к) ГОСТ 19.202-78 — Спецификация. Требования к содержанию и оформлению;

л) ГОСТ 19.402-78 — Описание программы;

м) ГОСТ 19.404-79 — Пояснительная записка. Требования к содержанию и оформлению;

н) ГОСТ 19.502-78 — Описание применения. Требования к содержанию и оформлению;

о) ГОСТ 19.503-79 — Руководство системного программиста. Требования к содержанию и оформлению;

п) ГОСТ 19.504-79 — Руководство программиста. Требования к содержанию и оформлению.

Примечание — На практике студент также должен руководствоваться требованиями своего ***Задания*** и корпоративными ограничениями на проектирование.

Согласно требованиям к ТЗ на АС (см. ГОСТ 34.602-89 [9], раздел «1. ОБЩИЕ ПОЛОЖЕНИЯ»): «... Дополнительно могут быть разработаны ТЗ на части АС:

а) на подсистемы АС, комплексы задач АС и т. п. в соответствии с требованиями настоящего стандарта;

б) на комплектующие средства технического обеспечения и программно-технические комплексы в соответствии со стандартами ЕСКД и СРПП;

в) на программные средства в соответствии со стандартами ЕСПД;

г) на информационные изделия в соответствии с ГОСТ 19.201 и НТД, действующей в ведомстве заказчика АС».

Таким образом, согласно ГОСТ 19.102-77 [18], стадии и этапы работ по созданию ПО ИС представляются таблицей 1.4.

Таблица 1.4 — Стадии и этапы работ разработки ПО ИС [18]

Стадии разработки	***Этапы работ***
1. Техническое задание	Обоснование необходимости разработки программы. Научно-исследовательские работы. Разработка и утверждение технического задания.
2. Эскизный проект	Разработка эскизного проекта. Утверждение эскизного проекта.
3. Технический проект	Разработка технического проекта. Утверждение технического проекта.
4. Рабочий проект	Разработка программ. Разработка программной документации. Испытание программ.
5. Внедрение	Подготовка и передача программы.

В технически обоснованных случаях, что должно быть отражено в ЧТЗ на ПО ИС, допускается:

а) ***исключать*** вторую и третью стадии;

б) ***объединять*** и ***исключать*** этапы работ, а также вводить другие этапы работ по согласованию с заказчиком.

Конкретная необходимость использования ГОСТ серии ***19.xxx*** должна определяться дополнительно.

1.6 Учебная инфраструктура проектировщика ИС

Излишне напоминать, что любая работа, в том числе и учебная, всегда осуществляется в ограниченные сроки, по истечении которых студент должен представить результат этой работы на оценивание преподавателю или экзаменационной комиссии. В этих условиях поиск, изучение и практическая апробация необходимых инструментальных средств может занять значительное время и негативно сказаться на результатах работы.

Учебная цель данного подраздела — краткое описание конкретного состава рабочей среды и программных инструментальных средств, образующих учебную инфраструктуру проектировщика ИС и используемых для демонстрации учебных примеров в данном учебном пособии.

> Примечание — Для проведения проектных работ студенту необходима конкретная учебная программно-аппаратная инфраструктура, где такие работы могли бы быть выполнены.

Для обеспечения учебного процесса используется инфраструктурная среда ОС УПК АСУ [19], с которой студент хорошо знаком по результатам обучения дисциплине «*Операционные системы*» (*ОС*).

Непосредственно для изучаемой дисциплины подготовлена специальная рабочая область пользователя ***upk***, которая предоставляется студенту индивидуально, а также используется и сохраняется им в течение всего процесса обучения.

> Примечание — Студенту необходим учебный инструментальный прототип будущей инфраструктуры размещения проектируемой ИС, которая затем могла бы быть перемещена в производственную среду предприятия (организации).

Согласно общим современным представлениям проектируемая ИС должна представлять собой трёхзвенную распределенную сервис-ориентированную систему (см. рисунок 1.10). Учитывая, что важнейшей функцией ИС является наглядное и удобное для восприятия человеком представление информации, учебный процесс данной дисциплины ориентируется прежде всего на хорошо развитый аппарат Web-сервисов. Этот аппарат хорошо зарекомендовал себя на практике и успешно используется в учебном процессе вуза, например, портал ТУСУР.

Среди открытых и свободно доступных для скачивания, установки и последующего использования инструментальных средств является продукция организации-фонда ***Apache Software Foundation*** (ASF) [20], известного десятками своих программных изделий. Наиболее популярным из них является контейнер сервлетов ***Apache Tomcat*** [21], способный реализовывать как простейшие Web-сервисы, так и выполнять функции web-сервера.

Для целей обучения и основы прототипа проектируемой ИС будут использоваться два инструментальных средства ASF:

а) *сервер приложений* ***Apache TomEE*** [22], обеспечивающий распределённое размещение приложений будущей ИС;

б) *СУБД* ***Apache Derby*** [23], обеспечивающая приложения ИС как встроенным (*embedded*) вариантом использования, так и вариантом использования доступным через сеть.

Все указанные инструменты реализованы на основе языка Java, что обеспечивает им высокую степень переносимости между различными ОС и богатым арсеналом имеющихся приложений на языке Java.

Примечание — Студенту необходимы учебные инструментальные средства разработки ПО, с помощью которых он мог бы проводить как реализацию и тестирование ИС в целом, так и реализацию и тестирование её отдельных компонентов.

Среди бесплатных и широко доступных инструментальных средств разработки ПО, к тому же сами реализованные на языке Java, являются продукты некоммерческой организации ***Eclipse Foundation*** [24]. На момент написания данного учебного пособия эта организация предлагает тринадцать дистрибутивов различного назначения. В частности, в учебной инфраструктуре ОС УПК АСУ [19] установлен дистрибутив ***Eclipse IDE for C/C++ Developers***, используемый при изучении дисциплины «*Операционные системы*».

Непосредственно для проектных задач изучаемой дисциплины установлен и используется дистрибутив ***Eclipse IDE for Enterprise Java Developers*** [25], предназначенный для разработки на языке Java приложений уровня предприятия и хорошо интегрируемый с сервером приложений ***Apache TomEE***.

В качестве справочника или учебника по языку Java рекомендуется использовать достаточно полное руководство [26].

Примечание — Студенту необходимы учебные инструментальные средства концептуального проектирования ИС и подготовки проектной документации.

Современное проектирование ИС ориентировано на широкое использование методологии SADT, хорошо описанной в разных источниках, например, [5]. Эта методология требует построения различных диаграмм, необходимых как на этапах концептуального проектирования, так и на этапах реализации ИС.

Непосредственно в данной учебной дисциплине используются только свободно доступные продукты:

а) **Ramus Educational**, разработанный ***Ramus Soft Group*** [27] для целей построения диаграмм моделей IDEF0 и DFD;

б) **Eclipse Modeling Tools** [28], предназначенный для построения приложений на основе графических моделей.

Дополнительно для разных целей используются приложения стационарно установленного в ОС УПК АСУ ПО LibreOffice [29], продукты которого придерживаются открытого формата OpenDocument Format [30], а также широко известный браузер Mozilla Firefox [31].

Примечание — Проектировщик ИС должен хорошо знать состав, расположение и основные шаблоны применения своего инструментария.

Рассмотрим состав и варианты технологии применения инструментальных средств в отдельных пунктах данного подраздела.

1.6.1 Состав и размещение инструментальных средств

Учебная рабочая область студента имеет размер 400 МБ и находится в файле ***pis-home.ext4fs***, поэтому необходимо беречь его свободное пространство.

В целом, описание структуры и местоположение файлов ОС УПК АСУ достаточно полно описано в учебно-методическом пособии [16] и обычно рассматривается в материале первой лабораторной работы по дисциплине ОС, поэтому она здесь изучаться не будет. Далее мы считаем, что студент запустил учебную систему, подключил к ней личную рабочую об-

ласть и прошёл авторизацию пользователем ***upk***. В такой ситуации, размещение всех программных средств и их адресация видны и рассматриваются в общем древовидном адресном пространстве файловой системы. В таком виде и рассматривается размещение всех используемых в данном учебном пособии объектов.

Как проектировщик, студент должен хорошо представлять себе базовые адресные пространства, обозначенные следующими каталогами общими для многих UNIX-подобных ОС:

а) **/etc** — общий каталог размещения конфигурационных файлов;

б) **/opt** — общий каталог размещения дистрибутивов дополнительных инструментальных средств;

в) **/usr/bin** — общий каталог размещения запускаемых файлов и утилит ОС;

г) **/usr/lib/jvm** — общий каталог размещения дистрибутивов языка Java;

д) **/home/upk** — домашний каталог пользователя ***upk***, к которому подключена рабочая область студента.

Примечание — **Базовой областью** подключения необходимых дистрибутивов инструментальных средств проектировщика ИС является каталог ***/opt***, содержимое которого представлено таблицей 1.5.

Таблица 1.5 — Видимость ПО дистрибутивов инструментальных средств

Каталог	Содержимое каталога
/opt/derby	Дистрибутив Apache Derby
/opt/eclipseEE	Дистрибутив Eclipse Enterprise Edition
/opt/eclipseMT	Дистрибутив Eclipse Modeling Tools
/opt/tomee	Дистрибутив Apache TomEE

Непосредственное подключение перечисленных дистрибутивов к каталогам таблицы 1.5 — темы отдельных лабораторных работ. Главное, чтобы студент знал о них и мог использовать в настройках своих проектов.

Примечание — **Рабочей областью** использования учебного ПО проектировщика ИС является каталог ***/home/upk***, в котором выделены специальные области показанные в таблице 1.6.

Таблица 1.6 — Рабочие области ПО инструментальных средств пользователя upk

Каталог	Содержимое каталога
/home/upk/derby	Рабочая область СУБД Apache Derby
/home/upk/eclipseEE	Рабочая область среды разработки Eclipse Enterprise Edition
/home/upk/eclipseMT	Рабочая область дополнительной среды разработки Eclipse Modeling Tools
/home/upk/tomee	Рабочая область сервера приложений Apache TomEE

Существенное различие между базовой и рабочей областями следующее:

а) ***базовая область*** — образует видимую область дистрибутива установленного и обычно содержит достаточно большой объем ПО и может быть использована несколькими рабочими областями;

б) ***рабочая область*** — часть установленного дистрибутива и обычно содержит файлы конфигурации и результаты применения ПО дистрибутива, например, ПО СУБД

Apache Derby содержит в рабочей области сценарии запуска и остановки СУБД, а также каталоги размещения конкретных баз данных.

Проектная инфраструктура рабочей области пользователя ***upk*** должна иметь набор каталогов, общий список и назначение которых приведён в таблице 1.7.

Таблица 1.7 — Список каталогов проектной инфраструктуры ИС

Каталог	***Содержимое каталога***
/home/upk/bin	Каталог, в который разработчик ИС помещает разработанные им запускаемые программы.
/home/upk/lib	Каталог для размещения разработанных библиотек.
/home/upk/src	Каталог с исходными текстами дополнительно разработанных программ.
/home/upk/ramus-educational-1.1.1	Установленный дистрибутив и рабочая область системы Ramus Educational
/home/upk/Документы	Каталог для размещения учебных материалов по изучаемой дисциплине и другой документации проектировщика ИС.
/home/upk/Загрузки	Каталог размещения несистемно скачиваемых файлов.
/home/upk/Рабочий стол	Каталог размещения информации на рабочем столе пользователя.

Безусловно, проектная инфраструктура включает в себя и другие элементы размещения источников информации и конфигурационных файлов, отражающих особенности построения используемой ОС.

Примечание — **Проектирование** — это всегда достижение и строгое описание конкретной упорядоченности в размещении информации и данных.

Следует заметить, что описанное размещение учебной проектной инфраструктуры разработчика ИС не является строго обязательным. Многие её элементы можно расположить по другому, но они должны быть описаны и понятны студенту, поскольку используются как в самих процессах проектирования ИС, так и в оформлении соответствующей проектной документации.

1.6.2 Технология применения инструментальных средств проектирования

Примечание — Инструментальные средства разработчика ИС должны обеспечивать создание базовой трёхзвенной архитектуры «Клиент-сервер», которая представлена рисунком 1.10.

Хотя используемые инструменты проектирования могут учитывать особенности и специальные требования предметной области ИИС предприятия, они должны обеспечивать реализацию базовой трёхзвенной архитектуры, включающей представление информации, бизнес-логику и сохранение данных.

Инструментальная среда разработки ***Eclipse IDE for Enterprise Java Developers*** [25] имеет шаблон проектирования «***Dynamic Web Project***» интегрируемый с сервером приложений ***Apache TomEE*** и ***Apache Derby***. Такая конфигурация инструментальных средств позволяет сразу реализовывать простейшую трёхзвенную архитектуру Web-сервисов, схематично показанную на рисунке 1.11.

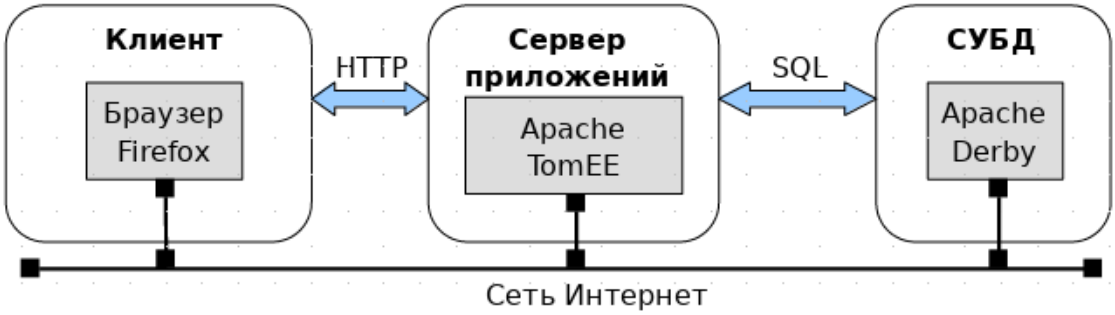

Рисунок 1.11 — Простейшая трёхзвенная архитектура Web-сервиса

Представленная архитектура Web-сервиса легко реализуется средствами учебной проектной инфраструктуры ОС УПК АСУ даже без подключения компьютера к внешней сети Интернет и рассматривается как базовая в учебном процессе данной дисциплины. Более того, используя хорошо описанные настройки, можно на одном компьютере установить несколько серверов приложений Apache TomEE и работать с ними.

Важнейшая особенность использования Apache TomEE состоит в том, что он:

а) сам имеет *большое количество реализованных технологий* создания распределенных приложений;

б) позволяет профессионально обеспечивать *мультиплексирование запросов* к нему (см. рисунок 1.12);

в) позволяет профессионально обеспечивать *демультиплексирование запросов* к различным СУБД (см. рисунок 1.13).

В среде Web-сервисов, реализация приведённых выше архитектур не вызывает затруднений, поскольку обеспечивается URL/URN-адресацией.

Примечание — Современные инструментальные средства разработчика ИС должны обеспечивать проектирование и создание распределенных систем.

Фактически архитектуры, показанные на рисунках 1.11 — 1.13, уже являются простейшими распределёнными системами.

В тех случаях, когда необходимо создать систему, распределённую на множестве функциональных центров (ФЦ) ИИС (см. рисунок 1.8), то на сервере приложений Apache TomEE следует:

а) ***реализовать*** множество независимых приложений, каждое из которых соответствует одному ФЦ;

б) ***связать*** независимые ФЦ через их интерфейсы, согласно поставленной задаче.

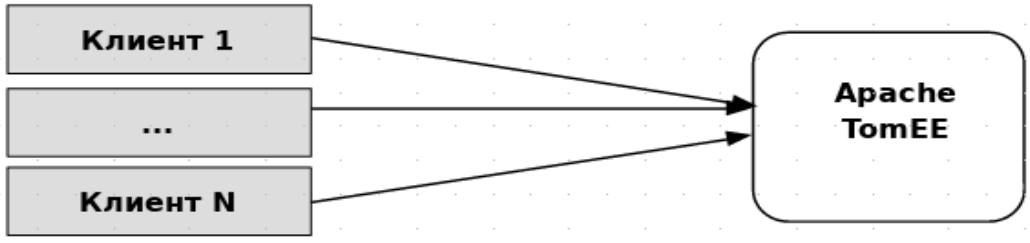

Рисунок 1.12 — Мультиплексирование запросов к серверу

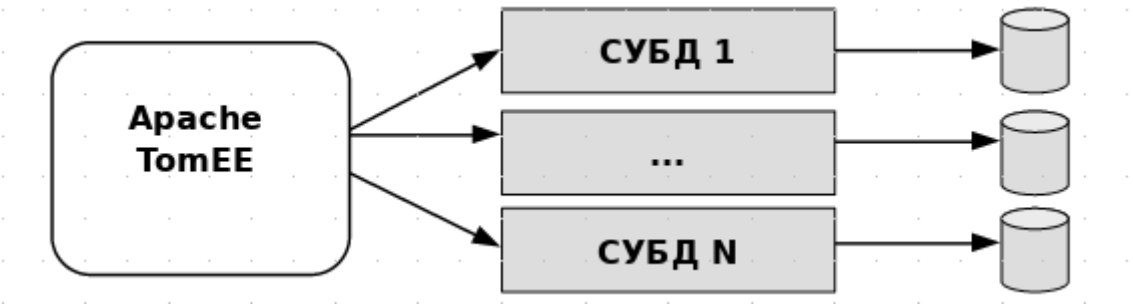

Рисунок 1.13 — Демультиплексирование запросов к СУБД

Двухзвенная часть учебной реализации такой распределенной системы показана на рисунке 1.14.

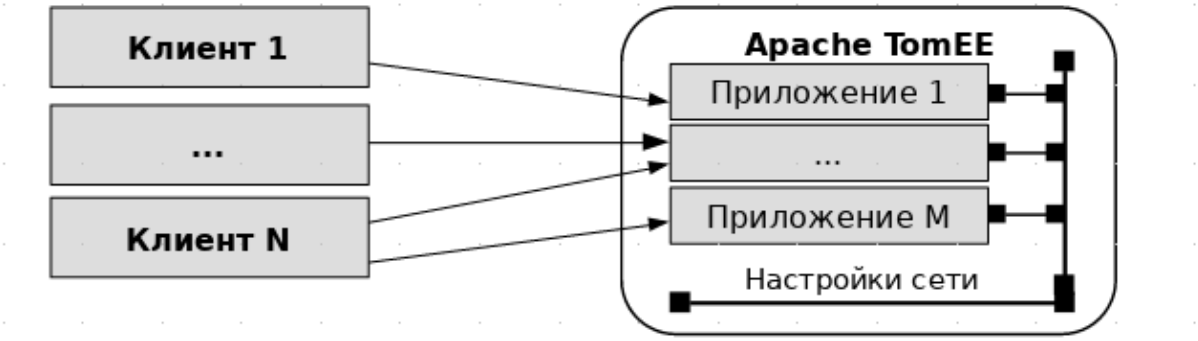

Рисунок 1.14 — Двухзвенная часть архитектуры распределённой системы

Естественно, что при необходимости такая двухзвенная система может быть дополнена требуемым доступом к СУБД.

> Примечание — Современные инструментальные средства разработчика ИС должны обеспечивать размещение компонент созданной системы в распределённой среде предметной области ИИС.

Поскольку доступ к Web-сервисам обеспечивается URL/URN-адресацией, то размещение отдельных приложений по компьютерам предметной области ИИС не вызывает затруднений. Здесь мы дополнительно учитываем возможность размещения на каждой машине сервера приложений Apache TomEE.

Что касается СУБД Apache Derby, то его размещение также не вызывает затруднений, поскольку он реализован на языке Java. Кроме того, сервер приложений способен работать практически с любыми типами СУБД.

> Примечание — **Современные распределённые системы** должны обеспечивать необходимый уровень безопасности работы с ними.

Сервер приложений Apache TomEE обеспечивает стандартный доступ к нему по протоколу HTTP. Дополнительными настройками можно обеспечить доступ по защищённому протоколу HTTPS. Имеются также инструменты для авторизации подключаемых пользователей и систем.

На этой позитивной ноте мы заканчиваем изучение теоретической части дисциплины и переходим к более практическим вопросам проектирования.

Вопросы для самопроверки

1. Какой перечень парадигм положен в основу изучаемой дисциплины? Перечислите их.
2. Какие две группы систем предлагает классификация СОД, например, в трактовке А.М. Ларионова?
3. Что такое — парадигма «Программа-массив» и в чём состоят её недостатки?
4. Как можно оценить сложность систем, придерживающихся концепции «Программа-массив»?
5. В чём состоит суть парадигмы информационного подхода в проектировании ИС?
6. Назовите основные части системного проектирования предметной области?
7. Что такое — методика SADT и какой набор стандартов был разработан специально для неё?
8. Назовите три наиболее известных стандарта, разработанных для проектирования по направлению ICAM Definitions?
9. Каким образом язык UML связан с информационным подходом проектирования ИС?
10. Каким образом технология СУБД связана с информационным подходом проектирования ИС?
11. Что такое — модель АСУ и на какие стандарты она опирается?
12. Что такое — ИС в пределах модели АС?
13. Перечислите стадии создания АС?
14. В чём состоит особенность парадигмы CALS-технологий?
15. Что такое — ИИС и где она используется?
16. На какие стандарты опирается понятие ЖЦИ?
17. Что такое — САПР и как она классифицируется?
18. Назовите базовые модели проектирования ИС.
19. Изобразите и поясните базовую трёхзвенную архитектуру распределённой системы «Клиент-сервер».
20. На основе какого сервера предлагается разрабатывать ИС?

П1 ЛАБОРАТОРНАЯ РАБОТА №1: Структура учебной части дистрибутива ОС УПК АСУ

Задача данной лабораторной работы является типичной для всех первых работ — установка и изучение структуры дистрибутива личной рабочей области, функционирующей под управлением ОС УПК АСУ.

Данная задача разбивается на две части:

1) **часть 1** предполагает получение у преподавателя архива рабочей области для данной дисциплины, размещение её на личном flashUSB студента и подключение, с целью выполнения работ от имени пользователя ***upk***; структура учебной инфраструктуры кратко описана в подразделе 1.6 данного учебного пособия;
2) **часть 2** предполагает краткое изучение инструментального средства Ramus Educational, что описано ниже — в подразделах П1.1 — П1.3; после установки Ramus в личной рабочей области, студенту необходимо запустить эту инструментальную систему и создать учебный проект с целью общего изучения правил работы с этим инструментом.

Замечание — **Результаты выполнения всех лабораторных работ** описываются в едином личном отчёте.

П1.1 Установка дистрибутива Ramus Educational

П1.1.1 Установка Ramus Educational в среду ОС MS Windows

Бесплатный дистрибутив системы *Ramus Educational* распространяется в виде установочного файла ***ramus-educational-1.1.1-setup.exe***, размером ***3.9*** МБайт.

Поместив дистрибутив в любое доступное для пользователя место, - следует запустить запустить его как исполняемый файл.

Появится стандартное окно инсталлятора ОС Windows и будут предложены:

а) *знакомство* и *согласие* с лицензией продукта;
б) *директория* установки дистрибутива;
в) *разрешение* на использование прав администратора ОС.

Получив утвердительные соглашения на все предложения, дистрибутив полностью установит систему, сразу же готовую для использования.

При отсутствии на компьютере программного обеспечения ***Java***, дистрибутив автоматически установит ***runtime***-систему ***Java*** версии 1.6.

Замечание — Указанный вариант установки проверен для ОС версий ***7.0*** и ***8.1***.

П1.1.2 Установка Ramus Educational в среду ОС Linux

Бесплатный дистрибутив системы *Ramus Educational* для ОС Linux распространяется в виде установочного ***jar***-архива ***ramus-educational-1.1.1-install.jar***, размером ***9.0*** МБайт, поэтому для его использования требуется уже предустановленная среда исполнения ***Java***, желательно версии - не ниже ***1.6***.

Поместив дистрибутив в корень домашней директории пользователя, следует в терминале выполнить команду:

```
java -jar ./ramus-educational-1.1.1-install.jar
```

В результате выполнения этой команды, появится окно с предложением прочитать и согласиться с лицензионным соглашением.

После соглашения с лицензией, дистрибутив системы будет установлен в директорию: ***~/ramus-educational-1.1.1***, которая станет домашней директорией дистрибутива.

Таким образом, в среде ОС Linux выполняется полностью локальная установка дистрибутива.

Рабочий запуск системы, осуществляется командой:

```
~/ramus-educational-1.1.1/bin/ramus
```

Замечание — Указанная выше установка системы проверена в среде ОС ***Arch Linux*** с графической оболочной ***Xfce4***. Учитывая большое разнообразие дистрибутивов ОС Linux, а также большое разнообразие используемых графических систем, невозможно гарантировать актуальность приведённой инструкции. В случае возникновения проблем, следует перейти к варианту использования ОС MS Windows.

Задание — Провести установку системы *Ramus Educational* в своей личной рабочей области пользователя ***upk***.

П1.2 Общее изучение системы Ramus Educational

Завершающим этапом выполнения самостоятельной лабораторной работы является изучение инструментальной среды системы ***Ramus***.

Цель такого изучения — обеспечить готовность обучающегося к применению инструментальной системы, как в процессе изучения теоретической части дисциплины, так и в процессах выполнения лабораторных работ и оформлении индивидуального отчёта.

Сам процесс обучения желательно разбить на два этапа:

а) ***сопровождение*** проектов системы;

б) ***манипуляция*** элементами диаграмм.

П1.2.1 Сопровождение проектов системы

Задача этого этапа — освоить надёжную работу с проектами ***Ramus***, поскольку потеря уже почти готового проекта приводит к значительным и непроизводительным затратам времени.

Сначала, необходимо научиться запускать систему. Для этого, желательно вывести соответствующую пиктограмму на рабочий стол пользователя.

После запуска системы, появляется окно, предлагающее открыть уже созданный проект, как показано на рисунке П1.1.

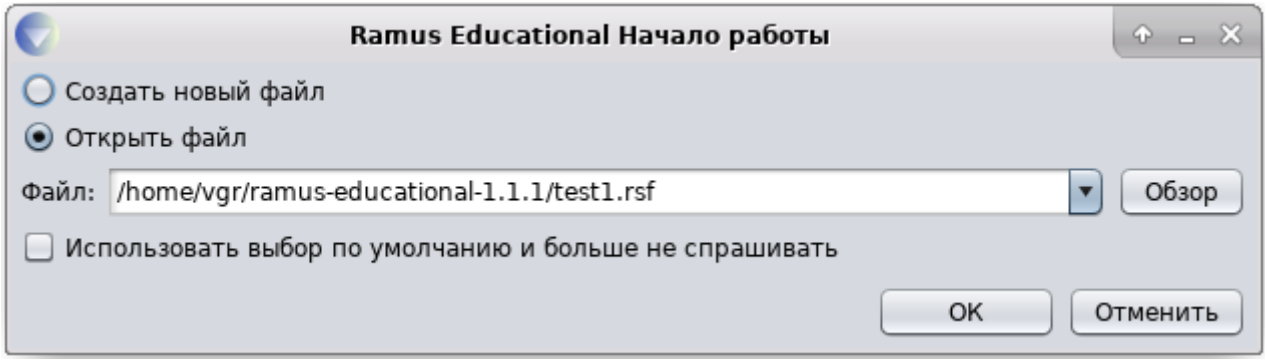

Рисунок П1.1 — Окно выбора проекта при запуске Ramus

Можно выбрать предлагаемый проект или найти нужный. Кнопка «***Обзор***» обеспечивает поиск по всей файловой системе. Файлы проектов имеют расширение ***.rsf***.

В случае создания нового проекта, запустятся окна дополнительных диалогов, которые также следует освоить.

Уже на примере рисунка П1.1 хорошо видно, что система Ramus поддерживает русифицированный интерфейс, поэтому самостоятельное изучение процесса загрузки системы не вызывает особых затруднений.

Далее, следует освоить элементы управления рабочим окном системы, первоначальный вид которого для открытого проекта ***test1.psf***, показан на рисунке П1.2.

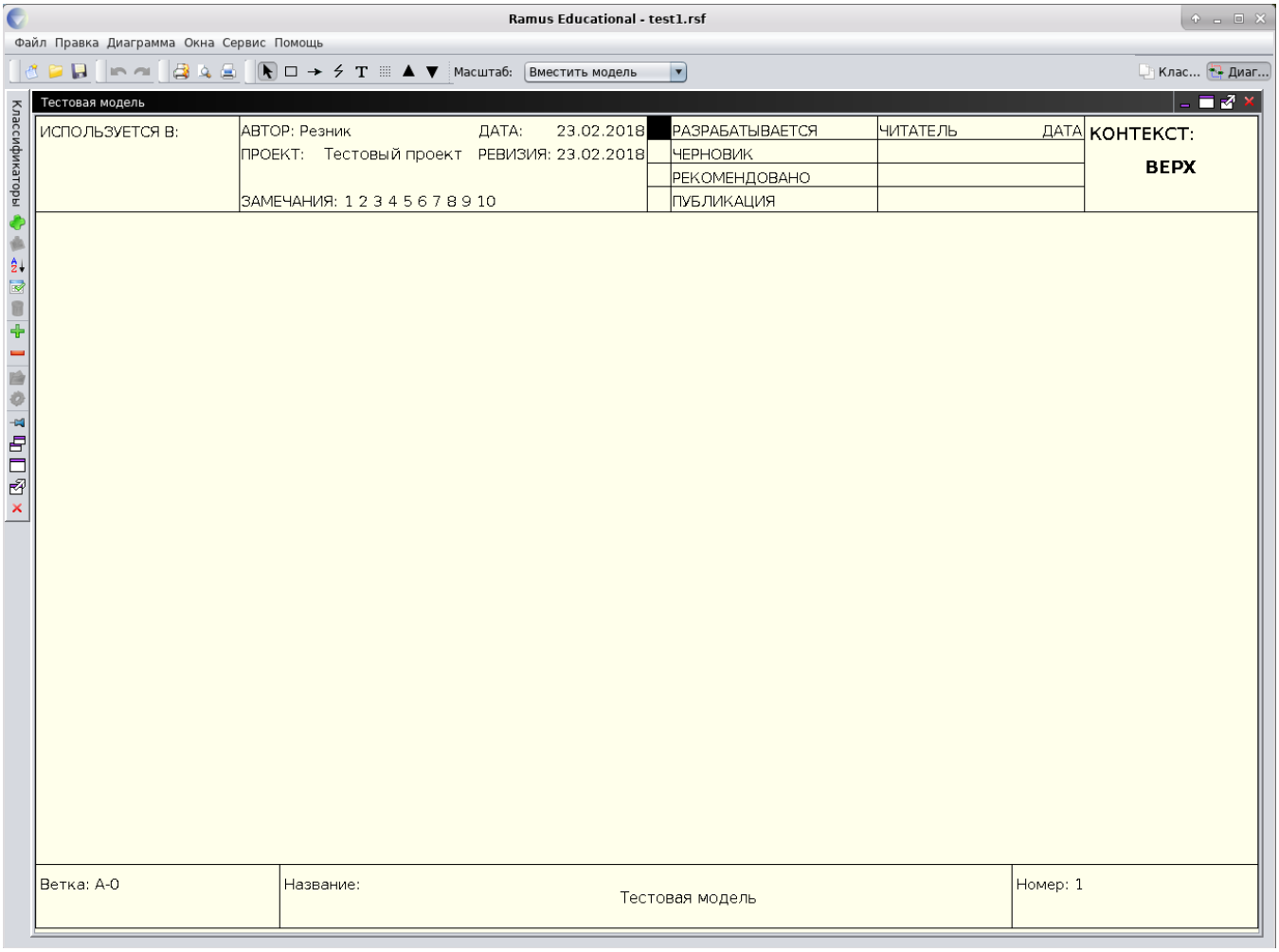

Рисунок П1.2 — Первоначальный вид нового проекта модели IDEF0

Пустой бланк диаграммы модели IDEF0 имеет заголовочную часть с данными, введёнными при создании проекта. Если установить на неё курсор мыши и нажать правую кнопку, то появится контекстное меню из двух пунктов:

- Свойства модели
- Свойства диаграммы

Эти пункты меню вызывают соответствующие окна диалога, позволяющие изменять надписи в заголовочной части бланка. Обычно это делается, если надписи были неправильно введены, во время создания проекта, или они требуют оперативной корректировки.

Следует освоить указанные средства изменения входных данных проекта.

В завершение этой части работы, следует освоить возможности пунктов главного меню «***Файл***», и прежде всего:

- пункты сохранения проекта;
- выход из системы Ramus.

П1.2.2 Манипуляция элементами диаграмм

Задача этого этапа — освоить общие средства манипулирования элементами проекта ***Ramus***.

Замечание — На этом этапе не следует проводить углублённое изучение состава и всех свойств графических элементов диаграмм IDEF0. Все это будет изучаться в последующих разделах. Главное, научиться пользоваться пунктами главного меню системы, а также элементами панели управления.

В главном меню системы следует выбрать пункт «***Помощь***», в котором активировать подпункт «***Справка***». Появится окно, содержащее общую документацию по системе Ramus, как показано на рисунке П1.3.

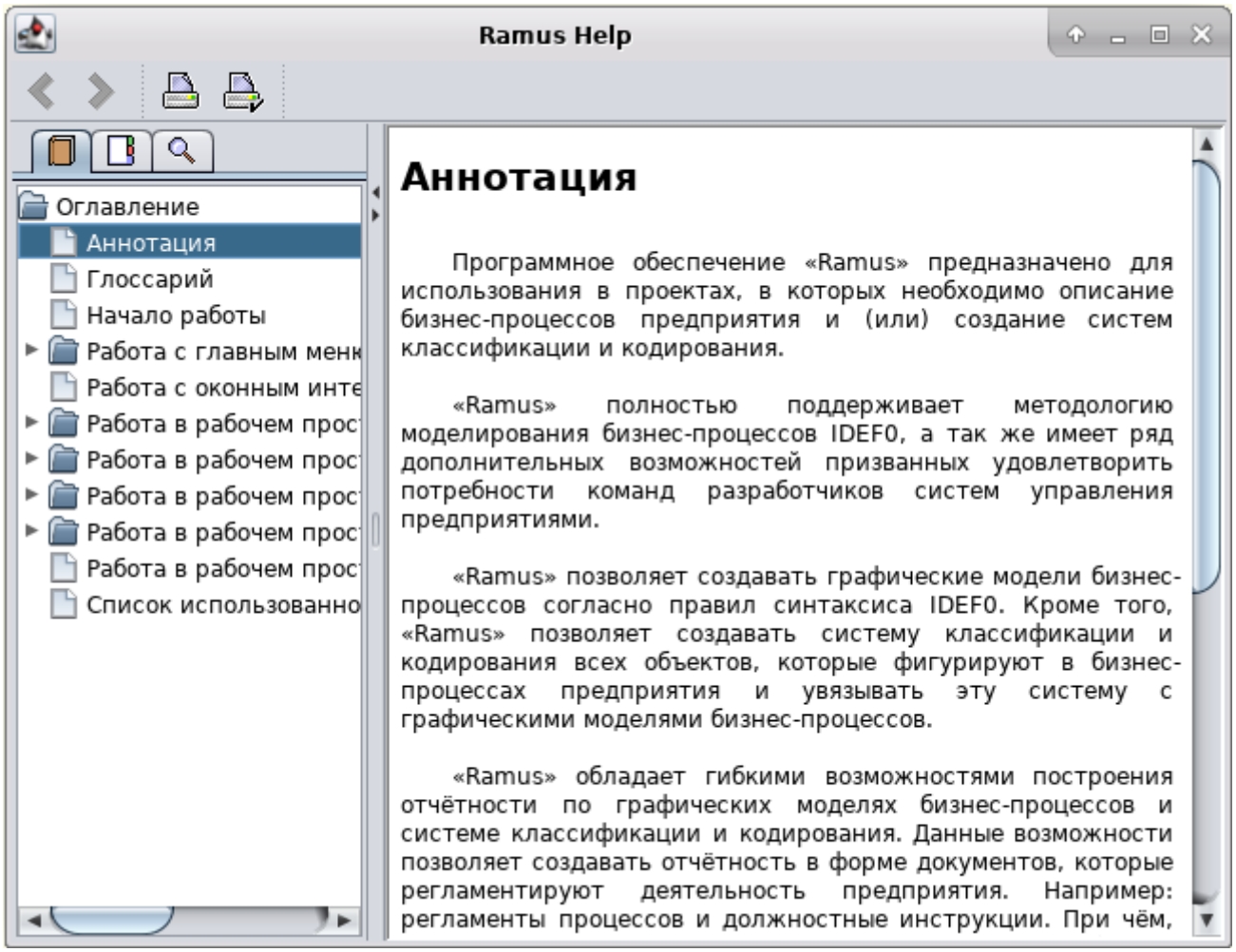

Рисунок П1.3 — Справочное окно системы Ramus

Следует, воспользовавшись окном документации:

а) освоить *приёмы* создания, именования и удаления: стрелок и блоков;
б) освоить *правила* создания и удаления диаграмм.

П1.3 Методические приёмы разработки функциональных моделей

Цель данного подраздела — определение ***места*** и ***роли*** обучающегося (как проектировщика) в процессах выработки концептуальных проектных решений.

П1.3.1 Организационные аспекты процесса проектирования

Прежде всего следует понимать, что концептуальное проектирование является не научным исследованием, а ***инженерной разработкой***, поэтому все решения должны быть ограничены уже *известными подходами* и иметь *разумное обоснование.*

Исходя из сказанного, выделим рассматриваемые вопросы, раскрывающие поставленную цель:

а) *формирование требований*, определяющее масштаб проектных решений;
б) *номинальный состав* участников проекта, формально определяющий состав и назначение участников выработки проектных решений;
в) *учебный состав* участников проекта, конкретизирующий состав участников проекта до уровня изучаемой дисциплины.

П1.3.2 Формирование требований

Формирование требований к ИС входит в группу работ, обычно называемых «***предпроектные исследования***». В разделе 1, эти исследования были названы группой стадий «***до ТЗ***»:

1) стадия 1 - «Формирование требований к АС»;
2) стадия 2 - «Разработка концепции АС».

Реально масштаб проектируемой ИС может не соответствовать уровню предприятия, но важно, чтобы этот масштаб был оценён достаточно адекватно.

Далее, на основе исходных данных требований к ИС, строится концептуальная модель «***As Is***» («*как есть»*), отражающая существующую систему. Проектные решения такой модели называют «***обратной разработкой***».

В случае, если проект не удовлетворяет требуемым целевым установкам ИС, в него вносятся изменения и, соответственно, строится новая концептуальная модель ИС. Проектные решения таких моделей называют «***To Be***» (*«как должно быть»*).

> Замечание — Проектные решения «***To Be***» не являются результатом простой фантазии или личного желания проектировщика. Над проектировщиком «всегда довлеет» ***внешняя целевая установка*** требований к ИС.

Непосредственно в пределах изучаемой дисциплины, обучающийся выбирает и согласует тему выпускной квалификационной работы. После утверждения темы, целевые требования к ИС становятся внешним фактором по отношению к обучающемуся и все его проектные решения оцениваются с этой позиции.

П1.3.3 Номинальный состав участников проекта

ГОСТ Р 50.1.028-2001 [8], стандартизирующий методику IDEF0, определяет следующий состав проектной группы:

а) **Руководитель проекта** - должностное лицо, осуществляющее административное управление проектом; утверждает проект, а также переводит его в состояния: РАЗРАБАТЫВАЕТСЯ, ЧЕРНОВИК, РЕКОМЕНДОВАНО и ПУБЛИКАЦИЯ;

б) **Авторы (разработчики) модели** - лица, создающие IDEF0-модели на основе материала, собранного из источников информации;

в) **Технический совет** - элемент организации процесса создания моделей, предлагающий арбитражные решения по моделированию и рекомендации по установлению статуса диаграмм, части и/или модели в целом;

г) **Эксперт** - выбираемое руководителем проекта лицо, обладающее специальными знаниями некоторых аспектов моделируемой области;

д) **Библиотекарь** - лицо, ответственное за хранение документации, изготовление копий, координацию обмена письменной и/или электронной информацией: *рассылка папок, получение рецензий, регистрация и публикация диаграмм и модели*;

е) **Источники информации** - *исходная информация* для IDEF0-модели поступает к разработчику из разных источников: *от людей и от документов*.

На рисунке П1.4 представлен *номинальный состав* проектной группы, предлагаемый стандартом.

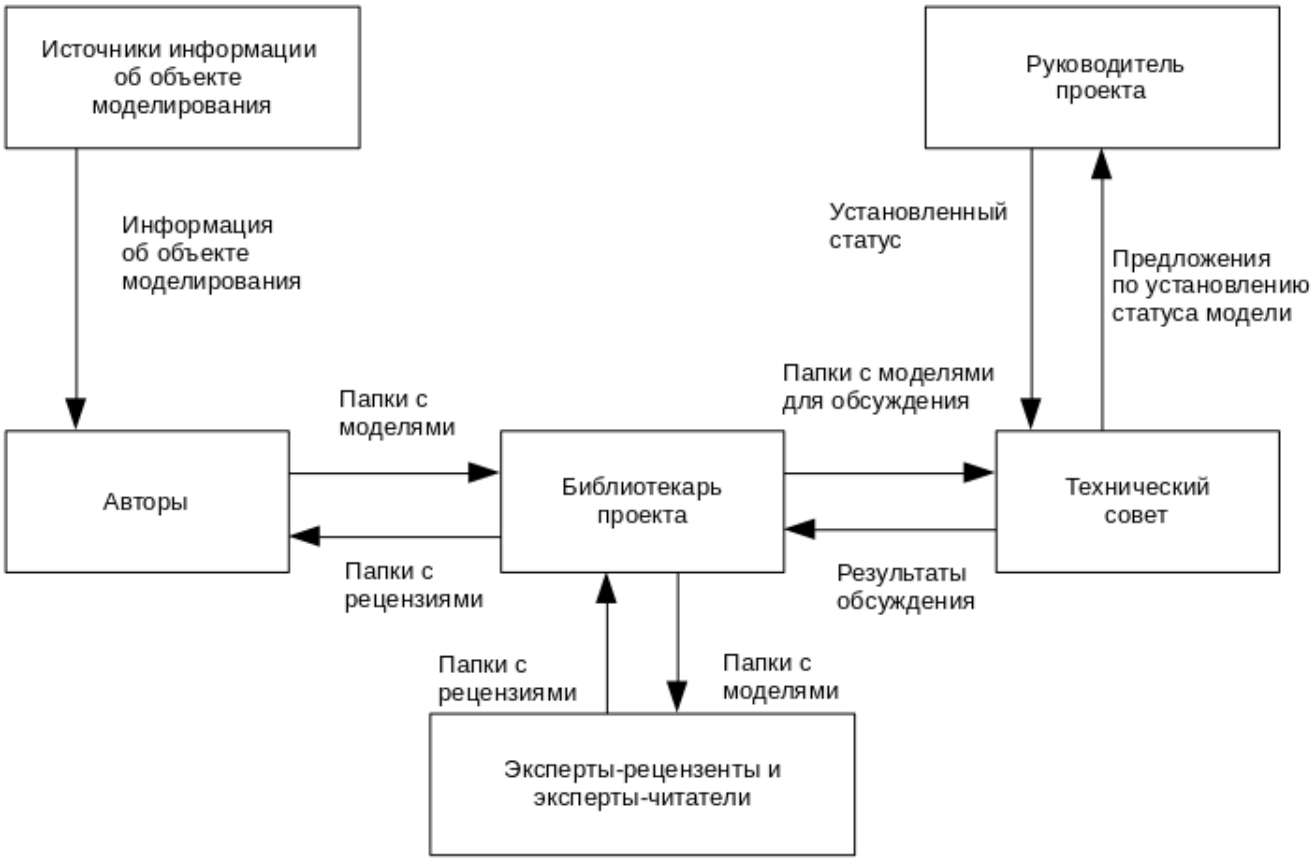

Рисунок П1.4 — Состав участников проектной группы

Хорошо видно, что на авторов (проектировщиков) воздействует три источника внешних целевых установок:

а) *источники информации* об объекте моделирования, которые проектировщик изучает в виде письменных документов;
б) *письменные замечания* и предложения экспертов-рецензентов;
в) *письменные замечания* и предложения членов технического совета.

В такой ситуации значительно снижаются:

а) ***значимость*** субъективных решений проектировщика на результаты проектирования;
б) ***персональная ответственность*** проектировщика за принятые проектные решения.

Замечание — Рисунок П1.4 отражает номинальный (эталонный) состав участников группы, рекомендуемый ГОСТ. Реальная ситуация, в которой находится проектировщик ИС может существенно отличаться от эталонной, что естественным образом увеличивает: *значимость* субъективных решений проектировщика; *персональная ответственность* проектировщика за принятые проектные решения.

П1.3.4 Практическое освоение системы Ramus

Выполнив пункты данного подраздела, следует просмотреть содержимое трёх источников информации [1 - 3], перечисленных в подразделе П1.4.

Далее необходимо:

1) выбрать нужный для вас источник информации, в котором необходимо выбрать пример создания диаграмм по методологии IDEF0;
2) реализовать выбранный пример в своей рабочей области пользователя upk;
3) отразить содержимое проделанной работы в личном отчёте.

Замечание — Нет необходимости разбираться в алгоритмах выбранной задачи и вдумываться в прикладную сущность создаваемых систем.
Главная задача — освоить саму технологию работы с системой Ramus.

П1.4 Список использованных источников

1. Аксенов К.А., Работа с CASE-средствами BPwin, Erwin. - Екатеринбург, 2004. - 50 с. [Файл] Method_BpWin_Erwin.pdf.
2. Грекул В.И., Проектирование информационных систем. Практикум: Учебное пособие / В.И. Грекул, Н.Л. Коровкина, Ю.В. Куприянов — М.: Национальный Открытый Университет «ИНТУИТ», 2012. — 187 с. [Файл] Grekoul.pdf.
3. Ramus - Методические указания.pdf.