[1 ОСНОВЫ ПОСТРОЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ (АС)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758273)

[1.1 Определение и классификация АС и их компонентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758274)

[1.1.1 Понятие системы и основные свойства систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758275)

[1.1.2 Автоматизированные системы и их классы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758276)

[*1.1.2.1 Определение понятия автоматизированной системы*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758277)

[*1.1.2.2 Классификация автоматизированных систем*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758278)

[1.1.3 Функциональные и обеспечивающие подсистемы АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758279)

[1.2 Основные этапы развития и современное состояние автоматизированных систем и информационных технологий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758280)

[1.2.1 История развития АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758281)

[1.2.2 Общая характеристика современного состояния информационных технологий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758282)

[1.2.3 Влияние информационных технологий и автоматизированных систем на деятельность предприятий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758283)

[1.3 Интегрированная автоматизированная система управления как пример большой сложной АС и ее зарубежные аналоги](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758284)

[1.3.1 Определение интегрированной автоматизированной системы управления предприятием](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758285)

[1.3.2 Принятая за рубежом концепция автоматизации ICAM (CIM)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758286)

[1.3.3 Характеристика MRP-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758287)

[1.3.4 Характеристика MRPII-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758288)

[1.3.5 Характеристика ERP-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758289)

[1.3.6 Характеристика CSRP-систем как компонентов ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758290)

[1.3.7 Характеристика  ERPII-систем как компонентов ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758291)

[1.3.8 Экономико-математические методы и модели в компонентах ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758292)

[1.3.9 Характеристика рынка автоматизированных систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758293)

[1.4 Системотехнические аспекты проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758294)

[1.4.1 Определение понятий «проект» и «проектирование АС»](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758295)

[1.4.2 Требования к АС и модель FURPS+](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758296)

[1.4.3 Предпосылки и принципы создания АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758297)

[*1.4.3.1 Основные предпосылки создания АС и участники этого процесса*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758298)

[*1.4.3.2 Основные принципы создания АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758299)

[*1.4.3.3 Принципы McKinsey успешного построения АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758300)

[1.4.4 Понятие жизненного цикла системы и виды моделей жизненного цикла](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758301)

[1.4.5  Определение понятий методологии и технологии проектирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758302)

[1.4.6 Сущность внешнего и внутреннего проектирования систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758303)

[1.4.7  Эволюция подходов к разработке АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758304)

[1.4.8  Общая характеристика современной методологии создания АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758305)

**1 ОСНОВЫ ПОСТРОЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ (АС)**

***1.1 Определение и классификация АС и их компонентов***

***“Уточните значение слов, и вы избавите  
человечество от большей части его заблуждений”***

***Рене Декарт***

**1.1.1 Понятие системы и основные свойства систем**

Термин «система» обозначает как реальные, так и абстрактные объекты и широко используется для образования других понятий, например банковская система, информационная система, кровеносная система, политическая система, система уравнений и др.

Известно большое число определений понятия «система», используемых в зависимости от контекста, области знаний и целей исследования.

**Систе́ма** (от др.-греч. σύστημα — «сочетание»)  — множество взаимосвязанных элементов, обособленное от среды и взаимодействующее с ней, как целое.

**Система** — это конечное множество функциональных элементов и отношений между ними, выделенное из среды в соответствии с определенной целью в рамках определенного временного интервала.

Системы обладают рядом свойств, которые можно разбить на несколько групп, которые представлены в таблице 1.1.

Таблица 1.1

Свойства систем с разбивкой на группы

|  |  |  |
| --- | --- | --- |
| № | Название свойства | Характеристика свойства |
| **Свойства систем, связанные с целями и функциями** | | |
| 1 | Синергичность | максимальный эффект деятельности системы достигается только в случае максимальной эффективности совместного функционирования её элементов для достижения общей цели. |
| 2 | Эмерджентность | появление у системы свойств, не присущих элементам системы; принципиальная несводимость свойства системы к сумме свойств составляющих её компонентов (неаддитивность). |
| 3 | Мультипликативность | и позитивные, и негативные эффекты функционирования компонентов в системе обладают свойством умножения, а не сложения. |
| 4 | Целенаправленность | наличие у системы цели (целей) и приоритет целей системы перед целями её элементов. |
| 5 | Альтернативность | наличие нескольких путей функционирования и развития (включая самоорганизацию и самосовершенствование). |
| **Свойства систем, связанные со структурой** | | |
| 1 | Структурность | упорядоченность элементов системы, при которой возможна ее декомпозиция на компоненты и установление связей между ними. |
| 2 | Иерархичность | каждый компонент системы может рассматриваться как система; сама система также может рассматриваться как элемент некоторой надсистемы (суперсистемы). |
| **Свойства систем, связанные с ресурсами и особенностями взаимодействия со средой** | | |
| 1 | Коммуникативность | существование сложной системы коммуникаций со средой. |
| 2 | Адаптивность | стремление к состоянию устойчивого равновесия, которое предполагает адаптацию параметров системы к изменяющимся параметрам внешней среды. |
| 3 | Надёжность | способность системы сохранять свой уровень качества функционирования при установленных условиях за установленный период времени. |
| 4 | Интерактивность | способность обмениваться информацией с внешними и внутренними объектами. |
| 5 | Эквифинальность | способность системы достигать состояний, не зависящих от исходных условий и определяющихся только параметрами системы. |

В качестве особого свойства систем следует выделить их подчинение закону необходимости разнообразия (**закону Эшби**): система должна обладать возможностью изменять своё состояние в ответ на возможное возмущение; разнообразие возмущений требует соответствующего ему разнообразия возможных состояний. В противном случае система не сможет отвечать задачам управления, выдвигаемым внешней средой, и будет малоэффективной и/или нежизнеспособной.

**1.1.2 Автоматизированные системы и их классы**

***1.1.2.1 Определение понятия автоматизированной системы***

Приведем одно из множества возможных определений.

**Автоматизи́рованная систе́ма** (АС) — система, состоящая из персонала, комплекса средств автоматизации его деятельности и регламентов работы, реализующая технологии выполнения установленных функций.

**Автоматизация** — применение саморегулирующихся технических средств, экономико-математических методов и систем управления, освобождающих человека от участия в процессах получения, преобразования, передачи и использования энергии, материалов или информации либо существенно уменьшающих степень этого участия или трудоёмкость выполняемых операций. Требует дополнительного применения датчиков (сенсоров), устройств ввода, управляющих устройств (контроллеров), исполнительных устройств, устройств вывода, использующих электронную технику и методы вычислений, иногда копирующие нервные и мыслительные функции человека.

Термины «автоматический» и «автоматизированный» близки по сути, т.к. в их основе лежит понятие «автоматизация», при этом термин «автоматизированный» подчеркивает относительно большую степень участия человека в процессе.

Если автоматизируемый процесс связан в основном с обработкой информации, то такая система называется автоматизированной информационной системой (АИС). Согласно широкому определению такой системы М.Р.Когаловским, это «комплекс, включающий вычислительное и коммуникационное оборудование, программное обеспечение, лингвистические средства и информационные ресурсы, а также системный персонал и обеспечивающий поддержку динамической информационной модели некоторой части реального мира для удовлетворения информационных потребностей пользователей».

АИС в узком смысле - программно-аппаратная система, предназначенная для автоматизации целенаправленной деятельности конечных пользователей, обеспечивающая, в соответствии с заложенной в нее логикой обработки, возможность получения, модификации, хранения и выдачи информации.

Важное примечание. «Ручные АИС» («без компьютера») существовать не могут, поскольку существующие определения предписывают *обязательное* наличие в составе любой информационной системы аппаратно-программных вычислительных средств, реализующих соответствующие информационные технологии (ИТ). Вследствие этого понятия «автоматизированная информационная система» (АИС), «компьютерная информационная система» (КИС) и просто «информационная система» (ИС) или «автоматизированная система» (АС) являются синонимами. Далее для их краткого обозначения будет использоваться аббревиатура АС.

***1.1.2.2 Классификация автоматизированных систем***

Наиболее существенным признаком классификации АС является **классификация АС по архитектуре.**В рамках этой классификации по степени распределённости различают:

·        *настольные* (*desktop*), или *локальные* АС, в которых все компоненты (базы данных, СУБД, клиентские приложения) находятся на одном компьютере;

·        *распределённые* (*distributed*) АС, в которых компоненты распределены по нескольким компьютерам.

Распределённые АС, в свою очередь, разделяют на:

·        *файл-серверные* АС (АС с архитектурой «файл-сервер»);

·        *клиент-серверные* АС (АС с архитектурой «клиент-сервер»).

В файл-серверных АС база данных находится на файловом сервере, а СУБД и клиентские приложения находятся на рабочих станциях.

В клиент-серверных АС база данных и СУБД находятся на сервере, а на рабочих станциях находятся клиентские приложения.

В свою очередь, клиент-серверные АС разделяют на *двухзвенные* и *многозвенные*.

В двухзвенных (англ. *two-tier*) АС всего два типа «звеньев»: сервер баз данных (БД), на котором находятся БД и СУБД (back-end), и рабочие станции, на которых находятся клиентские приложения (front-end). Клиентские приложения обращаются к СУБД напрямую.

В многозвенных (англ. *multi-tier*) АС добавляются промежуточные «звенья»: серверы приложений (*application servers*). Пользовательские клиентские приложения не обращаются к СУБД напрямую, они взаимодействуют с промежуточными звеньями. Типичный пример применения многозвенности —веб-приложения, использующие базы данных. В таких приложениях помимо звена СУБД и клиентского звена, выполняющегося в веб-браузере, имеется как минимум одно промежуточное звено — веб-сервер с соответствующим серверным программным обеспечением (ПО).

В таблице 1.2 приведены примеры классификации АС по другим признакам: по характеру обработки данных, по охвату задач (масштабности), по сфере применения.

Таблица 1.2

Примеры классификации АС

|  |  |  |
| --- | --- | --- |
| № | Название системы | Характеристика системы |
| **Классификация АС по характеру обработки данных** | | |
| 1 | *Информационно-справочная* или *информационно-поисковая АС* | нет сложных алгоритмов обработки данных, а целью системы является поиск и выдача информации в удобном виде |
| 2 | *АС обработки данных*, или *решающие АС* | данные подвергаются обработке по сложным алгоритмам;  к таким системам в первую очередь относят автоматизированные системы управления (АСУ) и автоматизированные системы поддержки принятия решений (АСППР), в том числе экспертные системы |
| **Классификация по охвату задач (масштабности)** | | |
| 1 | *Персональная* АС | предназначена для решения некоторого круга задач одного человека, например, автоматизированное рабочее место (АРМ) заведующего складом. |
| 2 | *Групповая* АС | ориентирована на коллективное использование информации членами рабочей группы или подразделения, например, АСУ цеха. |
| 3 | *Корпоративная АС* | в идеале охватывает все информационные процессы целого предприятия, достигая их полной согласованности, безызбыточности и прозрачности. Такие системы иногда называют системами комплексной автоматизации предприятия или интегрированными АСУ. |
| **Классификация по сфере применения** | | |
| 1 | *Экономическая АС* | предназначена для выполнения функций управления на предприятии, занимающемся производственной, торговой, финансовой или иной деятельностью в сфере экономики. |
| 2 | *Медицинская АС* | предназначена для использования в лечебном или лечебно-профилактическом учреждении. |
| 3 | *Географическая АС* | обеспечивает сбор, хранение, обработку, доступ, отображение и распространение пространственно-координированных данных (пространственных данных). |
| 4 | *Военная АС* | используется для управления войсками. |
| 5 | *Другие АС* | АС создаются для удовлетворения информационных потребностей пользователей в рамках конкретной предметной области. Каждой предметной области или сфере применения соответствует свой тип АС. Перечислять все эти типы не имеет смысла, так как количество предметных областей очень велико. |

Примечание. В данном курсе рассмотрение проблематики проектирования АС чаще всего будем проводить на примере экономических систем (объектом автоматизации в этом случае является производственное предприятие, фирма, компания и т.п.). Такие системы являются одними из самых распространенных и востребованных на практике. Их актуальность особенно высока в связи с происходящими в нашей стране социально-экономическими реформами и повышением роли бизнеса в развитии экономики. Однако, по мере необходимости, вопросы проектирования будут рассматриваться на примере АС из других предметных областей.

**1.1.3 Функциональные и обеспечивающие подсистемы АС**

Системные свойства структурности и иерархичности позволяют применить по отношению к АС метод декомпозиции и использовать понятие ранга системы. В результате декомпозиции системы в ее составе выделяются более простые составные элементы – подсистемы. *Подсистема* — система, являющаяся частью другой системы и способная выполнять относительно независимые функции, имеющая подцели, направленные на достижение общей цели системы.

С другой стороны, для системы может существовать система более высокого ранга – *Надсистема* (*суперсистема*) — более крупная система, частью которой является рассматриваемая система.

Описанная декомпозиция особенно полезна при работе с большими и сложными системами, так как позволяет заменить решение одной большой задачи анализа АС решением серии меньших задач. Глубина декомпозиции ограничивается целями исследования системы. В современных методиках типичной является декомпозиция на глубину 5-6 уровней.

Наиболее часто применяются:

функциональная декомпозиция АС, при которой основанием разбиения на функциональные подсистемы служит общность функций, выполняемых группами элементов;

структурная декомпозиция АС, для которой признак выделения подсистем — сильная связь между элементами по одному из типов отношений (связей), существующих в системе (информационных, логических, иерархических, энергетических и т. п.).

Часто декомпозиция осуществляется по нескольким основаниям, порядок их выбора зависит от квалификации и предпочтений системного аналитика.

Применительно к АС это приводит к выделению двух групп подсистем: функциональных и обеспечивающих (последние еще называют видами обеспечения АС, так как они обеспечивают возможности для нормальной работы функциональных подсистем).

Набор функциональных подсистем зависит от специфики основной системы. Например, в составе АСУ предприятия (АСУП) можно выделить следующие функциональные подсистемы:

технико-экономического планирования (ТЭП),

управления технической подготовкой производства (УТПП),

оперативного управления основным производством (ОУОП),

материально-технического снабжения (МТС),

сбыта и реализации продукции (СРП),

управления кадрами (УК),

управления финансами (УФ),

бухгалтерского учета (БУ) и др.

В рамках каждой функциональной подсистемы в ходе дальнейшей декомпозиции выделяют функциональные задачи, возникающие при реализации соответствующей функции и решаемые с заданной периодичностью. Иногда группы связанных между собой задач объединяют в комплексы задач. В таблице 1.3 приведены примеры функциональных задач для нескольких подсистем АСУП.

Таблица 1.3

Примеры функциональных задач подсистем АСУП.

|  |  |
| --- | --- |
| Подсистемы | Задачи |
| ТЭП | Расчет производственной программы, |
| Расчет нормативной себестоимости |
| Расчет нормативной калькуляции |
| Расчет нормативной трудоемкости производственной программы |
| УТПП | Расчет специфицированных норм расхода материалов на изделие, |
| Расчет нормативной трудоемкости и нормативной заработной платы на изделие |
| Расчет циклов и трудоемкости обработки деталей по цехам |
| ОУОП | Управление производством деталей, |
| Управление производством заготовок |
| Управление рабочими местами |

Набор видов обеспечения обычно типовой и, как правило, включает: организационное обеспечение (ОО), информационное обеспечение (ИО), программное обеспечение (ПО), техническое обеспечение (ТО), лингвистическое обеспечение (ЛО), правовое обеспечение (ПрО) и др.

***1.2 Основные этапы развития и современное состояние автоматизированных систем и информационных технологий***

**1.2.1 История развития АС**

История создания АС началась в середине 20-го века. АС эволюционировали параллельно с развитием научно-технического прогресса в разных отраслях, но в первую очередь в области вычислительной техники и информационных технологий. Эволюцию АС представим на примере АСУ. В настоящее время выделяют четыре поколения (или эры) АСУ. В табл. 1.4 представлены характеристики поколений с учётом качественных признаков.

*Основными причинами недостаточной эффективности АСУ I и II поколений* являются:

-      Неудовлетворительный общий уровень организации управления социалистическим предприятием и народным хозяйством в целом;

-      Отсутствие или низкий уровень подготовленных кадров на объектах внедрения систем;

-      Отставание в подготовке информационной базы;

-      Преимущественное использование централизованной обработки информации, приводящее к низкой оперативности решений многих задач в связи с отсутствием развитых средств связи;

-      Физическая разобщенность пользователей и ЭВМ;

-      Малая гибкость в эксплуатации.

Таблица 1.4

Характеристики поколений АСУ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Качественные признаки | Поколения АСУ | | | |
| I | II | III | IV |
| 1.     Уровень автоматиза-ции управ­ления | Автоматизация планово-экономических и учетно-функциональных функций, ориентирован­ных на традици­онные приемы и методы управле­ния производст­вом | Комплексы задач для функциональных подсистем управления программной деятельностью, реализуемых автоматизированными системами различного назначения (АСУП, АСУТП, САПР и др.) -  «островная автоматизация» | Интегрированные АСУ, охватывающие все производственные процессы от конструирования до реализации. | Гибкая адаптивная интегрированная система с элементами искусственного интеллекта, обеспечивающая безбумажное и безлюдное управление и перестройку при изменении параметров, ресурсов и продукции |
| 2.     Уровень функцио-нальной полноты | Простые программные, планово-расчетные и учетно-статистические функции, ориентированные на традиционные процедуры и методы управления | Оптимизация локальных планово-экономических функций, информационно-соответствующих режим организации основных функций управления, решение оперативных задач в диалоговом режиме | Интеграция локальных систем, использование методов оптимизационного и имитационного моделирования, экспертных систем и блоков обучения пользователей | Иерархическая, интегрированная самообучающаяся система, настраивающаяся на широкий класс объектов |
| 3.     Средства вычисли-тельной техники | Большие ЭВМ второго поколения, ограниченный набор периферийных устройств | Многомашинные вычислительные комплексы второго поколения, многофункциональные терминальные устройства, специализированные устройства контроля | Многомашинные многопроцессорные вычислительные комплексы третьего поколения, широкий спектр специализированных регистраторов информации и терминальных станций, персональные СОД, сети ЭВМ | СуперЭВМ четвертого поколения; сеть, объединяющая встроенные специализированные мини- и микро-ЭВМ, мобильные устройства. |
| 4.     Про-граммные средства | Пакетные ОС; прикладные программы, ориентированные на потребности конкретного объекта | Диалоговые ОС, СУБД функциональные пакеты прикладных программ | Система распределенной обработки информации, диалоговая интеллектуальная система, разработки программ и накопление знаний | Экспертные системы, системы управления базами знаний, инструментальные системы на основе языков высокого уровня, системы коллективного использования информации, средства Internet-программирования |
| 5.     Техно-логия создания программно-информаци-онных комплексов | Программирование в машинных кодах, применение алгоритмических языков программирования | Сборочное программирование, основанное на использовании библиотек типовых проектных решений, пакетов прикладных программ, модулей | Адаптация универсальных многопараметрических программ к конкретным условиям применения; интегральная CASE-технология создания | Диалоговые интеллектуальные автоматизированные системы разработки программ и накопления метазнаний;  CASE-технологии проектирования; Web-технологии проектирования |

*Характерными особенностями систем III и IV поколения* является модульная структура с возможностями разработки и внедрения системы по частям с последующим их развитием и наращиванием, а также наличие следующих свойств:

-        Функциональная полнота, обеспечивающая автоматизацию всех видов деятельности от технической подготовки производства до реализации готовых изделий;

-        Открытость в отношении наращивания состава функции и приспособляемость к изменению параметров объекта;

-        Вариантность алгоритмов и методов управления;

-        Наличие персональных средств, предоставленных в распоряжение пользователя;

-        Высокая скорость реакции;

-        Возможность работы с системой в активном режиме со стороны пользователя;

-        Децентрализация выполнения функций по функциональному, организационному и территориальному признакам;

-        Наличие средств управления распределенными процессами обработки данных и средств управления распределенными данными;

-        Наличие средств коммуникации разнородной вычислительной техники, обрабатывающего оборудования и промышленных контроллеров на базе стандартных сетевых протоколов.

Кроме того *системы IV поколения отличаются*:

-        Более совершенными средствами вычислительной техники, коммуникации и Интернет-технологий;

-        Большей гибкостью и адаптивностью;

-        Наличием средств искусственного интеллекта и экспертных систем принятия управленческих решений;

-        Возможностью самообучения;

-        Наличием блоков обучения пользователя.

Динамика изменения подхода к использованию АС на разных периодах представлена в таблице 1.5 и на рисунке 1.1.

Таблица 1.5

Изменение подхода к использованию автоматизированных систем

|  |  |  |  |
| --- | --- | --- | --- |
| **Период времени** | **Использование информации** | **Вид автоматизированных систем** | **Цель использования** |
| 1950 —1960 гг. | Бумажный поток расчётных документов | АС обработки расчётных документов на электромеханических бухгалтерских машинах | Повышение скорости обработки документов Упрощение процедуры обработки счетов и расчёта зарплаты |
| 1960 —1970 гг. | Основная помощь в подготовке отчётов | Управленческие АС для производственной информации | Ускорение процесса подготовки отчётности |
| 1970 —1980 гг. | Управленческий контроль производства и реализации | Системы поддержки принятия решений | Выработка наиболее рационального решения |
| 1980 гг. — настоящее время | Управление стратегией развития предприятия | Системы для высшего звена управления | Поддержка управления возможностями бизнеса |

Рис. 1.1 Изменение концепции ИС

С конца 20-го века концепция *использования* АС существенно изменяется. Они становятся стратегическим источником информации и используются на всех уровнях организации любого профиля. АС этого периода, предоставляя вовремя нужную информацию, помогают организации достичь успеха в своей деятельности, создавать новые товары и услуги, находить новые рынки сбыта, обеспечивать себе достойных партнёров, организовывать выпуск продукции по низкой цене и многое другое.

Роль информационных технологий (ИТ) также претерпевает существенные изменения даже в том случае, когда сам ключевой бизнес коммерческой компании или основная деятельность государственной организации принципиально остаются неизменными. Это обусловлено новыми возможностями ИТ в части доступности, распространенности, увеличения вычислительной мощности и производительности, интерактивности и т.д. Фактически, за время жизни одного поколения основной фокус применения информационных технологий сместился, как показано на рисунке 1.2, от автоматизации отдельных рутинных операций к преобразованию основной деятельности организации.

Рис. 1.2 Эволюция роли ИТ

**1.2.2 Общая характеристика современного состояния информационных технологий**

Современные ИТ, позволяющие создавать, хранить, перерабатывать, эффективно представлять данные и информацию, стали важным фактором конкурентоспособности и средством повышения эффективности управления всеми сферами общественной жизнедеятельности. Уровень информатизации (внедрения ИТ) является сегодня одним из главных факторов успешного развития всякого предприятия.

Cовременное состояние ИТ/АС можно охарактеризовать следующими положениями:

·  наличие большого количества программно-аппаратных комплексов и платформ для эффективного управления и сопровождения производства, промышленно функционирующих баз данных и хранилищ знаний большого объема, содержащих информацию по всем направлениям деятельности общества;

·  наличие технологий, обеспечивающих интерактивный доступ любого пользователя к информации и ресурсам — технической основой для этого служат открытые (Free) и корпоративные системы поиска информации (Information Retrieval Systems — IRS), государственные и коммерческие системы связи, глобальные (Global Network Systems) и региональные (RNS) информационно-вычислительные сети; международные соглашения, стандарты и протоколы обмена данными;

·  расширение функциональных возможностей ИТ, обеспечивающих распределенную работу баз и хранилищ данных с данными разнообразной структуры и содержания, мультиобъектных документов, гиперсред; создание локальных и интегрированных проблемно-ориентированных АС различного назначения на основе мощных серверов и локально-вычислительных сетей;

·  включение в АС экспертных систем (Expert System — ES), систем поддержки принятия решения (Decision Support System — DSS), систем поддержки исполнения (Executive Support System — ESS) и других технологий и средств.

Также можно выделить пять основных тенденций в развитии ИТ, которые представлены в табл. 1.6.

Таблица 1.6

Основные тенденции в развитии ИТ

|  |  |  |
| --- | --- | --- |
| **№** | **Тенденция** | **Описание** |
| 1 | *Глобализация* | Компании могут с помощью ИТ вести дела на мировом рынке, где угодно, немедленно получая исчерпывающую информацию. Происходит интернационализация программных средств и рынка информационного продукта. Получение преимуществ за счет постоянного распределения информационных расходов на более широкий географический регион становится необходимым элементом стратегии. |
| 2 | *Конвергенция* | Стираются различия между промышленными изделиями и услугами, информационным продуктом и средствами его получения, их профессиональным и бытовым использованием. Передача и прием цифровых, звуковых и видеосигналов объединяются в одних устройствах и системах. |
| 3 | *Усложнение информационных продуктов и услуг* | Информационный продукт в виде программно-аппаратных средств, баз и хранилищ данных, служб эксплуатации и экспертного обеспечения имеет тенденцию к постоянному развитию и усложнению. В то же время интерфейсная часть ИТ при всей сложности решаемых задач постоянно упрощается, делая все более комфортным интерактивное взаимодействие пользователя и системы. |
| 4 | *Способность к взаимодействию (Interoperability)* | Проблемы оптимального обмена данными между компьютерными информационными системами, между системой и пользователями, проблемы обработки и передачи данных и формирование требуемой информации приобрели статус ведущих технологических проблем. Современные программно-аппаратные средства и протоколы обмена данными позволяют решать их во все более полном объёме. |
| 5 | *Ликвидация промежуточных звеньев (Disintermediation)* | Развитие способности к взаимодействию однозначно ведет к упрощению доставки информационного продукта к потребителю. Становится ненужной цепочка посредников, если есть возможность размещать заказы и получать требуемое непосредственно с помощью ИТ. |

Наша страна постепенно входит в мировое экономическое пространство и неизбежно должна адаптироваться к законам его существования и развития. Глобализация и интегрированное развитие индустриальных экономик значительно расширяет возможности бизнеса. В развитых странах мира давно поняли, что именно информационные технологии и автоматизированные системы (ИТ/АС) обеспечивают мобильный доступ и аналитическую мощь, которые удовлетворяют потребности в руководстве предприятиями и проведении торговли в масштабе стран и континентов. Глобальная связь и системы управления доставляют потребителю информацию о предложениях, качестве и ценах и позволяют совершать сделки и заказы в течение 24 часов в сутки в любом месте, где есть доступ в сеть.

Однако, к сожалению, во многих наших компаниях пока еще развитие бизнеса и развитие информационных технологий происходят независимо. Зачастую информационные технологии рассматриваются как неизбежная, но чисто вспомогательная функция, выступающая как центр затрат. Чтобы изменить такую ситуацию, нужно обосновать, каким образом успех предприятия на рынке или деятельность государственной организации в обществе могут быть обеспечены за счет возможностей, предоставляемых современными технологиями. А для этого требуется строить эффективные, способные к постоянному развитию АС.

**1.2.3 Влияние информационных технологий и автоматизированных систем на деятельность предприятий**

Новые информационные технологии и реализованные на их основе автоматизированные системы являются мощным инструментом для организационных изменений, которые "вынуждают" предприятия перепроектировать свою структуру, область деятельности, коммуникации, ресурсы, т. е. провести полный реинжиниринг бизнес-процессов для достижения новых стратегических целей. В таблице 1.7 показаны некоторые технические и технологические новации, применение которых неизбежно приводит к необходимости изменений предприятия.

Таблица 1.7

Факторы, приводящие к необходимости изменений предприятия

|  |  |
| --- | --- |
| Информационные технологии | Организационные изменения |
| Глобальные сети | Международное разделение производства: действия компании не ограничиваются локализацией; глобальная сфера деятельности расширена; снижаются затраты на производство за счет дешевой рабочей силы, улучшается координация филиалов. |
| Сети предприятия | Совместная работа: организация процессов координируется поверх границ подразделений, распределенные производственные мощности становятся доминирующим фактором.  Управление процессами подчиняется единому плану. |
| Распределенное управление | Изменяются полномочия и ответственность: личности и группы имеют информацию и знания, чтобы действовать самостоятельно.  Бизнес-процессы перестают быть "черными ящиками". Затраты на текущее управление снижаются. Централизация и децентрализация хорошо сбалансированы. |
| Распределенное производство | Организация становится частично виртуальной: производство не привязано географически к одному месту. Информация и знания доставляются туда, где они необходимы, в нужном количестве и в нужное время.  Снижаются организационные и капитальные затраты, так как уменьшается потребность в недвижимом имуществе для размещения средств производства. |
| Графические интерфейсы пользователя | Все в организации, начиная с высших руководителей и кончая исполнителями, имеют доступ к необходимой информации и знаниям; управление процессами автоматизируется, контроль становится простой процедурой.  Организационные процессы и документооборот упрощаются, так как управленческие воздействия движутся от бумажного воплощения к цифровому. |

Внедрение АС может приводить к организационным изменениям различной степени: от минимальных до далеко идущих. Всё зависит от стратегии развития компании, предметной области её деятельности, от развитости сети бизнес-процессов, от степени интегрированности информационных ресурсов и, конечно, от степени решимости и настойчивости высшего руководства предприятия довести начатые преобразования до логического завершения.

Возможные изменения в компании под воздействием ИТ/АС представлены в таблице 1.8.

Таблица 1.8

Возможности изменений в компании под воздействием ИТ/АС

|  |  |
| --- | --- |
| Сфера изменений | Организационное воздействие (результат) ИТ/АС |
| Деловая | ИТ преобразуют неструктурированные процессы в частично структурированные и структурированные, пригодные к автоматизации подготовки принятия решения |
| Автоматизация | ИТ заменяют или уменьшают роль исполнителя в выполнении стандартных (рутинных) функций и операций |
| Анализ | ИТ обеспечивают аналитика необходимой информацией и мощными аналитическими средствами |
| Информационная | ИТ доставляют всю необходимую информацию в управленческие и производственные процессы конечному потребителю |
| Параллельность и доступ | ИТ позволяют выстраивать процессы в нужной последовательности с возможностью параллельного выполнения однотипных операция и одновременного доступа многих устройств и исполнителей |
| Управления данными и знаниями | ИТ организуют сбор, обработку, систематизацию данных, формирование и распространение знаний, экспертных и аудиторских действий для улучшения процессов |
| Отслеживание и контроль | ИТ обеспечивают детальное отслеживание выполнения процессов и контроль исполнения управленческих воздействий |
| Интеграция | ИТ напрямую объединяют части деятельности во взаимосвязанные процессы, которые раньше были связаны через посредников и промежуточные управленческие звенья |
| Географическая и телекоммуникационная | ИТ быстро передают информацию для выполнения процессов, независимо от места их выполнения |

На рисунке 1.3 показаны четыре основных класса структурных изменений в компании, которые поддерживаются информационными технологиями. Каждый из них имеет свои последствия и риски.

Наиболее распространенная форма организационных изменений с помощью ИТ/АС — автоматизация бизнес-процессов (Business Process Automation — BPA). Первые приложения, разработанные с помощью ИТ, затронули финансовые операции и документооборот, так как это наиболее формализованная часть деловых процессов компании. Расчеты и исполнение платежей, контроль транзакций и перемещения документов, прямой доступ клиентов к своим депозитам — вот стандартные примеры ранней автоматизации. Риск внедрения этих технологий был минимальным, выигрыш очень большим.

Рис. 1.3 Уровни структурных изменений в компании

Более глубокая форма организационного изменения, уже затрагивающая структуру производства, — рационализация рабочих процедур или улучшение процессов (Business Process Improvement — BPI). Для наведения порядка в сложных и распределенных процедурах и процессах необходимо изменять порядок их выполнения. Суть изменений — рациональное выстраивание технологических процедур, экономия процессного пространства и времени. Рационализация также не привносит большого дополнительного риска, так как она может начинаться с локальных процедур и процессов и только после получения экономического эффекта распространяться на все предприятие.

Более серьезный тип изменений — реинжиниринг (перепроектирование) бизнес-процессов (Business Process Reenginee-ring — BPR), в течение которого процессы заново идентифицируются, анализируются, переосмысляются и изменяются с целью оптимизировать производство и радикально уменьшить затраты. Использование ИТ помогает реализовать все эти процессы с наибольшей эффективностью. Реинжиниринг бизнеса реорганизует производственные и управленческие процессы, комбинирует и улучшает их, устраняет дублирование однотипных операций. Всё это требует нового, свежего видения проблем предприятия и его места в сложившихся рыночных отношениях и в современном мире.

Процедуры BPA, BPI и BPR, как правило, ограничены отдельными функциями, процессами, подразделениями компании или отдельными частями бизнеса. Риск таких изменений становится ощутимым, если предприятие должным образом не подготовлено к необходимым изменениям как в процедурной или процессной областях, так и в системе управления деятельностью предприятия.

Новые ИТ, в конечном счете, призваны изменить природу всей организации, трансформируя ее цели и стратегические устремления (Paradigm Shift — PS). Например, освоение принципиально новой ниши рынка, открытие филиалов компании в других странах, приобретение другой компании или слияние с компанией партнера и т. д. Такие организационные изменения обладают наибольшим риском, но они несут и наивысшую отдачу. Руководство компании должно осознанно подходить к изменениям такого типа, понимая всю меру ответственности за принимаемые глобальные решения.

В таблице 1.9 показано качественное распределение компаний по признаку отсталости или успешности внедрения и применения новых ИТ-технологий.

Таблица 1.9

Распределение компаний по признаку отсталости или успешности

внедрения и применения новых ИТ-технологий

|  |  |  |  |
| --- | --- | --- | --- |
| Рast (Оставшиеся в прошлом) | Crisis (В кризисном состоянии) | Forward (Конкуренто-способные) | Leading (Лидирующие) |
| Руководство компании не верит в возможности ИТ для развития бизнеса.  Затраты на ИТ были необоснованными, недостаточными или неэффективными.  Базовый принцип приобретения вычислительных средств — дешевизна и быстрота установки без планирования и проработки решений.  АС не поддерживается, не модифицируется, не развивается.  Эксплуатация отдана случайным людям.  Новые разработки, как правило, отсутствуют.  Обучение персонала никогда не производилось. | Руководство компании не участвует в планировании ИТ-деятельности.  ИТ-отдел не имеет самостоятельного значения.  Затраты на ИТ не повышаются с изменениями требований рынка и успехами конкурентов.  Вычислительное и сетевое оборудование приобретается без разработанного и утвержденного ИТ-проекта от случая к случаю.  Основная часть ИТ бюджета расходуется на эксплуатацию и поддержку.  АС наращивается хаотически, сложность растет в ущерб пониманию и гибкости.  Новые разработки, как правило, неэффективны и не окупаются.  Средства на обучение не выделяются. | Деятельность в области приобретения, разработки и внедрения ИТ планируется в соответствие с основными потребностями деятельности компании.  Расходы находятся под контролем.  Вычислительная среда ИТ является распределенной, надежной и понятной в применении.  Используются современные международные стандарты и платформы для разработки приложений.  Основные и вспомогательные бизнес-процессы поддерживаются ИТ-приложениями.  Компания готова проводить реинжиниринг основных бизнес-процессов.  Новые разработки, как правило, эффективны, затраты с течением времени окупаются полностью.  Проводится регулярное повышение квалификации персонала. | Высшие руководители компании формируют политику в области применения и развития ИТ.  Компания умело применяет новейшие технологии для ведения и развития своего бизнеса.  ИТ-расходы нацелены на получение конкурентоспособного преимущества.  Развитая, надежная и удобная инфраструктура, управляемая и легко настраиваемая конфигурация, удобные интерфейсы.  Возможно использование готовых решений в условиях гибкого реинжиниринга, заказные решения интегрируются с существующими и открыты для дальнейшей разработки приложений.  Компания имеет собственный учебный центр. |

В развитии вычислительных и информационных систем предприятий в настоящее время основная тенденция заключается во все большей интеграции ИТ/АС для максимальной отдачи, повышения эффективности использования и роста "возврата инвестиций".

***1.3 Интегрированная автоматизированная система управления как пример большой сложной АС и ее зарубежные аналоги***

**1.3.1 Определение интегрированной автоматизированной системы управления предприятием**

Практика разработки АСУ больших предприятий показала нецелесообразность создания на них (как это делалось ранее и делается иногда сейчас) группы самостоятельных несвязанных или слабо связанных АСУ, реализующих частные функции или задачи автоматизации управления (так называемая «островная», «кусочная» или «лоскутная» автоматизация). Доказана целесообразность создания единых интегрированных АСУ (ИАСУ).

***ИАСУ предприятия*** – многоуровневая автоматизированная система управления, предназначенная для сквозной комплексной автоматизации функций управления инженерно-технической, административно-хозяйственной, производственно-технологической и социальной деятельностью промышленных предприятий и обеспечивающая наиболее эффективное решение задач по планированию выпуска, разработке, освоению, производству и реализации продукции в соответствии с требованиями экономической целесообразности.

Поясним некоторые термины данного определения.

***Сквозной характер автоматизации*** предполагает равномерно распределённый уровень автоматизации по всем элементам производства, а также взаимную стыковку всех элементов системы посредством согласованных интерфейсов, т.е. без дополнительных интерфейсных преобразователей, как аппаратных, так и программных.

***Комплексный характер автоматизации*** предполагает охват всех элементов производства, существенно влияющих на характер функционирования предприятия и принципиально поддающихся автоматизации.

Результат ***автоматизации*** (определение дано в п. 1.1.2) – либо безлюдная технология, либо такая деятельность, в которой человек освобожден от рутинного нетворческого труда.

*Наиболее существенное требование к ИАСУ* – интеграция отдельных компонентов (модулей, частей и подсистем) в единую систему и обеспечение её гибкости, т.е. возможности быстрой переналадки на выпуск новых изделий.

Проблема создания ИАСУ связана с проблемой создания такого производственного комплекса, в котором объектом автоматизации является производственный процесс в широком смысле: от научно-исследовательских работ и конструирования изделий до их изготовления, испытания, складирования и реализации. В отечественной литературе такие системы имеют и другие названия: ИКП – Интегрированное компьютеризированное производство, СКАП - Система комплексной автоматизации производства, КАП - Комплексно автоматизированное производство.

Компоненты ИАСУ – это достаточно самостоятельные большие АС, которые сами состоят из функциональных и обеспечивающих подсистем. Существует достаточно много вариантов выделения типовых компонентов ИАСУ; два из них показаны в табл. 1.10 и табл. 1.11. На различных предприятиях в зависимости от их размера и сложности могут создаваться ИАСУ с разными комбинациями компонентов.

До начала 80-х гг. ХХ века СССР был мировым лидером в области теории и практики создания ИАСУ. Однако с этого момента и особенно после распада СССР лидерство было утеряно. На первые позиции вышли западные экономически развитые страны и, в первую очередь, США. Соответственно, в литературе и на практике в нашей стране стали широко использоваться принятые за рубежом концепции создания АС.

Таблица 1.10

Вариант №1 выделения компонентов ИАСУ

|  |  |
| --- | --- |
| *Аббревиатура* | *Расшифровка названия компонента* |
| АСНИ | Автоматизированная система научных исследований |
| САПР-К | Система автоматизации проектных работ конструктора |
| САПР-Т | Система автоматизации проектных работ технолога |
| АСУП | АСУ предприятием |
| АСУТП | АСУ технологическим процессом |
| АСУГПС | АСУ гибкой производственной системой |
| АСУАСС | АСУ автоматизированной складской системой |
| АСУАТС | АСУ автоматизированной транспортной системой |
| КСУКП | Комплексная система управления качеством продукции |

Таблица 1.11

Вариант №2 выделения компонентов ИАСУ и их функции

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Подсистемы маркетинга** | **Производствен-ные подсистемы** | **Финансовые и учетные подсистемы** | **Подсистема кадров (человеческих ресурсов)** | **Прочие подсистемы (например, АС руководства)** | | Исследование рынка и прогнозиро-вание продаж | Планирование объемов работ и разработка кален-дарных планов | Управление портфелем   заказов | Анализ и прог-нозирование потребности в трудовых ресурсах | Контроль за деятельностью фирмы | | Управление продажами | Оперативный контроль и управление  производством | Управление   кредитной политикой | Ведение архивов записей о персонале | Выявление оперативных проблем | | Рекоменда-ции по производству новой продукции | Анализ работы оборудования | Разработка финансового плана | Анализ и планирование подготовки кадров | Анализ управленческих и стратегических ситуаций | | Анализ и установление цены | Участие в формировании заказов поставщикам | Финансовый анализ и прогнозирование |  | Обеспечение процесса выработки стратеги-ческих  решений | | Учет заказов | Управление запасами | Контроль бюджета, бухгалтерский учет и расчет зарплат |  |  | |

**1.3.2 Принятая за рубежом концепция автоматизации ICAM (CIM)**

Зарубежным аналогом ИАСУ являются системы, создаваемые согласно концепции ICAM (Integrated Computer- Aided Manufacturing), имеющей и другое название CIM (Computer Integrated Manufacturing). Концепция ICAM появилась в США в 80-х годах ХХ века в ходе реализации предложенной военно-воздушными силами (ВВС) США программы компьютеризации промышленности.

Вариант выделения укрупненных компонентов ICAM (CIM), объединяемых на основе компьютерной технологии CA (Computer Aided), и соответствующие им компоненты ИАСУ, представлены в табл. 1.12.

Таблица 1.12

Основные компоненты систем согласно концепции ICAM

и их соответствие ИАСУ

|  |  |  |
| --- | --- | --- |
| *Компоненты ICAM* (*CIM)* | *Назначение* | *Соответствующие компоненты ИАСУ* |
| CAE - Computer Aided Engineering | автоматизация инженерной разработки изделий | АСНИ,  подсистема УТПП АСУП |
| CAD - Computer Aided Design | автоматизация конструирования изделий | САПР-К |
| CAPP- Computer Aided Process Planning | автоматизация технологического проектирования | САПР-Т - |
| MIS – Management Information System | автоматизация управления предприятием | АСУП |
| CAM - Computer Aided Manufacturing | автоматизация управления производственными процессами | АСУГПС, АСУТП |
| CAST - Computer Aided Storage and Transportation | автоматизация управления складированием и транспортом | АСУАСС, АСУАТС |
| CAQA - Computer Aided Quality Assurance | автоматизация контроля качества | КСУКП |
| PPC – Production Planning and Control | автоматизация планирования и управления | подсистема ТЭП АСУП |
| CAA - Computer Aided Accounting | автоматизация бухгалтерского учета | подсистема БУ АСУП |
| CAB - Computer Aided Budgeting | автоматизация финансовых расчётов | подсистема УФ АСУП |
| Computer Aided Marketing | автоматизация маркетинга | подсистема МТС АСУП и подсистема СРП АСУП |

В рамках ICAM, с одной стороны, создано целое семейство проектных методологий IDEF, являющихся, по сути, стандартами процессов проектирования и позволяющих отображать и анализировать модели деятельности широкого спектра сложных систем в различных разрезах. С другой стороны, в рамках ICAM создано значительное количество подходов и технологий реализации проектов реальных АС. Многие из таких подходов и технологий нашли свое эффективное применение при создании АС в различных отраслях промышленности по всему миру. Некоторые из них приобрели статус международных стандартов.

За рубежом в спецификации АС, разрабатываемой для массовой продажи, как правило, указывается - какие стандарты и технологии управления она поддерживает. В таблице 1.13 перечислены и кратко охарактеризованы некоторые, наиболее важные технологии и стандарты создания АС для крупных производственных предприятий согласно ICAM.

Приведенные в таблице и многие другие зарубежные технологии и стандарты обеспечивают создание интегрированных АС с разной степенью интеграции (в смысле количества интегрируемых в рамках системы функций) и с разными теоретическими подходами к реализации автоматизируемых функций. По сути дела, все эти системы укладываются в понятие ИАСУ и различаются масштабом и теоретической базой реализации (или иначе – используемыми экономико-математическими методами). В качестве примера в таблице 1.14 приведено сопоставление организации решения задач производственного планирования в MRPII / ERP системах и в традиционных АСУ.

Далее более детально рассмотрим наиболее известные и чаще всего используемые настоящее время в мировой практике для реализации полнофункциональных интегрированных АСУ стандартные технологии: MRP, MRPII, ERP, CSRP, ERPII.

                                                                                                                                                       Таблица 1.13

Технологии и стандарты создания АС в рамках ICAM

|  |  |
| --- | --- |
| *Наименование и аббревиатура* | *Назначение* |
| SIC (Statistical Inventory Control) | Планирования запасов, при котором используются статистические методы для моделирования спроса и времени пополнения товарных запасов (для производственных компаний – с учетом времени изготовления). Основано на расчете нормативных характеристик запасов по каждому виду товарного ассортимента исходя из наблюдений за их фактической оборачиваемостью на протяжении достаточно представительного периода |
| DRP (Distribution Requirements Planning), | Планирование запасов, которое использует модели пополнения запасов на основе обработки совокупности заявок дистрибьюторов (обобщаются различные договоренности с клиентами и наблюдения за их поведением) |
| MRP (Material Requirements Planning) | Планирование поставок материалов, исходя из данных о комплектации производимой продукции и плана продаж. Основная идея MRP систем состоит в том, что любая учетная единица материалов или комплектующих, необходимых для производства изделия, должна быть в наличии в нужное время и в нужном количестве |
| MES (Manufacturing Execution Systems; с 2004 года термин расшифровывается как Manufacturing Enterprise Solutions) | Системы оперативного контроля хода производственного процесса, которые имеют собственный модуль детальной оптимизации ODS, используемый диспетчером для расчёта и коррекции текущих внутрицеховых производственных расписаний. Предназначены для решения задач синхронизации, координации, анализа и оптимизации выпуска продукции в рамках какого-либо производства |
| CRP (Capacity Requirements Planning) | Планирование производственных мощностей, исходя из данных о технологии производимой продукции и прогноза спроса |
| MRPII (Manufacture Resource Planning) | Планирование материальных, мощностных и финансовых ресурсов, необходимых для производства. Стандартизовано ISO как концепция управления производством и запасами. Обеспечивает функциональную поддержку выполнения следующего цикла: “планирование заказов —> планирование потребности в сырье и материалах —> планирование производственных ресурсов —> контроль над исполнением производственной программы —> обратная связь” |
| APS (Advanced Planning & Scheduling) | Усовершенствованное планирование — концепция оптимизированного (или синхронного) производственного планирования, главной особенностью которой является возможность быстрого составления планов с учётом имеющихся ресурсов и производственных ограничений (переналадки оборудования, доступность оснастки, связи между машинами и др.) и быстрого перепланирования по заранее составленным сценариям оптимизации. Концепция APS совместила в себе основные элементы концепций MRP, CRP, MRPII. Цель систем класса APS - объединение цепочки поставщиков и потребителей для достижения максимальной прибыли и лучшего удовлетворения требований рынка за счет усовершенствованного планирования на базе информационных технологий |
| ERP (Enterprise Resource Planning) | Финансово-ориентированное планирование ресурсов предприятия, необходимых для получения, изготовления, отгрузки и учета заказов потребителей на основе интеграции всех отделов и подразделений компании |
| SCM (Supply Chain Management) | Управление цепочками поставок. Реализация бизнес-процессов на базе внешних предприятий и торговых площадок. Основано на референтной модели SCOR, стандартизованной Supply Chain Council |
| CRM (Customer Relationship Management) | Управление взаимоотношениями с заказчиками. Комплекс методов и средств, нацеленный на завоевание, удовлетворение требований и сохранение платежеспособных клиентов |
| ERPII (Enterprise Resource & Relationship Processing) | Управление ресурсами и взаимоотношениями предприятия. Объединяет в себе 3 вышеперечисленные технологии |
| CSRP (Customer Synchronized Relationship Planning ) | Планирование ресурсов, синхронизированное с покупателем |
| Workflow | Технология управления потоком работ при помощи программного обеспечения, способного интерпретировать описание процесса, взаимодействовать с его участниками и при необходимости вызывать соответствующие программные приложения |
| OLAP (Online Analytical Processing) | Оперативный анализ данных. Технология поддержки принятия управленческих решений на основе концепции многомерных кубов информации |
| BPM (Business Performance Management) | Системы управления эффективностью бизнеса, которые позволяют связывать операционные результаты деятельности предприятия с эффективностью реализации миссии компании. |
| CALS (Continuous Acquisition and Lifecycle Support) | Непрерывная информационная поддержка поставок и жизненного цикла. Описывает совокупность принципов и технологий информационной поддержки жизненного цикла продукции на всех его стадиях. Объединяет в себе практически все вышеперечисленные подходы и технологии |

Таблица 1.14

Иерархия задач производственного планирования в MRPII / ERP - системах и традиционных АСУ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Основные задачи планирования производства** | **Характеристика основных задач** | **Степень детализации планов и горизонт планирования** | **Вспомогательные задачи балансировки планов и производственных мощностей** | **Соответствующие задачи планирования в традиционных АСУ** |
| Production Plan – PP (план по производству) | План по товарным группам изделий, в натуральном, стоимостном выра-жении или в нормо-часах, формиру-ется на основе плана сбыта в дирекции предприятия. План используется для увязки стратегиче-ских целей предприятия, его произ-водственных возможностей и спроса | План охватывает предприятие в целом. Формируется на один-два года с разбивкой по кварталам, месяцам | Resource Plan - смета затрат для выполнения плана по производству - охватывает годовые потребности в ключевых ресурсах (включая материальные и трудовые ресурсы, ресурсы и производственные мощности) | Производственная программа - план производства и реализации продукции (согласование портфеля заказов и производственных мощностей) |
| Master Production Schedule - MPS (Комплексный график производства) | Формируются исходные данные для MRP с учетом портфеля заказов и прогнозов выпуска групп изделий. План служит для контроля за сроками выпуска заказов, управления заказами, находящимися в производстве, поддержания заданных уровней хранения на складе готовой продукции | План охватывает предприятие в целом. Формируется на год или полгода (но не менее длительности производственного цикла) с разбивкой на кварталы, месяцы, недели | Rough-cut Capacity Planning - предварительное планирование производственных мощностей (проверка сбалансированности плана по ресурсам и по группам технологического оборудования, загрузка учитывается в нормо-часах) | Уточнение производственной прог-рамммы с учетом фактически приня-тых заказов, и ее равномерное распре-деление по фонду времени. Комплекс-ный график производства содержит информацию о заказах, которую по степени детализации можно отнести к производственной программе, объем-ному и оперативному планированию (в терминах АСУ) |
| Material Requirements Planning - MRP (планирование потребностей в материальных ресурсах) | Автоматическое формирование заказов на запуск в производство и выпуск продукции, на поставку материальных ресурсов. Динамическое назначение приоритетов заказам при выполнении очередных операций | Планы для подразделе-ний и служб предприя-тия. Охватывает интер-вал времени до месяца (реже - до года) с дневной (недельной) детализацией | Capacity Requirements Planning - детальное Планирование производственных мощностей | Распределение производственной про-граммы по производственным подраз-делениям и согласование с планами для обеспечивающих подразделений. Разузлование и оперативное планиро-вание: детализация планов по номенк-латуре, формирование планов по запу-ску в производство и выпуску изделий |

**1.3.3 Характеристика MRP-систем как компонента ICAM**

В таких системах реализуется методология планирования материальных потребностей предприятия MRP (Material Requirement Planning). Необходимость планирования вызвана тем, что основные задержки в производстве продукции связаны, в первую очередь, с неритмичными поставками заказанного оборудования и комплектующих изделий. Вследствие этого снижается эффективность производства (из-за недопроизводства продукции), а на складах возникает переизбыток материалов, поступивших ранее намеченного срока, скапливается готовая и внеплановая продукция. Кроме того, из-за нарушения баланса поставок комплектующих возникают различные сложности с их учетом в процессе производства и сопровождения продукции.

Первые реализации таких систем появились в 60-е годы ХХ века (например, при реализации проекта "Боинг-747", для которого требовались сотни тысяч листов документов и деталей). Алгоритм работы такой системы следующий. На входе задаются основной производственный план и список номенклатуры изделий (т.е. из чего изделие состоит). Далее производится операция "разузлование (раскрытие)" — расчет себестоимости и/или потребности в материалах. В результате расчета получается план закупок. Естественное развитие таких систем — добавить учет запасов на складах и учет времени выполнения операции. В результате получается: план закупок и план производства + исправление к планам, если меняется портфель заказов.

Основными целями MRP-систем являются:

·  удовлетворение потребности в материалах, компонентах и продукции для планирования производства и доставки потребителям;

·  поддержка уровней запасов не выше запланированных;

·  планирование производственных операций, расписаний доставки, закупочных операций.

Методология MRP является реализацией двух известных принципов "Вовремя заказать" (Order In Time) и "Вовремя произвести" (Kanban), объединенных в методологию "Вовремя выполнить" (Just In Time — JIT). По сути, эта методология представляет собой алгоритм оптимального управления заказами на готовую продукцию, производством и запасами сырья и материалов, реализуемый с помощью компьютерной системы.

С целью повышения эффективности планирования в конце 70-х гг. в MRP-системах была реализована идея Оливера Уайта и Джорджа Плосла воспроизведения замкнутого цикла (Closed Loop Material Requirement Planning), подразумевающая составление согласованной производственной программы и её контроль на цеховом уровне. К базовым функциям планирования производственных мощностей и планирования потребностей в материалах были добавлены дополнительные функции — в первую очередь, оценка результатов деятельности (Performance Measurement), например, контроль соответствия количества произведенной продукции количеству использованных в процессе сборки комплектующих. Также добавилось составление регулярных отчетов о задержках заказов, об объемах и динамике продаж продукции, о поставщиках и др. Созданные в процессе работы модифицированной MRP-системы отчеты анализировались и учитывались на дальнейших этапах планирования, изменяя (при необходимости) программу производства и план заказов (обеспечивая, тем самым, гибкость планирования по отношению к таким внешним факторам, как уровень спроса, текущее состояние дел у поставщиков, наличие комплектующих и др.).

Итак, главной задачей MRP-систем является обеспечение наличия на складе и в производственных помещениях необходимого количества требуемых материалов (комплектующих) в любой момент времени в рамках срока планирования. Программные системы, реализованные на базе MRP-методологии, позволили оптимально регулировать поставки, контролировать складские запасы и саму технологию производства. Использование MRP-систем позволило уменьшить объем постоянных складских запасов за счёт оптимизации процесса поставок.

Основные достоинства и недостатки MRP-систем приведены в табл. 1.15.

Таблица 1.15

Основные достоинства и недостатки MRP-систем

|  |  |
| --- | --- |
| Достоинства | ·     организационная эффективность планирования производственных запасов,  ·     автоматизация их учета,  ·     уменьшение ошибок в планировании запасов и затрат на складское хранение материальных ресурсов. |
| Недостатки | ·      значительный объем вводимых данных и их предварительной обработки;  ·      возрастание логистических затрат на обработку заказов и транспортировку при стремлении фирмы ещё больше уменьшить запасы материальных ресурсов или перейти на работу с малыми заказами с высокой частотой их выполнения;  ·      нечувствительность к кратковременным изменениям спроса;  ·      наличие отказов из-за большой размерности системы и ее сложности. |

Тем не менее, в 60-х и 70-х годах ХХ века многие крупные производственные компании — автомобильные, судосборочные, авиастроительные — успешно использовали MRP-системы для повышения эффективности производства.

**1.3.4 Характеристика MRPII-систем как компонента ICAM**

При расчете потребности в материалах в MRP-системах не учитываются производственные мощности, величина и неравномерность их загрузки, стоимость рабочей силы и т. д. В конце 70-х гг. ХХ века методология MRP-систем с замкнутым циклом была трансформирована в систему планирования производственных ресурcов в масштабах предприятия (Manufactory Resource Planning), которая получила название MRPII (римскую цифру II добавили вследствие идентичности аббревиатур).  Стандарт MRPII был разработан в США и поддерживается Американским обществом по управлению производством и запасами (American Production and Inventory Control Society — APICS.

К 1980 году сложилась формула: MRPII = MRP + "пропускная способность производства" (Manufacturing Capacity).  В АС, построенной на базе стандарта MRPII, должны быть реализованы 16 групп функций, показанных в таблице 1.16.

Таблица 1.16

Группы функций, реализуемых на базе стандарта MRP II

|  |  |
| --- | --- |
| № | Название группы функций на русском и английском языке |
| 1 | Планирование продаж и производства                 (Sales and Operation Planning) |
| 2 | Управление спросом                                               (Demand Management) |
| 3 | Составление плана производства                          (Master Production Scheduling) |
| 4 | Планирование потребностей в материалах          (Material Requirement Planning) |
| 5 | Спецификация продуктов                                      (Bill of Materials) |
| 6 | Управление складом                                               (Inventory Transaction Subsystem) |
| 7 | Плановые поставки                                                 (Scheduled Receipts Subsystem) |
| 8 | Управление на уровне производственного цеха (Shop Flow Control) |
| 9 | Планирование производственных мощностей    (Capacity Requirement Planning) |
| 10 | Контроль входа/выхода                                          (Input/Output Control) |
| 11 | Материально-техническое снабжение                 (Purchasing) |
| 12 | Планирование распределения ресурсов               (Distribution Resourse Planning) |
| 13 | Планирование и контроль производственных операций (Tooling Planning and Control) |
| 14 | Финансовое планирование                                                (Financial Planning) |
| 15 | Моделирование                                                       (Simulation) |
| 16 | Оценка результатов деятельности                                    (Performance Measurement) |

АС, реализованная на базе MRPII, предназначена для эффективного планирования всех ресурсов предприятия (включая финансовые и кадровые). Основная суть MRPII-концепции состоит в том, что прогнозирование, планирование и контроль производства осуществляется по всему жизненному циклу продукции, начиная от закупки сырья и заканчивая отгрузкой продукции потребителю.

Задачей АС класса MRPII является оптимальное формирование потока материалов (сырья), полуфабрикатов (комплектующих) и готовых изделий. Система имеет целью интеграцию основных процессов, реализуемых предприятием: планирование и контроль выполнения плана, затраты, снабжение, производство, продажа, управление запасами, загрузка основных средств и т. д.

В такой системе интегрировано большое число модулей, результаты работы которых анализируются MRPII -системой в целом, что и обеспечивает ее гибкость по отношению к различным внешним факторам — например, текущему спросу на продукцию, котировке цен и др. (рис. 1.4).

В результате применения MRPII-стандарта реализуются:

·  долгосрочное, оперативное и детальное планирование деятельности предприятия с возможностью корректировки плановых данных на основе оперативной информации;

·  оптимизация производственных и материальных потоков со значительным сокращением непроизводственных затрат и реальным сокращением материальных ресурсов на складах;

·  возврат инвестиций, произведенных в информационные технологии;

·  возможность поэтапного внедрения и развития системы, с учетом инвестиционной политики конкретного предприятия;

·  отражение финансовой деятельности предприятия в целом.

В настоящее время к стандарту MRPII "подключили" идеологию JIT (точно в срок), различные комбинации с элементами "Системы Канбан", добавили систему оптимизации "узких мест". Всё это в сочетании с системным подходом, целостностью управленческих процессов, реинжинирингом процессов, осознанием ценности работников образовало методологию MRPII+, которая легла в основу разработки ядра стандартизированной ERP-системы.

Рис. 1.4  Общая схема планирования ресурсов производственного предприятия

**1.3.5 Характеристика ERP-систем как компонента ICAM**

Системы класса MRPII в интеграции с модулем финансового планирования (Finance Requirements Planning — FRP) получили название систем планирования ресурсов предприятий (Enterprise Resource Planning — ERP).

Существует немало определений ERP-систем. Одно из них наиболее часто встречающихся — следующее: ERP-система — это набор интегрированных приложений, позволяющих создать интегрированную информационную среду (ИИС) для автоматизации планирования, учета, контроля и анализа всех основных бизнес-операций предприятия. Основой ИИС предприятия являются именно ERP-системы.

В основе ERP-систем лежит принцип создания единого хранилища (репозитория) данных, содержащего всю корпоративную бизнес-информацию: плановую, финансовую, производственную, данные по персоналу и др. Наличие единого корпоративного репозитория устраняет необходимость в передаче данных от одной системы к другой (например, от производственной системы к финансовой или к кадровой). Такая система обеспечивает также одновременную доступность к информации любого числа сотрудников предприятия, обладающих соответствующими полномочиями. Цель ERP-систем - не только улучшение управления производственной деятельностью предприятия, но и уменьшение затрат и усилий на поддержку его внутренних информационных потоков.

Как правило, ERP-системы строятся по модульному принципу и в той или иной степени охватывают все ключевые процессы деятельности компании (рис. 1.5).

Рис. 1.5Традиционная схема ERP

Используемый в ERP-системах программный инструментарий позволяет проводить производственное планирование, моделировать поток заказов и оценивать возможность их реализации в службах и подразделениях предприятия, увязывая его со сбытом. Основные функции ERP систем приведены в табл. 1.17.

Таблица 1.17

Основные функции ERP систем

|  |  |
| --- | --- |
| № | Содержание функции |
| 1 | Ведение конструкторских и технологических спецификаций, определяющих состав производимых изделий, а также материальные ресурсы и операции, необходимые для его изготовления |
| 2 | Формирование планов продаж и производства |
| 3 | Планирование потребностей в материалах и комплектующих, сроков и объемов поставок для выполнения плана производства продукции |
| 4 | Управление запасами и закупками: ведение договоров, реализация централизованных закупок, обеспечение учета и оптимизации складских и цеховых запасов |
| 5 | Планирование производственных мощностей от укрупненного планирования до использования отдельных станков и оборудования |
| 6 | Оперативное управление финансами, включая составление финансового плана и осуществление контроля его исполнения, финансовый и управленческий учет |
| 7 | Управления проектами, включая планирование этапов и ресурсов, необходимых для их реализации |

ERP-системы предназначены для управления всей финансовой и хозяйственной деятельностью предприятия. Они используются для оперативного предоставления руководству предприятия информации, необходимой для принятия управленческих решений, а также для создания инфраструктуры электронного обмена данными предприятия с поставщиками и потребителями. ERP-системы позволяет использовать одну интегрированную программу вместо нескольких разрозненных. Единая система может управлять обработкой, логистикой, дистрибуцией, запасами, доставкой, выставлением счетов-фактур и бухгалтерским учётом.

Реализуемая в ERP-системах система разграничения доступа к информации предназначена (в комплексе с другими мерами информационной безопасности предприятия) для противодействия как внешним угрозам (например, промышленному шпионажу), так и внутренним (например, хищениям данных). Внедряемые в связке с системами контроля качества и поддержки отношений с клиентами, ERP-системы нацелены на максимальное удовлетворение потребностей компаний в средствах управления бизнесом.

Однако, несмотря на то, что ERP-системы сфокусированы на автоматизации управленческих процессов, сопровождении бизнес-процессов и снижении операционных затрат, они не способны обеспечить всеобъемлющий, легкий и быстрый доступ к необходимой управленческой информации. Оказалось, что не вся информация, необходимая высшему руководству, менеджерам и специалистам на местах, имеется в наличии в ERP-системе. Это особенно критично для компаний, где применяется не одна, а несколько ERP-систем, доставшихся в наследство в результате слияний и приобретений.

**1.3.6 Характеристика CSRP-систем как компонентов ICAM**

В 90-е годы жёсткая конкурентная борьба была сконцентрирована вокруг производства без брака и поставок "точно вовремя" (JIT), а её усиление заставило производителей искать решения по улучшению и ускорению производственного процесса. Они направляли свои ресурсы на то, как сделать продукт лучше, дешевле и быстро и на то, как улучшить производственную эффективность. Этому служили системы класса MRP/ERP. Однако с начала ХХI-го века в условиях насыщения рынка качественными товарами разных производителей особенно обострилась борьба за клиента (покупателя, заказчика).

Для достижения успехов в этой борьбе была разработана новая методология автоматизации деятельности предприятий "Планирование ресурсов предприятия, синхронизированное с запросами потребителя" (Customer Synchronized Resource Planning — CSRP). Она охватывает взаимодействие предприятия с клиентами: оформление наряд-заказа, техническое задание, поддержку клиентов, планирование ресурсов в зависимости от объема и состава клиентских заказов. Если стандарты MRP/MRPII/ERP ориентированы на управление запасами и мощностями, планирование, производство и продажу продукта, то в стандарт CSRP включен полный цикл жизненного цикла изделия — от его проектирования с учетом требований заказчика до гарантийного и сервисного обслуживания после продажи. По сути дела, ERP-система дополняется функциональными модулями, реализующими концепцию управления взаимоотношениями с клиентом (Customer Relationship Management — CRM), как это показано на рис. 1.6.

Рис. 1.6Интеграция и синхронизация информации о покупателе с основными функциями предприятия

Методология CSRP использует проверенную, интегрированную функциональность ERP-систем и перенаправляет производственное планирование от производства к покупателю. CSRP предоставляет действенные методы и приложения для создания продуктов с повышенной ценностью для покупателя. Всё это в сумме составляет CSRP-технологию. Для ее внедрения необходимо:

·  оптимизировать производственную деятельность (операции), построив эффективную производственную инфраструктуру на основе ERP;

·  интегрировать покупателя и сфокусированные на покупателе подразделения организации с основными планирующими и производственными подразделениями;

·  внедрить открытые технологии, чтобы создать технологическую инфраструктуру, которая может поддерживать интеграцию покупателей, поставщиков и приложений управления производством.

CSRP — это бизнес-методология, которая переносит ту часть деятельности предприятия, которая ориентирована на покупателя, в центр системы управления бизнесом. Концепция CSRP устанавливает методологию ведения бизнеса, основанную на текущей информации о покупателе (требования), и на прогнозах его активности (ожидания). CSRP сдвигает фокус предприятия с планирования от потребностей производства к планированию от заказов покупателей. Информация о покупателях и производимые услуги "вплавляются" в информационную основу организации.

Методология CSRP предоставляет действенные методы и приложения для создания продукции, модифицируемой под конкретного покупателя. Основными механизмами этого технологического решения является наличие модуля, называемого конфигуратором продукции (Product Configurator), и расширенное регулируемое управление производственными графиками в условиях ограниченных мощностей и временных интервалов (Advanced Planning and Scheduling — APS).

Для реализации моментальной модификации производственных планов в методологии CSRP конфигуратор продукции напрямую связан с модулем расширенного планирования и диспетчерского управления (APS-модуль), который использует принципиально новую "математику" для расчета и оптимизации производственных графиков и оптимальной загрузки оборудования (методы математической оптимизации, использующие специальные алгоритмы и работоспособные при большой размерности обрабатываемых данных). В отличие от методологии, основанной на стандарте MRPII, в которой расчет согласованных производственных графиков производится в режиме Off Line, методология CSRP на базе конфигуратора и модуля APS позволяет рассчитать несколько вариантов производственного графика в режиме On Line в момент принятия заказа с оценкой возможных затрат на конкретные ресурсы и переналадку оборудования.

Таким образом, планирование производства и всей деятельности переопределяется и становится планированием заказов покупателей для организации динамичного производства.

Выгоды успешного применения CSRP — это повышение качества товаров, снижение времени поставки, повышение ценности продуктов для покупателя и так далее. В результате достигается снижение производственных издержек, создание инфраструктуры, приспособленной для создания продуктов, удовлетворяющих потребности покупателя, улучшение обратной связи с покупателями и обеспечение лучших услуг для покупателей.

Укрупненное представление о соотношении рассмотренных компонентов ICAM, а также сравнение соответствующих стандартов управлениядано на рис. 1.7.

Рис. 1.7 Соотношение компонентов ICAM

**1.3.7 Характеристика  ERPII-систем как компонентов ICAM**

Давно теоретически и практически доказано, что эффективность бизнеса требует сокращения на предприятиях непроизводственных затрат. Реальным резервом такого сокращения является оптимальная организация движения сырья и комплектующих изделий для переработки и сборки. Уже в MRPII-системах успешно решались подобные задачи, но только для отдельных компактных предприятий с чётким технологическим циклом. При этом традиционные логистические и управленческие операции по заказу, доставке, складированию, отпуску в производство (управление складскими запасами), дополненные требованиями технологий "В срок заказать" (Order In Time) и "В срок произвести" (Kanban) легли в основу методологии "Точно в срок" (Just In Time - JIT).

В частности, производственнно-учётные карты заказов "канбан" абсолютно точно оговаривают и предписывают количество заказанных комплектующих изделий на сборочный стапель (точку производственного цикла, в которой необходимы эти изделия) и точку времени, к которой изделия должны быть поставлены. Ключевые моменты JIT технологии представлены в табл. 1.18.

Таблица 1.18

Ключевые моменты JIT технологии

|  |  |
| --- | --- |
| № | Описание |
| 1 | Чёткое планирование необходимого объёма материально-производственных запасов |
| 2 | Достижение оптимального уровня запасов на всех участках производства |
| 3 | Уменьшение или полное исключение простоев |
| 4 | Уменьшение размера партий (закупаемой или производимой) продукции и повышенная маневренность партий |
| 5 | Уменьшение времени передачи, обработки, доставки продукции |
| 6 | Сведение до минимума складских площадей и операций |

При внедрении этой технологии необходимо предусмотреть неритмичность поставок по вине поставщиков, возможность поломки оборудования, изменения состава персонала и т. д.

Однако динамика современного бизнеса потребовала применения JIT технологии к распределенным предприятиям широкого спектра направлений деятельности, т.е. наполнения ее новым содержанием.

Процесс автоматизированного управления сложными логистическими процессами на базе математических моделей, описывающих алгоритмы взаимодействия внешних и внутренних поставщиков, схемы и траектории движения материальных ценностей, получил название «управление цепочками поставок» (Supply Chain Management - SCM). Системы, с помощью которых осуществляется такое управление, стали называться SCM-системами. Реализуемая SCM-системой стратегия является двунаправленной — она охватывает как поставку сырья и комплектующих изделий на предприятие, так и доставку "точно в срок" готового продукта на рынок.

Выделяют семь основных принципов концепции SCM (см. табл. 1.19).

Таблица 1.19

Основные принципы концепции SCM

|  |  |
| --- | --- |
| № | Суть принципа |
| 1 | Внимательно следить за рыночным спросом, и производить планирование, опираясь на них |
| 2 | Изучать пространственно-временное распределение продаж и сегментировать потребителей на основе потребности в товарах и сервисах |
| 3 | В равной степени ориентировать логистическую сеть на поставщика и клиента |
| 4 | Стратегически планировать поставки |
| 5 | Разрабатывать стратегию цепочек движения материальных ресурсов |
| 6 | Активно использовать методы привлечения новых каналов распределения |
| 7 | Использовать методы линейного программирования, математическое моделирование и информационные технологии для увеличения точности прогноза и разработки сетевых графиков поставок и оптимальных маршрутов движения |

SCM-системы, равно как и системы CRM и CSRP, "продолжают" стандартную корпоративную ERP-систему во внешнюю среду, образуя в совокупности расширенную систему управления предприятием ERPII (рис. 1.8).

Рис. 1.8Расширенная система управления предприятием на базе ERPII

Такая интегрированная система позволяет реализовать базовое положение стратегии SCM: "доставить нужный товар — в нужное место — точно в срок — с низкими издержками — с нужным сервисом для клиента".

Эффективное управление цепочкой поставок позволяет увеличить доходы за счет поддержания запасов товаров на необходимом для обеспечения спроса уровне, в результате увеличиваются продажи, снижается необходимость уценки товаров для распродажи запасов. Это приводит также к снижению расходов на транспортировку, хранение товаров, дополнительных трудовых затрат за счет оптимального планирования операционных объектов и запасов.

Кроме того, инструменты прогнозирования спроса и планирования цепочек внутренних и внешних поставок обосновывают рекламную активность, сезонное увеличение/уменьшение спроса, рассчитывают оптимальные сроки поставок и т.д. И, наконец, они позволяют лучше использовать активы производственных предприятий, а также розничных и сетевых сбытовых компаний за счет оптимального планирования и размещения товаров в имеющемся пространстве складов и магазинов.

**1.3.8 Экономико-математические методы и модели в компонентах ICAM**

Очевидно, что для реализации функций компонентов ICAM используются различные экономико-математические методы и модели. Спектр их весьма широк и зависит от предметной области, характерной для конкретной АС. Построением конкретных моделей занимаются соответствующие разделы высшей и прикладной математики, в частности: теория массового обслуживания, исследование операций, теория расписаний, теория вероятностей и математическая статистика и т.д.

Приведем несколько примеров экономико-математических моделей и методов реализации функциональных задач в ERP-системах:

•  модели линейного программирования - задачи стратегического управления, планирование производства и продаж;

•  сетевые модели - оперативное планирование;

•  методы регрессионного анализа, обработки временных рядов, экспертных оценок - оценка спроса и т.д.

Еще более расширился спектр используемых экономико-математических методов и моделей после соединения ERP-систем с сервисами OLAP. OLAP-сервис представляет собой инструмент для анализа больших объемов данных в режиме реального времени. Взаимодействуя с OLAP- системой, пользователь может осуществлять гибкий просмотр информации, получать произвольные срезы данных и выполнять аналитические операции детализации, свертки, сквозного распределения, сравнения во времени одновременно по многим параметрам. Вся работа с OLAP-системой происходит в терминах предметной области и позволяет строить статистически обоснованные модели деловой ситуации на основе оперативного анализа данных, содержащихся в многомерном хранилище.

Многомерный OLAP-куб и система соответствующих математических алгоритмов статистической обработки позволяет анализировать данные любой сложности на любых временных интервалах. Главной особенностью является то, что эти средства ориентированы на использование не специалистом в области информационных технологий, не экспертом-статистиком, а профессионалом в прикладной области управления - менеджером отдела, департамента, управления, и, наконец, директором. Средства предназначены для общения аналитика с проблемой, а не с компьютером

При этом доступны некоторые стандартные методы анализа, логически следующие из природы OLAP-технологии и представленные в таблице 1.20.

Таблица 1.20

Стандартные методы анализа, следующие из природы OLAP-технологии

|  |  |  |
| --- | --- | --- |
| № | Название метода | Примеры использования метода |
| 1 | *Факторный (структурный) анализ* | Анализ структуры продаж для выявления важнейших составляющих в интересующем разрезе |
| 2 | *Анализ динамики* (регрессионный анализ - выявление трендов). | Выявление тенденций, сезонных колебаний и т.п. Наглядно динамику отображает график типа "Линия" |
| 3 | *Анализ зависимостей* (корреляционный анализ) | Сравнение объемов продаж разных товаров во времени для выявления необходимого ассортимента - "корзины". Для этого также удобно использовать график типа "Линия" |
| 4 | *Сопоставление* (сравнительный анализ) | Сравнение результатов продаж во времени, или за заданный период, или для заданной группы товаров. В зависимости от количества анализируемых факторов (от 1 до 3-х) используется диаграмма типа "Пирог" или "Столбцы" |
| 5 | *Дисперсионный анализ* | Исследование распределения вероятностей и доверительных интервалов рассматриваемых показателей. Применяется для прогнозирования и оценки рисков |

OLAP-системы являются частью более общего понятия "интеллектуальные ресурсы предприятия" или "средства интеллектуального бизнес-анализа" (Business Intelligence - BI), которое включает в себя помимо традиционного OLAP-сервиса средства организации совместного использования данных и информации, возникающих в процессе работы пользователей хранилища. При этом также используются соответствующие экономико-математические методы и модели.

В заключение отметим, что соединение ERP-систем с сервисами OLAP, системой сбалансированных показателей (Balanced Score Card) и системой функционально-стоимостного управления привело к появлению и развитию особого класса систем еще более высокого уровня интеграции - BPM (Business Performance Management) — управление эффективностью бизнеса. BPM-системы позволяют связывать операционные результаты деятельности предприятия с эффективностью реализации миссии компании, т.е. с глобальной целью ее деятельности.

**1.3.9 Характеристика рынка автоматизированных систем**

Количество отечественных и зарубежных фирм-производителей АС достаточно велико. Соответственно, велико количество представленных ими на рынке готовых АС. Эти системы, как продукты производственно-технического назначения, различаются по реализованной функциональности, масштабу решаемых задач и степени интегрированности компонентов. Примеры систем приведены в табл. 1.21.

Таблица 1.21

Классификация рынка АС

|  |  |  |  |
| --- | --- | --- | --- |
| **Локальные системы** | **Малые интегрированные системы** | **Средние интегрированные системы** | **Крупные интегрированные системы** |
| ·      1C  ·      БЭСТ  ·      Инотек  ·      ИНФИН  ·      Инфософт  ·      Супер-Менеджер  ·      Турбо-Бухгалтер  ·      Инфо-Бухгалтер | ·     Concorde XAL Exact  ·     NS-2000 Platinum PRO/MIS  ·     Scala SunSystems  ·     БЭСТ-ПРО  ·     1C-Предприятие  ·     БОСС-Корпорация  ·     Галактика  ·     Парус  ·     Ресурс  ·     Эталон | ·  MS Dynamics (ранее Axapta)  ·  JD Edwards  ·  MFG-Pro  ·  SyteLine | ·    My SAP  ·    Baan  ·    BPCS  ·    Oracle Applications |

Это интегрированные систем разного масштаба, которые наиболее популярны в настоящее время. Их можно рассматривать как готовые программные продукты, являющиеся основой для реализации на предприятиях АСУ различных классов.

Примеры названий фирм-производителей (поставщиков) наиболее известных современных ERP-систем и адреса их сайтов представлены в таблице 1.22. Примером полезного источника информации по современным ERP-системам является  Интернет ресурс   <http://www.erp-online.ru/erp/> .

Таблица 1.22

Фирмы-производители наиболее известных современных ERP-систем

и адреса их сайтов

|  |  |  |
| --- | --- | --- |
| № | Название ERP-системы и фирмы-производителя | Адрес  сайта |
| 1 | My SAP (SAP AG) | http://www.sap.com |
| 2 | Oracle Applications (Oracle) | <http://www.oracle.ru> |
| 3 | Baan IV (Baan) | <http://www.baansupport.com> |
| 4 | iRenaissance (ROSS Systems) | <http://www.rossinc.com> |
| 5 | SyteLine (SYMIX) | <http://www.frontstep.ru> |
| 6 | MS Dynamics (ранее Axapta, Damgaard Data Int.) | <http://www.microsoft.com> |
| 7 | MFG/PRO\* (QAD) | <http://www.qad.com> |
| 8 | Галактика (Корпорация "Галактика") | <http://www.galaktika.ru> |
| 9 | БОСС-Корпорация (Компания "АйТи") | <http://www.it.ru> |
| 10 | 1С: Предприятие (Компания 1С) | <http://www.1c.ru> |

Важно понимать, что классические ERP-системы, в отличие от так называемого "коробочного" программного обеспечения, относятся к категории "тяжелых" программных продуктов, требующих достаточно длительной настройки, для того чтобы начать ими пользоваться. Выбор АС этого уровня, приобретение и внедрение, как правило, требуют тщательного планирования в рамках длительного проекта с участием партнерской компании — поставщика или консультанта. Некоторые характеристики представленных на рынке АС с точки зрения их внедрения приведены в табл. 1.23.

Таблица 1.23

Соотношение стоимостных оценок внедрения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Признак для сравнения* | **Локальные системы** | **Малые интегрированные системы** | **Средние интегрированные системы** | **Крупные интергированные системы** |
| *Внедрение* | Простое, коробочный вариант | Поэтапное или коробочный вариант. Более 4 месяцев. | Только поэтапное. Более 6-9 месяцев. | Поэтапное, сложное. Более 9-12 месяцев. |
| *Функциональная полнота* | Учетные системы (по направлениям) | Комплексный учет и управление финансами | Комплексное управление, учет, управление, производство | |
| *Соотношение затрат (лицензия / внедрение / оборудование)* | 1/0.5/2 | 1/1/1 | 1/2/1 | 1/1-5/1 |
| *Ориентировочная стоимость* | 5-50 тыс. дол. | 50-300 тыс. дол. | 200-500 тыс. дол. | 500 тыс. > 1 млн. дол. |

***1.4 Системотехнические аспекты проектирования АС***

**1.4.1 Определение понятий «проект» и «проектирование АС»**

Понятие «проект»  определяется в разных источниках по-разному. Например, известны такие определения проекта как процесса [60]:

проект – это временное предприятие (усилие), предназначенное для создания уникальных продуктов, услуг или результатов;

проект – это уникальный набор скоординированных действий с определенным началом и завершением, осуществляемых индивидуумом или организацией для решения специфических задач с определенным расписанием, затратами и параметрами исполнения.

**Системное проектирование** (Systems Engineering) в широком смысле – это междисциплинарный подход и средства, предназначенные для создания успешных систем. Он фокусируется на определении нужд потребителя и требуемой функциональности в начале цикла разработки, на документации требований, с переходом к конструкторскому синтезу и комплексной аттестации системы при полном учете таких проблем, как функционирование, производительность, испытания, изготовление, затраты и планирование, обучение и сопровождение, вплоть до вывода из эксплуатации. Системное проектирование интегрирует все нужные дисциплины и группы специалистов в командные усилия, формируя структурированный процесс разработки, который выполняется от создания концепции до осуществления продуктивной работы системы. В системном проектировании учитываются как нужды бизнеса, так и технические потребности всех клиентов для получения качественного продукта, который отвечает потребностям пользователей.

Проектирование АС – это выполнение проекта, целью которого является создание АС, обладающей заданными потребительскими свойствами и удовлетворяющей поставленным перед ней требованиям. Требования - это условия или возможности, которым должна соответствовать система, описанные в исходных данных, на основании которых проектируются и создаются АС.

Иногда проект определяют через целевой результат. Тогда проект АС – это техническая документация, в которой подробно описаны все решения  
по созданию и эксплуатации АС, и сама АС, созданная в ходе реализации проектных решений, введенная в действие и функционирующая в течение заданного временного интервала.

Созданная на основе проектных решений АС неотделима от объекта автоматизации, так как органично взаимодействует с ним или даже непосредственно встраивается в его функционирование, становясь его неотъемлемой частью. В связи с этим иногда считают, что АС состоит из проектных документов, а также организационно-технического и программно-информационного комплекса (ОТиПИК) (рис. 1.9). При этом ОТиПИК создается в результате реализации проектных решений и обеспечивает как внедрение, так и последующее функционирование проекта АС.

Рис. 1.9 Укрупненное представление состава и взаимодействий АС

**1.4.2 Требования к АС и модель FURPS+**

Выделяют требования к продукту и процессу. В нашем случае продукт - это проект и, соответственно, АС, а процесс – это процесс проектирования АС. Требования к продукту являются основополагающим классом требований, т.к. определяют его качество. Требования к процессу призваны снизить риск получить в результате проектирования продукт с опозданием либо ненадлежащего качества. Для этого проводится регламентация процессов и их контроль (аудит).

В связи с большой сложностью АС и многообразием требований к ним для упрощения проблемы проектирования используют абстракцию и декомпозицию требований. При этом требования разделяются по уровням. Уровни требований связаны, с одной стороны, с уровнем абстракции системы, с другой - с уровнем управления на предприятии. Обычно выделяют три уровня требований (примеры требований каждого уровня приведены в таблице 1.24):

·   Верхний уровень - бизнес-требования (business requirements), которые обычно формулируются топ-менеджерами, либо акционерами предприятия.

·  Второй уровень - требования пользователей (user requirements). Требования пользователей часто бывают плохо структурированными, дублирующимися, противоречивыми. Поэтому для создания АС важен третий уровень, в котором осуществляется формализация требований.

·   Третий уровень - функциональные требования (functional requirements).

Таблица 1.24

Примеры требований каждого уровня

|  |  |
| --- | --- |
| Название уровня требований | Примеры требований |
| Верхний уровень - бизнес-требования (business requirements) | Система должна сократить срок оборачиваемости обрабатываемых на предприятии заказов в три раза |
| Второй уровень - требования пользователей (user requirements) | Система должна представлять диалоговые средства для ввода исчерпывающей информации о заказе, последующей фиксации информации в базе данных и маршрутизации информации о заказе к сотруднику, отвечающему за его планирование и исполнение. |
| Третий уровень - функциональные требования (functional requirements) | Электронный заказ может быть создан, отредактирован, удален и перемещен с участка на участок |

Существуют объективные противоречия между требованиями различных уровней. Так, очевидным бизнес-требованием является требование о полноте информации, собираемой на рабочих местах пользователей в единую базу данных. Чем полнее информация - тем глубже база для анализа деятельности и принятия решений. С другой стороны, конкретному пользователю системы вполне может быть достаточно использования только той части информации, которая влияет на выполнение его основных функций.

Выделяют также функциональные и нефункциональные требования (характеристики продукта).

Функциональные требования регламентируют функционирование или поведение системы (behavioral requirements). Функциональные требования отвечают на вопрос "что должна делать система" в тех или иных ситуациях. Они устанавливают цели, задачи и сервисы, предоставляемые системой пользователю. Функциональные требования записываются, как правило, при посредстве предписывающих правил, например: "Система должна позволять кладовщику формировать приходные и расходные накладные".

Нефункциональные требования, соответственно, регламентируют внутренние и внешние условия или атрибуты функционирования системы. Выделяют следующие основные группы нефункциональных требований:

·  Внешние интерфейсы (External Interfaces),

·  Атрибуты качества (Quality Attributes),

·  Ограничения (Constraints).

Среди внешних интерфейсов в большинстве современных АС наиболее важным является интерфейс пользователя (User Interface, UI). Кроме того, выделяются интерфейсы с внешними устройствами (аппаратные интерфейсы), программные интерфейсы и интерфейсы передачи информации (коммуникационные интерфейсы).

Основные атрибуты качества достаточно хорошо раскрыты в модели FURPS+, используемой в спецификациях одной из проектных методологий Rational Unified Process (RUP), которая будет рассмотрена позднее.

Аббревиатура FURPS обозначает следующие категории требований:

·  Functionality (Функциональность)

·  Usability (Применимость)

·  Reliability (Надежность)

·  Performance (Производительность)

·  Supportability (Эксплуатационная пригодность).

Символ "+" расширяет FURPS-модель, добавляя к ней: ограничения проекта, требования выполнения, требования к интерфейсу, физические требования.

Кроме того, в спецификациях RUP выделяются такие категории требований, как: требования, указывающие на необходимость согласованности с некоторыми юридическими и нормативными актами; требования к лицензированию; требования к документированию.

**1.4.3 Предпосылки и принципы создания АС**

***1.4.3.1 Основные предпосылки создания АС и участники этого процесса***

Для того, чтобы начать создавать АС, на предприятии должны иметь место определенные предпосылки, в том числе:

-           наличие источников получения гарантированного эффекта, обеспечивающих окупаемость затрат, не ниже вложенных;

-           обеспечение требуемого уровня автоматизации процессов и объектов управления, которые должны соответствовать целям АС и требованиям к ее компонентам;

-           перестройка и модернизация организационной, производственно-технологической структур объекта в соответствии с требованиями к созданию АС и в сроки, обусловленные разработкой и вводом АС в действие;

-           подготовленность объекта к созданию АС в соответствии с требованием методических материалов;

-           гарантия комплектования АС техническими средствами (ТС) и программными средствами (ПС) в соответствии со спецификой технической документации, а также обеспечение АС персоналом требуемой квалификацией в заданный срок.

Центральный участник процесса создания и развития АС – это заказчик, для которого создается и развивается АС. Именно заказчик осуществляет создание системы.

***«Создание АС»*** предполагает комплексное встраивание всех элементов системы в производственно-экономическую деятельность заказчика в качестве неотъемлемой части этой деятельности, что требует от него освоения средств и методов работы в условиях АС.

Только приведенная в органичную взаимосвязь с автоматизируемым объектом АС может считаться созданной.

Для создания АС заказчик привлекает разработчика (исполнителя). Это либо сторонняя по отношению к заказчику научно-исследовательская проектная организация либо собственная подчиненная заказчику структурная единица. Организационно-правовые отношения заказчиков и разработчиков регулируются хозяйственными договорами.

Кроме заказчика и основного исполнителя в создании АС могут участвовать соисполнители – привлекаемые специализированные организации: проектные; строительные; пусконаладочные (специализирующиеся на проектировании, строительстве и т.д.; связанные с созданием новых зданий, сооружений, наладкой и монтажом оборудования, программных средств и т.д.). Отношения с соисполнителями также регулируются хозяйственными договорами, заключенными между ними и заказчиками, либо между ними и основным исполнителем, который является в этом случае субподрядчиком.

Функции, выполняемые Заказчиком и Разработчиком АС, определяются принятой методологией и технологией проектирования.

***1.4.3.2 Основные принципы создания АС***

При создании АС необходимо руководствоваться рядом основных принципов, представленных в таблице 1.25.

Таблица 1.25

Основные принципы создания АС

|  |  |  |
| --- | --- | --- |
| № | Название принципа | Суть принципа |
| 1 | Принцип системного подхода | При создании, развитии или функционировании АС должны быть установлены и сохранены связи между структурными элементами, обеспечивающие целостность системы |
| 2 | Принцип непрерывного развития | Создание условий для постоянного совершенствования АС |
| 3 | Принцип новых задач | Следует не просто использовать вычислительную технику для решения известных задач в рамках традиционно сложившихся методов и приемов, а перестраивать эти методы в соответствии с большими возможностями средств вычислительной техники (СВТ), ставить и решать новые ранее недоступные задачи |
| 4 | Принцип первого руководителя | Работы по созданию и внедрению АС должен возглавлять первый руководитель объекта или его ближайший заместитель. Принцип важен в связи с необходимостью решать много организационных вопросов, входящих в компетенцию первого руководителя объекта |
| 5 | Принцип совместимости и согласованности | Обеспечение возможности взаимодействия АС различных видов и уровней в процессе их совместного функционирования на объекте с учетом согласованности пропускных способностей всех элементов и связей |
| 6 | Принцип стандартизации и унификации | Рациональное применение типовых, унифицированных и стандартизированных элементов при создании и развитии АС. Этот принцип должен рассматриваться в тесном единстве с принципами непрерывного развития и новых задач. |
| 7 | Принцип человеко-машинной работы | Система должна допускать возможность, где это необходимо и обосновано, непосредственной работы специалистов, пользователей на ЭВМ в диалоговом режиме, причём должна обеспечиваться дружественность этого диалога и создание удобных АРМ |
| 8 | Принцип однократного ввода данных в ЭВМ и многократного их использования | В системе должен обеспечиваться однократный ввод данных в ЭВМ и многократное их использование различными пользователями |
| 9 | Принцип автоматизации документооборота | Предполагает автоматизированный сбор, хранение, обработку и передачу документов, т.е. ориентирует на использование безбумажных информационных технологий. |
| 10 | Принцип функциональ-ной надежности, помехоустойчивости и живучести | Определяет необходимость создания систем, способных выполнять свои функции в течение заданного промежутка времени в условиях действия дестабилизирующих факторов и возмущений |
| 11 | Принцип эффективности | Заключается в необходимости достижения рационального соотношения между затратами на создании АС и целевыми эффектами, получаемыми при ее функционировании |

***1.4.3.3 Принципы McKinsey успешного построения АС***

Анализируя опыт развертывания корпоративных информационных систем с учетом изложенных выше предпосылок и принципов создания АС, эксперты аналитического агентства McKinsey в 1997 г. сформулировали интегральные принципы, следование которым является обязательным условием успеха:

1.  **Экономическая необходимость.** Основанием для внедрения новой информационной системы на предприятии должна быть экономическая необходимость, а не просто появление новых технологий.

2.  **Финансовая целесообразность.** Объем финансирования новой информационной системы должен определяться соображениями финансовой выгоды.

3.  **Простота структуры.** Новая информационная система должна иметь простую и гибкую структуру.

4.  **Быстрая отдача.** Нужно выбирать решения, которые приносят конкретную пользу бизнесу практически с момента внедрения.

5.  **Постоянное совершенствование.** Следует постоянно совершенствовать корпоративную информационную систему, добиваясь оптимальных значений производительности, надежности и удобства использования.

6.  **Органичное взаимодействие подразделений бизнеса и IT.** IT-служба должна хорошо разбираться в бизнесе, а специалисты других подразделений — знать методы эффективного использования IT; внедрять новую информационную систему они должны совместно.

На основании сформулированных принципов аналитическое агентство McKinsey предложило для воплощения каждого из них следующие рекомендации, представленные в табл. 1.26.

Таблица 1.26

Рекомендации для воплощения принципов  McKinsey

|  |  |  |
| --- | --- | --- |
| № | Принципы | Рекомендации |
| 1 | **Экономиическая необходимость** | Управляйте IT так же, как остальными бизнес-функциями — как одним из средств выполнения конкретной работы. Непосредственно свяжите IT с важными для бизнеса стратегиями, основными стоимостными факторами и повседневными деловыми процессами. Назначьте руководителей подразделений ответственными за новые приложения. Поручите IT-службе создание и поддержку эффективной производственной архитектуры. Не относитесь к информационной системе как к «черному ящику» — обязуйте технических специалистов применять деловую лексику. |
| 2 | **Финансовая целесообразность** | Подходите к решениям о новых крупных инвестициях в IT, как к остальным финансовым решениям. Свяжите инвестиции с реальными задачами совер-шенствования бизнеса. Используйте при принятии решений анализ затрат и преимуществ, который обеспечит упорядоченность и систематичность. Не применяйте в качестве единственного критерия при разработке проекта АС снижение стоимости. Распространите принятые в бизнесе подходы на менее определенные, менее поддающиеся количественному учету и более важные стратегические решения. Выходите за рамки отдельных проектов, учитывая при разработке приложений и производственной архитектуры долговременные перспективы. Оценивайте текущие расходы на IT, исходя из поставленных целей в области стоимости и обслуживания. Избегайте крупных единовременных затрат на аппаратно-программное обеспечение; стремитесь к постоянному обновлению. |
| 3 | **Простота структуры** | Устанавливайте стандарты архитектуры АС и глубоко анализируйте плюсы и минусы использования иных стандартов. Упрощайте систему, сокращая число используемых технологий и платформ. Применяйте для приложений модульную архитектуру. Используйте кросс-платформные решения для повышения гибкости и простоты внедрения приложений. Консервативно подходите к выбору технологий. Условием выбора новейших технологий должна быть значительная бизнес-отдача. Учитывайте такие коммерческие аспекты, как отраслевые стандарты и вероятность получения в будущем технической поддержки. Подходите к конфигурированию баз данных и приложений стратегически, стараясь обеспечить успешно работающую оптимальную структуру. |
| 4 | **Быстрая отдача** | Осуществляйте постепенный переход к новым решениям, а не глобальную замену. Разработку новых систем разбивайте на этапы. Устанавливайте промежуточные цели (с интервалом, как правило, не больше шести месяцев), которые обеспечивают реальное продвижение в бизнесе. Используйте по возможности стандартное проверенное программное обеспечение; сведите его модификацию к минимуму. Сконцентрируйте усилия на тех 20% функций, которые отвечают за 80% деятельности компании, и быстро добейтесь их полной реализации. Если возможно, реализуйте действующий макет проекта. Постоянно сравнивайте развитие крупных проектов с эталонами и с запланированными результатами; при необходимости осуществляйте коррекцию. Регулярно проводите ревизию завершенных проектов и переоценку этапов разработки |
| 5 | **Постоянное совершенствование** | Относитесь к управлению эксплуатацией АИС, как к управлению заводом: определите стандарты производительности и конкретные задачи улучшения управления стоимостью, качества обслуживания и времени отклика. Реорганизуйте работу информационных систем для получения наиболее экономичной модели. Централизуйте основные функции (такие как работа справочной службы, информационного центра и управление сетью). Стандартизируйте конфигурацию настольных компьютеров и ограничьте возможность ее изменения. Консолидируйте закупки в области информа-ционных технологий. Строго контролируйте эффективность эксплуатации системы и передавайте обслуживание неэффективных функций сторонним организациям. Разработайте эталонные тесты, чтобы можно было ставить конкретные цели в области поддержки и стоимости эксплуатации системы. |
| 6 | **Органичное взаимодействие подразделений бизнеса и IT** | Правильно распределите обязанности. Генеральный директор должен активно участвовать в принятии решений в области IT, определяя верное направление. Руководитель IT-службы должен быть отнесен к бизнес-руководителям и входить в состав высшего руководства. Обеспечьте руководство со стороны представителей бизнес-подразделений и их осведомленность в области IT. Обучите специалистов по IT основам бизнеса. Выработайте совместную процедуру принятия обоснованных решений руководителями бизнес-подразделений и руководителем IT-службы. Стимулируйте изучение деловых возможностей. Требуйте представления различных точек зрения и всесторонних обсуждений при решении сложных вопросов. Создайте условия для высокопроизводительной работы IT-службы, включая материальное стимулирование для повышения производительности. |

**1.4.4 Понятие жизненного цикла системы и виды моделей жизненного цикла**

Одним из базовых понятий, связанных с проектированием АС, является понятие ее жизненного цикла (ЖЦ). ЖЦ АС - это непрерывный процесс, который начинается с момента принятия решения о необходимости ее создания и заканчивается в момент ее полного изъятия из эксплуатации.

ЖЦ АС состоит из некоторой последовательности стадий и выполняемых на них поэтапно процессов. Для каждого этапа определяются состав и последовательность выполняемых работ, получаемые результаты, методы и средства, необходимые для выполнения работ, роли и ответственность участников и т.д. Такое формальное описание ЖЦ АС позволяет спланировать и организовать процесс коллективной разработки и обеспечить управление этим процессом.

Таким образом, ЖЦ АС можно представить как ряд событий, происходящих с системой в процессе ее создания и использования. Такое представление обычно называют моделью ЖЦ, которая отражает различные состояния системы, начиная с момента возникновения необходимости в данной АС и заканчивая моментом ее полного выхода из употребления. Модель ЖЦ - структура, содержащая процессы, действия и задачи, которые осуществляются в ходе разработки, функционирования и сопровождения АС в течение всей ее жизни, от определения требований до завершения ее использования.

**Стратегия проектирования АС**определяется использованием соответствующей модели жизненного цикла, определяющей последовательность стадий проектирования и выполняемых в них процессов.

Известно не менее семи моделей ЖЦ АС, из которых наибольшее распространение получили следующие две основные модели ЖЦ:

·      каскадная (или водопадная) модель (70-85 г.г.);

·      спиральная модель (86-90 г.г.).

В изначально существовавших однородных АС каждое приложение представляло собой единое целое. Для разработки такого типа приложений применялся каскадный способ. Его основной характеристикой является разбиение всей разработки на этапы, причем переход с одного этапа на следующий происходит только после того, как будет полностью завершена работа на текущем (рис. 1.10). Каждый этап завершается выпуском полного комплекта документации, достаточной для того, чтобы разработка могла быть продолжена другой командой разработчиков.

Положительные стороны применения каскадного подхода заключаются в следующем:

·      на каждом этапе формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;

·      выполняемые в логичной последовательности этапы работ позволяют планировать сроки завершения всех работ и соответствующие затраты.

Рис. 1.10  Каскадная схема разработки АС

Каскадный подход хорошо зарекомендовал себя при построении АС, для которых в самом начале разработки можно достаточно точно и полно сформулировать все требования, с тем чтобы предоставить разработчикам свободу реализовать их как можно лучше с технической точки зрения. В эту категорию попадают сложные расчетные системы, системы реального времени и другие подобные задачи. Однако, в процессе использования этого подхода обнаружился ряд его недостатков, вызванных прежде всего тем, что реальный процесс создания АС никогда полностью не укладывался в такую жесткую схему. В процессе создания АС постоянно возникала потребность в возврате к предыдущим этапам и уточнении или пересмотре ранее принятых решений. В результате реальный процесс создания АС принимал следующий вид (рис. 1.11):

Рис. 1.11  Реальный процесс разработки АС  по каскадной схеме

Основным недостатком каскадного подхода является существенное запаздывание с получением результатов. Согласование результатов с пользователями производится только в точках, планируемых после завершения каждого этапа работ, требования к АС "заморожены" в виде технического задания на все время ее создания. Таким образом, пользователи могут внести свои замечания только после того, как работа над системой будет полностью завершена. В случае неточного изложения требований или их изменения в течение длительного периода создания АС пользователи получают систему, не удовлетворяющую их потребностям. Модели (как функциональные, так и информационные) автоматизируемого объекта могут устареть одновременно с их утверждением.

Для преодоления перечисленных проблем была предложена спиральная модель ЖЦ (рис. 1.12), делающая упор на начальные этапы ЖЦ: анализ и проектирование. На этих этапах реализуемость технических решений проверяется путем создания прототипов. Каждый виток спирали соответствует созданию фрагмента или версии АС, на нем уточняются цели и характеристики проекта, определяется его качество и планируются работы следующего витка спирали. Так углубляются и последовательно конкретизируются детали проекта и в результате выбирается обоснованный вариант, который доводится до реализации.

Разработка итерациями отражает объективно существующий спиральный цикл создания системы. Неполное завершение работ на каждом этапе позволяет переходить на следующий этап, не дожидаясь полного завершения работы на текущем. При итеративном способе разработки недостающую работу можно будет выполнить на следующей итерации. Главная же задача - как можно быстрее показать пользователям системы работоспособный продукт, тем самым активизируя процесс уточнения и дополнения требований.

Рис 1.12  Спиральная модель ЖЦ

Основная проблема спирального цикла - определение момента перехода на следующий этап. Для ее решения необходимо ввести временные ограничения на каждый из этапов жизненного цикла. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа закончена. План составляется на основе статистических данных, полученных в предыдущих проектах, и личного опыта разработчиков.

**1.4.5  Определение понятий методологии и технологии проектирования**

Как правило, создание АС базируется напарадигме*«Методология – Метод – Нотация - Средство»*и связанном с ними понятии*«Технология проектирования»*. Определение этих терминов дано в таблице 1.27.

Таблица 1.27

Определение ключевых понятий, связанных с проектированием АС

|  |  |
| --- | --- |
| Понятие | Определение |
| *Парадигма* | Система понятий и/или взглядов, используемых для осмысленной деятельности в некоторой предметной области |
| *Методология* | Совокупность правил, которые определяют: 1) руководящие указания для оценки и выбора проекта разрабатываемой АС, 2) шаги работы и их последовательность, 3) способы распределения и назначения методов |
| *Метод (методика)* | Систематическая процедура или техника генерации компонентов АС, включая их описание с использованием определенной нотации. Например, проектирование потоков и структур данных |
| *Нотации* | Правила и способы описания структуры системы, элементов данных, этапов обработки, использующие графы, диаграммы, таблицы, блок-схемы, формальные и естественные языки |
| *Средство* | Инструментарий для применения методов на практике, т.е. для их поддержки и усиления. Эти инструменты поддерживают работу пользователей при создании и редактировании графического проекта в диалоговом режиме, способствуют организации проекта в виде иерархии уровней абстракции, выполняют проверки соответствия компонентов и т.д. |
| *Технология проектирования* | Составная часть методологии, представляющая собой совокупность методов и средств проектирования, в том числе организационных приемов и инструментально-технических средств, ориентированных на создание или модернизацию проекта. |

Благодаря эффективной технологии создание АС становится процессом создания промышленных, строго регламентированных изделий.

Основой технологии проектирования является ***технологический процесс***, который определяет действия, их последовательность, исполнителей и ресурсы необходимые для выполнения этих действий.

Технологический процесс состоит из относительно самостоятельных фрагментов – технологических операций. Графическое представление технологии, как упорядоченной взаимосвязанной совокупности технологических операций, образует  ***технологическую сеть проектирования***.

*Общие требования к технологии проектирования*:

1.      Охват технологией большого числа этапов ЖЦ АС;

2.      Взаимосвязь этапов;

3.      Обеспечение связи с другими проектами (например, преемственность стандартов);

4.      Обеспечение представления семантических требований к создаваемой системе и минимизация потерь информации при переходе от одного уровня представления к другому.

Последнее требование можно детализировать.

Современные технологии проектирования АС должны обеспечивать представление следующей информации:

-      описание объекта автоматизации, а также места разрабатываемой АС и целей, которые должны быть достигнуты в процессе разработки системы, короче: описание предметной области и целей автоматизации;

-      описание функциональных возможностей АС, достаточное для решения вопроса о том, что поставленные цели автоматизации достижимы;

-      спецификации проекта, гарантирующие достижение заданных технических характеристик системы (спецификация – это достаточно точное и полное описание требований, ясно сформулированное в терминах, характерных для целей данной задачи, а не для ее реализации);

-      описание (план) реализации предлагаемой системы, достаточное для оценки времени разработки системы и необходимых для этих целей трудозатрат.

Конкретные методы на каждом шаге проектирования могут быть заданы жестко, если они однозначно определены в выбранной технологии, но и могут выбираться проектировщиком в каждом конкретном случае с учетом опыта и квалификации, если технология допускает творчество разработчика.

**1.4.6 Сущность внешнего и внутреннего проектирования систем**

Можно выделить логические составляющие проектирования: внешнее и внутреннее (или, соответственно, проектирование на макро-  и микро- уровне).

***Внешнее проектирование*** решает задачу, которая формулируется так – выяснить взаимодействие системы с внешней средой, и определить, что и зачем будет делать система и почему она должна действовать так, а не иначе. При этом формируется укрупненный облик будущей системы.

***Внутреннее проектирование*** определяет содержание самой системы и отвечает на следующие системные вопросы: как, какими методами, какими способами и средствами система будет выполнять свои функции; кто, где и когда будет выполнять необходимые для этого операции и процедуры.

Внешнее и внутреннее проектирование связаны между собой и, как правило, итерационно повторяются друг за другом в процессе разработки системы. Начинать нужно всегда с внешнего проектирования. При этом вначале задаются максимальные требования к системе, как если бы она обладала идеальными возможностями бесконечной мощности. Затем определяют можно ли известными методами удовлетворить эти требования принципиально (не обращая внимания на то, может ли разработчик получить требуемую возможность в данный момент).

При положительном прогнозе оценивается реальность использования нужных методов и средств. Если это невозможно, то выясняется, какие другие реально допустимые методы и средства могут быть использованы, в какой степени при этом видоизменяются требования к системе, насколько должны быть снижены исходные требования и является ли это снижение приемлемым для того, чтобы АС оставалась эффективной. При отрицательном ответе работу над системой следует прекратить до тех пор, пока не будут найдены пути преодоления возникших трудностей.

Нельзя подгонять характеристики АС под те средства, которые доступны разработчику, поскольку успешность результатов будет нулевой или даже приведет к отрицательным последствиям.

На этапе внутреннего проектирования можно использовать концептуальные подходы, базирующиеся на методах: Единичной нити; Большой нагрузки; Конфликтных ситуаций. Суть методов, а также их достоинства и недостатки представлены в таблице. 1.28.

Таблица 1.28

Характеристика методов внутреннего проектирования

|  |  |
| --- | --- |
| Название | Суть метода, его достоинства и недостатки |
| Метод единичной нити | Заключается в анализе и последующем синтезе для разрабатываемой системы реакции системы в целом и ее элементов на каждый возможный вид входных воздействий в отдельности. Результат анализа удобно представить в виде цепочки последовательных реакций различных элементов системы, обеспечивающих нужную реакции системы в целом. В отдельных случаях целесообразно прослеживать цепочку реакций в обратном направлении: от выхода к входу (например, при анализе процесса сборки изделия). Анализ информационных потоков следует проводить, начиная с входа (например, для выяснения какие внутренние документы порождаются системой при появлении на ее входе одного заказа). В результате строятся модели материальных и информационных потоков.  Метод  помогает четче структурировать систему (выявить составные части, их связи и локализовать границы системы), оценить время реакции подразделений и системы в целом на различные входные воздействия.  Достоинства: простота и наглядность.  Недостатки: невозможность формализации в рамках метода задачи синтеза структуры и поиска возможных решений по отдельным элементам системы; трудность анализа при большом числе входных воздействий различного рода. |
| Метод большой нагрузки | Как и в предыдущем методе, в качестве инструмента используется модель (графическая, аналитическая, графоаналитическая) движения материальных и информационных потоков. Однако, в отличие от метода единичной нити, где используются логические методы анализа, здесь используются модели и методы теории массового обслуживания, так как большая нагрузка влечет появление очередей.  Достоинства: возможность формализации анализа и синтеза; учет единичной нити.  Недостатки: не учитываются отклонения от нормального режима работы системы и ее элементов, а также непредвиденные воздействия внешней среды. |
| Метод конфликтных ситуаций | Заключается в выявлении возможных конфликтов в системе, их анализе и принятии решений о поведении системы в каждом аномальном случае. Простейший вариант реакции – игнорирование аномальных сигналов, для чего требуется блок обнаружения. Но это не всегда допустимо. В таком случае надо считать каждый тип отклонений особым сигналом и определять реакцию на него системы теми же методами, которые применялись к обычным входным воздействиям, а при анализе – предусматривать необходимые меры по обработке конфликтных ситуаций.  Достоинства: обеспечивает повышение надежности и живучести системы.  Недостатки: трудность выявления конфликтных ситуаций; трудность или невозможность формализации; громоздкость при большой нагрузке. |

**1.4.7  Эволюция подходов к разработке АС**

Индустрия разработки АС развивалась, начиная с 1960-х годов, и к концу ХХ века приобрела вполне законченные формы.

В начальном периоде развития проектной индустрии в основу методологии *проектирования АС* был заложен подход **"снизу-вверх"**, когда система создавалась как набор приложений, наиболее важных в данный момент для поддержки деятельности предприятия. Основной целью этих проектов было не создание тиражируемых продуктов, а обслуживание текущих потребностей конкретного учреждения в рамках "лоскутной автоматизации". Типичным явлением в таких системах была непрерывная доработка программных продуктов для удовлетворения все новых и новых пожеланий отдельных работников. Как следствие, и работа программистов, и создаваемые АС вызывали недовольство руководителей и пользователей системы.

Следующий период связан с осознанием того факта, что существует потребность в достаточно стандартных программных средствах автоматизации деятельности различных учреждений и предприятий. Системы начали создавать согласно методологии проектирования на основе подхода **"сверху-вниз"**, т.е. в предположении, что одна типовая программа должна удовлетворять потребности многих пользователей. Однако сама идея использования универсальной программы накладывает существенные ограничения на возможности разработчиков по формированию структуры базы данных, экранных форм, по выбору алгоритмов расчета. Заложенные "сверху" жесткие рамки не дают возможности гибко адаптировать систему к специфике деятельности конкретного предприятия. Решение этих задач требует серьезных доработок системы. На практике выяснилось, что материальные и временные затраты на внедрение системы и ее доводку под требования заказчика, как правило, значительно превышают запланированные показатели.

Согласно статистическим данным, собранным Standish Group (США), из 8380 проектов, обследованных в США в 1994 году, которые выполнялись на основе описанного подхода, неудачными оказались более 30% проектов, общая стоимость которых превышала 80 миллиардов долларов. При этом оказались выполненными в срок лишь 16% от общего числа проектов, а перерасход средств составил 189% от запланированного бюджета.

В то же время, потребность в АС постоянно росла, как и требования к их качеству и возможности комплексного использования корпоративных данных в управлении и планировании деятельности организаций. На рубеже веков возникло четкое понимание необходимости формирования новой современной методологии (или даже ряда методологий) построения АС.

В результате была предложена и начала активно распространяться методология проектирования, реализующая подход «**многокомпонентности».**Его суть в том, что АС создается как совокупность взаимоувязанных компонентов (универсальных функциональных модулей). При этом определенное подмножество модулей образует ядро системы, выполняющее базовые стандартные действия, инвариантные по отношению к конкретной предметной области внедрения АС. В результате, при адаптации АС к принятым в организации условиям работы, проведение модернизации одного из компонентов не затрагивает центральную часть (ядро) и другие ее компоненты, что значительно повышает надежность, продолжительность жизни АС и обеспечивает наиболее полное выполнение требуемых функций.

Внедрение методологии должно приводить к снижению сложности процесса создания АС за счет полного и точного описания этого процесса, а также применения современных методов и технологий создания АС в течение всего жизненного цикла АС - от замысла до реализации.

**1.4.8  Общая характеристика современной методологии создания АС**

Цель современной методологии заключается в регламентации процесса проектирования АС и обеспечении такого управления этим процессом, при котором гарантировано выполнение требований как к самой АС, так и к характеристикам процесса разработки. Основные задачи, решению которых должна способствовать такая методология проектирования АС, представлены в таблице 1.29.

Таблица 1.29

Задачи, решаемые современной методологией проектирования АС

|  |  |
| --- | --- |
| № | Суть задачи |
| 1 | Обеспечивать создание АС, отвечающих целям и задачам организации, а также предъявляемым требованиям по автоматизации деловых процессов заказчика |
| 2 | Гарантировать создание системы с заданным качеством в заданные сроки и в рамках установленного бюджета проекта |
| 3 | Поддерживать удобную дисциплину сопровождения, модификации и наращивания системы |
| 4 | Обеспечивать преемственность разработки, т.е. использование в разрабатываемой АС существующей информационной инфраструктуры организации (задела в области информационных технологий) |

**Согласно современной методологии, процесс создания АС представляет собой процесс построения и последовательного преобразования ряда согласованных моделей на всех этапах ЖЦ АС.** На каждом этапе ЖЦ создаются специфичные для него модели – объекта автоматизации (организации и т.п.), требований к АС, проекта АС, требований к приложениям и т.д. Модели формируются рабочими группами команды проекта, сохраняются и накапливаются в репозитории проекта. Создание моделей, их контроль, преобразование и предоставление в коллективное пользование осуществляется с использованием специальных программных инструментов - CASE-средств.

Для современной методологии создания АС характерно то, что проектирование охватывает три основные области:

·      проектирование объектов данных, которые будут реализованы в базе данных;

·      проектирование приложений, экранных форм, отчетов и других программ, которые будут обеспечивать выполнение запросов к данным;

·      выбор конкретной вычислительной и коммуникационной среды и/или технологии, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры (файл-сервер или клиент-сервер), параллельной или распределенной обработки данных и т.п.

Процесс создания АС делится на ряд шагов (стадий, этапов), ограниченных некоторыми временными рамками и заканчивающихся выпуском конкретного продукта (моделей, программ, документации и пр.).

Обычно выделяют следующие *этапы создания АС*: формирование требований к системе, проектирование, реализация, тестирование, ввод в действие, эксплуатация и сопровождение.

Начальным этапом процесса создания АС является моделирование бизнес-процессов, протекающих в организации и реализующих ее цели и задачи. Такая модель организации позволяет сформулировать основные требования к АС. Это фундаментальное положение методологии обеспечивает объективность в выработке требований к проектированию системы. Множество моделей описания требований к АС затем преобразуется в систему моделей, описывающих концептуальный проект АС. Формируются модели архитектуры АС, требований к программному обеспечению (ПО) и информационному обеспечению (ИО). Затем формируется архитектура ПО и ИО, выделяются корпоративные БД и отдельные приложения, формируются модели требований к приложениям и проводится их разработка, тестирование и интеграция.

Задача формирования требований к АС является одной из наиболее ответственных, трудно формализуемых и наиболее дорогих и тяжелых для исправления в случае ошибки. Современные инструментальные средства и программные продукты позволяют достаточно быстро создавать АС по готовым требованиям. Но зачастую эти системы не удовлетворяют заказчиков, требуют многочисленных доработок, что приводит к резкому удорожанию фактической стоимости АС. Основной причиной такого положения является неправильное, неточное или неполное определение требований к АС на этапе анализа.

На этапе проектирования прежде всего формируются модели данных. Полученная в процессе анализа информационная модель сначала преобразуется в логическую, а затем в физическую модель данных.

Параллельно с проектированием схемы базы данных выполняется проектирование процессов, чтобы получить спецификации (описания) всех модулей АС. Оба эти процесса проектирования тесно связаны, поскольку часть бизнес-логики обычно реализуется в базе данных (ограничения, триггеры, хранимые процедуры). Главная цель проектирования процессов заключается в отображении функций, полученных на этапе анализа, в модули АС. При проектировании модулей определяют интерфейсы программ: разметку меню, вид окон, горячие клавиши и связанные с ними вызовы.

Конечными продуктами этапа проектирования являются  схема базы данных и набор спецификаций модулей системы (они строятся на базе моделей функций).

Кроме того, на этапе проектирования разрабатывается архитектура АС, включая выбор платформы и операционной системы, а также определение конкретных характеристик архитектуры, перечисленных в таблице 1.30.

Таблица 1.30

Конкретные характеристики архитектуры АС,

определяемые на этапе проектирования

|  |  |
| --- | --- |
| № | Определяемая характеристика архитектуры |
| 1 | Будет ли это архитектура "файл-сервер" или "клиент-сервер" |
| 2 | Будет ли это 3-уровневая архитектура со следующими слоями: сервер БД, сервер приложений, клиентское ПО |
| 3 | Будет ли БД централизованной или распределенной. Если БД будет распределенной, то какие механизмы поддержки согласованности и актуальности данных будут использоваться |
| 4 | Будет ли БД однородной, то есть, будут ли все серверы БД продуктами одного и того же производителя. Если БД не будет однородной, то какое ПО будет использовано для обмена данными между СУБД разных производителей |
| 5 | Будут ли для достижения должной производительности использоваться параллельные серверы БД |

Этап проектирования завершается разработкой технического проекта АС.

На этапе реализации осуществляется создание ПО АС, установка технических средств, разработка эксплуатационной документации.

Этап тестирования обычно оказывается распределенным во времени.

После завершения разработки отдельного модуля системы выполняют автономный тест, который преследует две основные цели:

·      обнаружение отказов модуля (жестких сбоев);

·      проверка соответствия модуля спецификации (наличие всех необходимых функций, отсутствие лишних функций).

После того как автономный тест успешно пройден, модуль включается в состав разработанной части системы и группа сгенерированных модулей проходит тесты связей, которые должны отследить их взаимное влияние.

Далее группа модулей тестируется на надежность работы, то есть проводят, во-первых, тесты имитации отказов системы, а во-вторых, тесты наработки на отказ, в том числе тесты, имитирующие пиковую нагрузку на систему.

Затем весь комплект модулей проходит системный тест - тест внутренней приемки продукта, показывающий уровень его качества. Сюда входят тесты функциональности и тесты надежности системы.

Последний тест АС - приемо-сдаточные испытания. Такой тест предусматривает показ информационной системы заказчику и должен содержать группу тестов, моделирующих реальные бизнес-процессы, чтобы показать соответствие реализации требованиям заказчика.

Необходимость контролировать процесс создания АС, гарантировать достижение целей разработки и соблюдение различных ограничений (бюджетных, временных и пр.) привело к широкому использованию в этой сфере методов и средств программной инженерии, а именно: структурного анализа, объектно-ориентированного моделирования, CASE-систем.

[1 ОСНОВЫ ПОСТРОЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ (АС)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758337)

[1.1 Определение и классификация АС и их компонентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758338)

[1.1.1 Понятие системы и основные свойства систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758339)

[1.1.2 Автоматизированные системы и их классы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758340)

[*1.1.2.1 Определение понятия автоматизированной системы*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758341)

[*1.1.2.2 Классификация автоматизированных систем*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758342)

[1.1.3 Функциональные и обеспечивающие подсистемы АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758343)

[1.2 Основные этапы развития и современное состояние автоматизированных систем и информационных технологий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758344)

[1.2.1 История развития АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758345)

[1.2.2 Общая характеристика современного состояния информационных технологий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758346)

[1.2.3 Влияние информационных технологий и автоматизированных систем на деятельность предприятий](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758347)

[1.3 Интегрированная автоматизированная система управления как пример большой сложной АС и ее зарубежные аналоги](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758348)

[1.3.1 Определение интегрированной автоматизированной системы управления предприятием](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758349)

[1.3.2 Принятая за рубежом концепция автоматизации ICAM (CIM)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758350)

[1.3.3 Характеристика MRP-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758351)

[1.3.4 Характеристика MRPII-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758352)

[1.3.5 Характеристика ERP-систем как компонента ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758353)

[1.3.6 Характеристика CSRP-систем как компонентов ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758354)

[1.3.7 Характеристика  ERPII-систем как компонентов ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758355)

[1.3.8 Экономико-математические методы и модели в компонентах ICAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758356)

[1.3.9 Характеристика рынка автоматизированных систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758357)

[1.4 Системотехнические аспекты проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758358)

[1.4.1 Определение понятий «проект» и «проектирование АС»](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758359)

[1.4.2 Требования к АС и модель FURPS+](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758360)

[1.4.3 Предпосылки и принципы создания АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758361)

[*1.4.3.1 Основные предпосылки создания АС и участники этого процесса*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758362)

[*1.4.3.2 Основные принципы создания АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758363)

[*1.4.3.3 Принципы McKinsey успешного построения АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758364)

[1.4.4 Понятие жизненного цикла системы и виды моделей жизненного цикла](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758365)

[1.4.5  Определение понятий методологии и технологии проектирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758366)

[1.4.6 Сущность внешнего и внутреннего проектирования систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758367)

[1.4.7  Эволюция подходов к разработке АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758368)

[1.4.8  Общая характеристика современной методологии создания АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk1\1%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc469758369)

[2 ИНДУСТРИАЛЬНЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627147)

[2.1 Классификация и описание методов проектирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627148)

[*2.1.1 Схема* *классификации методов проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627149)

[*2.1.2 Метод оригинального проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627150)

[*2.1.3 Методы типового проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627151)

[*2.1.4 Метод автоматизированного проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627152)

[2.2 Регламентация процессов проектирования АС отечественными и международными стандартами](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627153)

[*2.2.1 Реализуемые концептуальные подходы* *и общая характеристика стандартов проектирования АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627154)

[2.2.1.1 Cleanroom и RAD - концептуальныеподходы к организации процесса проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627155)

[2.2.1.2 Перечень и общая характеристика стандартов проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627156)

[*2.2.2  Основы канонического проектирования согласно ГОСТ 34.601-90*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627157)

[*2.2.2.1  Перечень и состав стадий создания АС согласно отечественным стандартам*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627158)

[*2.2.2.2  Виды и структура документации на АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627159)

[*2.2.2.3  Документ «Технико-экономическое обоснование создания АС»*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627160)

[*2.2.2.4  Документ «Техническое задание на создание АС»*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627161)

[*2.2.2.5  Состав комплексов документов стадий проектирования АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627162)

[*2.2.2.6  Разновидности и общая характеристика схем в документации АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627163)

[*2.2.2.7  Правила выполнения схем в документации АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627164)

[*2.2.2.8  Правила оформления текстовых документов*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627165)

[2.2.3  Международный стандарт ISO/IEC 12207](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627166)

[2.2.4  Международный стандарт ISO/IEC 15288](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627167)

[2.2.5  Корпоративный стандарт CDM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627168)

[2.2.6  Корпоративный стандарт RUP](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627169)

[2.2.7  Корпоративный стандарт MSF](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627170)

[2.2.8  Корпоративный стандарт DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627171)

[2.2.8.1 Методологии проектирования АС как программные продукты](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627172)

[2.2.8.2 Сущность методологии DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627173)

[2.2.8.3 Модели, создаваемые в рамках подхода DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627174)

[2.2.9 Семейство гибких методологий Agile](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627175)

[2.2.9.1 Общая характеристика гибких методологий разработки семейства Agile](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627176)

[2.2.9.2 Экстремальное программирование как пример гибкой методологии разработки](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627177)

[2.2.9.3  Обзор гибкой методологии разработки систем OpenUP](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627178)

[2.2.9.4  Гибкая методология разработки систем SCRUM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627179)

**2 ИНДУСТРИАЛЬНЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ АС**

***2.1 Классификация и описание методов проектирования***

***2.1.1 Схема* *классификации методов проектирования***

Развитие проектных методологий сопровождалось развитием  индустриальных методов проектирования АС с целью снизить затраты и сократить сроки разработки при одновременном повышении качества создаваемых систем. Наиболее часто используется следующая классификация методов проектирования (рис. 2.1).

Рис. 2.1 Методы проектирования АС

***2.1.2 Метод оригинального проектирования***

Метод ориентирован на создание индивидуальных проектов, отражающих особенности только одного соответствующего проекту объекта. Этим же методом создаются методики проведения проектных работ. В качестве средств проектирования используются библиотеки стандартных процедур, реализующих типовые операторы обработки данных. Достоинства и недостатки метода перечислены в таблице 2.1.

Таблица 2.1.

Достоинства и недостатки метода оригинального проектирования

|  |  |
| --- | --- |
| Достоинства | ·      детальный учет особенностей объекта автоматизации |
| Недостатки | ·      высокая трудоемкость  ·      большие сроки проектирования  ·      трудности, возникающие при модернизации объекта |

***2.1.3 Методы типового проектирования***

Типовое проектирование АС предполагает создание системы из готовых типовых элементов. Основополагающим требованием для применения методов типового проектирования является возможность декомпозиции проектируемой АС на множество составляющих компонентов (подсистем, комплексов задач, программных модулей и т.д.). Для реализации выделенных компонентов выбираются имеющиеся на рынке типовые проектные решения, которые настраиваются на особенности конкретного предприятия.

Типовое проектное решение (ТПР) - это тиражируемое (пригодное к многократному использованию) проектное решение.

Принятая классификация ТПР основана на уровне декомпозиции системы. Выделяются следующие классы ТПР:

·      элементные ТПР - типовые решения по задаче или по отдельному виду обеспечения задачи (информационному, программному, техническому, математическому, организационному);

·      подсистемные ТПР - в качестве элементов типизации выступают отдельные подсистемы, разработанные с учетом функциональной полноты и минимизации внешних информационных связей;

·      объектные ТПР - типовые отраслевые проекты, которые включают полный набор функциональных и обеспечивающих подсистем АС.

Каждое типовое решение предполагает наличие, кроме собственно функциональных элементов (программных или аппаратных), документации с детальным описанием ТПР и процедур настройки в соответствии с требованиями разрабатываемой системы.

Основные особенности различных классов ТПР приведены в таблице 2.2.

Таблица 2.2.

Достоинства и недостатки ТПР

|  |  |  |
| --- | --- | --- |
| **Класс ТПР Реализация ТПР** | **Достоинства** | **Недостатки** |
| Элементные ТПР Библиотеки методо-ориентированных программ | ·        обеспечивается применение модульного подхода к проектированию и документированию АС | ·      большие затраты времени на сопряжение разнородных элементов вследствие информационной, программной и технической несовместимости  ·      большие затраты времени на доработку ТПР отдельных элементов |
| Подсистемные ТПР Пакеты прикладных программ | ·        достигается высокая степень интеграции элементов АС  ·        позволяют осуществлять: модульное проектирование; параметрическую настройку программных компонентов на различные объекты управления  ·        обеспечивают: сокращение затрат на проектирование и программирование взаимосвязанных компонентов; хорошее документирование отображаемых процессов обработки информации | ·      адаптивность ТПР недостаточна с позиции непрерывного инжиниринга деловых процессов  ·      возникают проблемы в комплексировании разных функциональных подсистем, особенно в случае использования решений нескольких производителей программного обеспечения |
| Объектные ТПР Отраслевые проекты АС | ·        комплексирование всех компонентов АС за счет методологического единства и информационной, программной и технической совместимости  ·        открытость архитектуры — позволяет устанавливать ТПР на разных программно-технических платформах  ·        масштабируемость — допускает конфигурацию АС для переменного числа рабочих мест  ·        конфигурируемость — позволяет выбирать необходимое подмножество компонентов | ·      проблемы привязки типового проекта к конкретному объекту управления, что вызывает в некоторых случаях даже необходимость изменения организационно-экономической структуры объекта автоматизации |

Для реализации типового проектирования используются два подхода: параметрически-ориентированное и модельно-ориентированное проектирование.

Параметрически-ориентированное проектирование включает следующие этапы: определение критериев оценки пригодности пакетов прикладных программ (ППП) для решения поставленных задач, анализ и оценка доступных ППП по сформулированным критериям, выбор и закупка наиболее подходящего пакета, настройка параметров (доработка) закупленного ППП.

Группы критериев оценки ППП представлены в таблице 2.3.

Таблица 2.3

Группы критериев оценки ППП

|  |  |
| --- | --- |
| № | Критерий |
| 1 | назначение и возможности пакета |
| 2 | отличительные признаки и свойства пакета |
| 3 | требования к техническим и программным средствам |
| 4 | документация пакета |
| 5 | факторы финансового порядка |
| 6 | особенности установки пакета |
| 7 | особенности эксплуатации пакета |
| 8 | помощь поставщика по внедрению и поддержанию пакета |
| 9 | оценка качества пакета и опыт его использования |
| 10 | перспективы развития пакета |

Внутри каждой группы критериев выделяется некоторое подмножество частных показателей, детализирующих каждый из десяти выделенных аспектов анализа выбираемых ППП.

Числовые значения показателей для конкретных ППП устанавливаются экспертами по выбранной шкале оценок (например, 10-бальной). На их основе формируются групповые оценки и комплексная оценка пакета (путем вычисления средневзвешенных значений). Нормированные взвешивающие коэффициенты также получаются экспертным путем.

Модельно-ориентированное проектирование заключается в адаптации состава и характеристик типовой АС в соответствии с моделью объекта автоматизации. Технология проектирования в этом случае должна обеспечивать единые средства для работы как с моделью типовой АС, так и с моделью конкретного предприятия.

Типовая АС в специальной базе метаинформации - репозитории - содержит модель объекта автоматизации, на основе которой осуществляется конфигурирование программного обеспечения. Таким образом, модельно-ориентированное проектирование АС предполагает, прежде всего, построение модели объекта автоматизации с использованием специального программного инструментария (например, SAP Business Engineering Workbench (BEW), BAAN Enterprise Modeler). Возможно также создание системы на базе типовой модели АС из репозитория, который поставляется вместе с программным продуктом и расширяется по мере накопления опыта проектирования АС для различных отраслей и типов производства.

Репозиторий содержит базовую (ссылочную) модель АС, типовые (референтные) модели определенных классов АС, модели конкретных АС предприятий.

Базовая модель АС в репозитории содержит описание бизнес-функций, бизнес-процессов, бизнес-объектов, бизнес-правил, организационной структуры, которые поддерживаются программными модулями типовой АС.

Типовые модели описывают конфигурации АС для определенных отраслей или типов производства.

Модель конкретного предприятия строится либо путем выбора фрагментов основной или типовой модели в соответствии со специфическими особенностями предприятия (BAAN Enterprise Modeler), либо путем автоматизированной адаптации этих моделей в результате экспертного опроса (SAP Business Engineering Workbench).

Построенная модель предприятия в виде метаописания хранится в репозитории и при необходимости может быть откорректирована. На основе этой модели автоматически осуществляется конфигурирование и настройка АС.

Бизнес-правила определяют условия корректности совместного применения различных компонентов АС и используются для поддержания целостности создаваемой системы. Модель бизнес-функций представляет собой иерархическую декомпозицию функциональной деятельности предприятия.

Модель бизнес-процессов отражает выполнение работ для функций самого нижнего уровня модели бизнес-функций. Для отображения процессов используется модель управления событиями (ЕРС - Event-driven Process Chain). Именно модель бизнес-процессов позволяет выполнить настройку программных модулей - приложений АС в соответствии с характерными особенностями конкретного предприятия.

Модели бизнес-объектов используются для интеграции приложений, поддерживающих исполнение различных бизнес-процессов.

Модель организационной структуры предприятия представляет собой традиционную иерархическую структуру подчинения подразделений и персонала.

Внедрение типовой АС начинается с анализа требований к конкретной АС, которые выявляются на основе результатов предпроектного обследования объекта автоматизации. Для оценки соответствия этим требованиям программных продуктов может использоваться описанная выше методика оценки ППП. После выбора программного продукта на базе имеющихся в нем референтных моделей строится предварительная модель АС, в которой отражаются все особенности реализации АС для конкретного предприятия. Предварительная модель является основой для выбора типовой модели системы и определения перечня компонентов, которые будут реализованы с использованием других программных средств или потребуют разработки с помощью имеющихся в составе типовой АС инструментальных средств (например, ABAP в SAP, Tools в BAAN).

Реализация типового проекта предусматривает выполнение операций, указанных в таблице 2.4.

Таблица 2.4

Операции для реализация типового проекта

|  |  |
| --- | --- |
| № | Операция |
| 1 | установку глобальных параметров системы |
| 2 | задание структуры объекта автоматизации |
| 3 | определение структуры основных данных |
| 4 | задание перечня реализуемых функций и процессов |
| 5 | описание интерфейсов |
| 6 | описание отчетов |
| 7 | настройка авторизации доступа |
| 8 | настройка системы архивирования |

Достоинства и недостатки метода перечислены в таблице 2.5.

Таблица 2.5.

Достоинства и недостатки метода типового проектирования

|  |  |
| --- | --- |
| Достоинства | ·      значительное снижение трудоемкости и сроков разработки и внедрения АС, если удалось найти подходящий типовой проект |
| Недостатки | ·      трудность применения метода из-за больших затрат на настройку проекта  ·      необходимость проведения структурных и организационных изменений на объекте с целью приведения в соответствие с требованиями типового проекта;  ·      сложность модернизации АС  ·      повышенные требования к квалификации разработчиков типовых проектов |

***2.1.4 Метод автоматизированного проектирования***

Метод ориентирован на использование формализованных моделей и методов в процессе создания АС и применение ЭВМ для реализации этих методов на всех стадиях создания АС. Автоматизация создания АС осуществляется не в результате перенесения в систему готовых проектных решений, а путем автоматизации проектных процедур (операций), а значит, как и в методе оригинального проектирования, можно учитывать специфику объекта.

Для реализации метода используются CASE-средства (см. ниже раздел 6).

Использование современных CASE-средств значительно изменяет структуру затрат по фазам ЖЦ АС по сравнению с традиционной разработкой (табл. 2.6) за счет ряда факторов, показанных в табл. 2.7. Достоинства и недостатки метода перечислены в таблице 2.8.

Таблица 2.6

Структура затрат по фазам ЖЦ АС

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Способ разработки | Анализ | Проектирование | Кодирование | Тестирование |
| Традиционная разработка | 20% | 15% | 20% | 45% |
| Использование CASE-технологий | 40% | 40% | 5% | 15% |

Таблица 2.7

Факторы изменения структуры ЖЦ

|  |  |  |
| --- | --- | --- |
| № | Традиционная разработка | CASE |
| 1 | Основные усилия - на кодирование и тестирование | Основные усилия – на анализ и проектирование |
| 2 | «Бумажные» спецификации | Быстрое итеративное прототипирование |
| 3 | Ручное кодирование | Автоматическая кодогенерация |
| 4 | Ручное документирование | Автоматическая генерация документации |
| 5 | Тестирование кодов | Автоматический контроль проекта |
| 6 | Сопровождение кодов | Сопровождение спецификаций проекта |

Таблица 2.8.

Достоинства и недостатки метода автоматизированного проектирования

|  |  |
| --- | --- |
| Достоинства | ·      высокая степень автоматизации проектирования на всех стадиях создания системы;  ·      возможность оперативной модернизации проектных решений в процессе развития объекта  ·      автоматизация основной части документооборота;  ·      значительное снижение трудоемкости создания системы  ·      высокая эффективность проектных решений |
| Недостатки | ·      дороговизна инструментальных средств |

***2.2 Регламентация процессов проектирования АС отечественными и международными стандартами***

***2.2.1 Реализуемые концептуальные подходы* *и общая характеристика стандартов проектирования АС***

**2.2.1.1 Cleanroom и RAD - концептуальныеподходы к организации процесса проектирования АС**

В целом стандарты процессов проектирования различаются по:

¾         конкретности и детализации содержащихся требований;

¾         открытости и гибкости, адаптируемости к конкретным условиям;

¾         степени обязательности для организаций разного типа;

¾         прикладной области.

При этом существующие стандарты можно разбить на две группы в зависимости от того, какой из двух принципиально различающихся концептуальных подходов к организации процессов проектирования они поддерживают: Cleanroom или RAD.

**Cleanroom Software Engineering** (концепция «чистой комнаты») — процесс разработки АС и их программного обеспечения, предназначенный для создания систем с сертифицируемым уровнем надёжности. Основная идея cleanroom состоит в том, что предупреждение дефектов лучше, чем их устранение.

Название *Cleanroom* («*чистая комната*») взято из электронной промышленности — так называются помещения с высокой степенью защиты от загрязнений, позволяющие предотвратить появление дефектов в процессе производства микросхем. Впервые процесс был применён во второй половине 80-х годов. Основные принципы Cleanroom приведены в табл. 2.9.

Таблица 2.9

Основные принципы Cleanroom

|  |  |
| --- | --- |
| № | Описание |
| 1. | Разработка программного обеспечения основывается на формальных методах |
| 2. | Инкрементальная реализация в рамках статистического контроля качества |
| 3. | Статистическое тестирование |
| 4. | Формальная верификация |

Методология предполагает детальное и тщательное выполнение всех проектных работ с большими затратами времени на отработку деталей и нюансов проекта, учитывающих все возможные ситуации, в том числе конфликты и большую нагрузку.

**RAD** (от англ. ***R****apid****A****pplication****D****evelopment* — быстрая разработка приложений) — концепция организации разработки, уделяющая особое внимание быстроте и удобству выполнения работ, созданию технологического процесса, позволяющего разработчику максимально быстро создавать системы и компьютерные программы. С конца XX века RAD получила широкое распространение и одобрение. Концепцию RAD также часто связывают с концепцией визуального программирования.

Основателем RAD считается сотрудник IBM Джеймс Мартин, который в 1980-х годах сформулировал основные принципы RAD, основываясь на идеях Барри Бойма и Скотта Шульца. Основные принципы RAD приведены в табл. 2.10

Таблица 2.10

Основные принципы RAD

|  |  |
| --- | --- |
| № | Принцип |
| 1. | Инструментарий должен быть нацелен на минимизацию времени разработки |
| 2. | Создание *прототипа* для уточнения требований заказчика |
| 3. | Цикличность разработки: каждая новая версия продукта основывается на оценке  заказчиком результата работы предыдущей версии |
| 4. | Минимизация времени разработки версии, за счёт переноса уже готовых модулей и добавления функциональности в новую версию |
| 5 | Команда разработчиков должна тесно сотрудничать, каждый участник должен быть готов выполнять несколько обязанностей |
| 6 | Управление проектом должно минимизировать длительность цикла разработки |

**2.2.1.2 Перечень и общая характеристика стандартов проектирования АС**

Основные отечественные и зарубежные стандарты, регламентирующие ЖЦ АС и их компонентов, а значит определяющие и процессы их проектирования, представлены в таблице 2.11.

Таблица 2.11

 Отечественные и зарубежные стандарты, регламентирующие ЖЦ АС

|  |  |
| --- | --- |
| Стандарт | Краткое описание |
| ГОСТ  34.601-90 | Комплекс стандартов на автоматизированные системы. Распространяется на АС и устанавливает стадии и этапы их создания. Кроме того, в стандарте содержится описание содержания работ на каждом этапе. Стадии и этапы работы, закрепленные в стандарте, в большей степени соответствуют каскадной модели жизненного цикла. |
| ISO/IEC  12207: 1995 | (Information technology - Software life cycle processes – Информационные технологии. Процессы жизненного цикла программного обеспечения) - Стандарт на процессы и организацию ЖЦ. Распространяется на все виды заказного ПО. Стандарт не содержит описания фаз, стадий и этапов. (ISO - International Organization of Standardization - Международная организация по стандартизации, IEC - International Electrotechnical Commission - Международная комиссия по электротехнике). |
| ISO/IEC  15288:2002 | (Systems engineering. System life cycle processes. – Системо-техника. Процессы жизненного цикла системы) - стандарт на процессы ЖЦ систем. Стандарт применим для широкого класса систем, но его основное предназначение - поддержка создания АС. |
| CDM | (Custom Development Method - методика Oracle по разработке прикладных информационных систем) - технологический материал, детализированный до уровня заготовок проектных документов, рассчитанных на использование в проектах с применением Oracle. Применяется для классической каскадной модели ЖЦ (предусмотрены все работы/задачи и этапы), а также для технологий "быстрой разработки" (Fast Track) или "облегченного подхода", рекомендуемых в случае малых проектов. |
| RUP | (Rational Unified Process)- предлагает итеративную модель разработки, включающую четыре фазы: начало, исследование, построение и внедрение. Каждая фаза может быть разбита на этапы (итерации), в результате которых выпускается версия для внутреннего или внешнего использования. Прохождение через четыре основные фазы называется циклом разработки, каждый цикл завершается генерацией версии системы. Если после этого работа над проектом не прекращается, то полученный продукт продолжает развиваться и снова минует те же фазы. Суть работы в рамках RUP - это создание и сопровождение моделей на базе языка UML, поддерживающего объектно-ориентированный подход к проектированию АС. |
| MSF | (Microsoft Solution Framework) - сходна с RUP, так же включает четыре фазы: анализ, проектирование, разработка, стабилизация; является итерационной, предполагает использование объектно-ориентированного моделирования. MSF в сравнении с RUP в большей степени ориентирована на разработку бизнес-приложений. |
| DATARUN | Стандарт организации проектных работ с ориентацией на структурно-аналитическую электронную методологию системного проектирования, т.е на архитектуру готовых программных средств. |
| Agile | (От англ. *Agile software development*)  - Гибкая методология разработки. Это концептуальный каркас, в рамках которого выполняется разработка АС и их ПО. Представляет собой семейство процессов разработки, а не единственный подход. Определяется международным соглашением Agile Manifesto, которое разработано и принято 11-13 февраля 2001 года |

Отметим, что серия стандартов ГОСТ 34.601-90 считается некоторыми разработчиками АС бюрократическими до вредности и консервативными до устарелости, поскольку весьма детально и жестко регламентируют структуру и содержание ЖЦ и проектной документации АС. Тем не менее, по тем же причинам этот стандарт предпочитают многие заказчики, так как он позволяет очень четко и методически правильно организовывать работы и оформлять документы при проектировании АС.

Стандарты CDM, RUP, MSF и DATARUN являются корпоративными. Однако в силу глобальности соответствующих корпораций и распространенности их продуктов эти стандарты можно считать международными де-факто.

Перечисленные стандарты в основном ориентированы на реализацию методологии Cleanroom.

Семейство методологий Agile  также не регламентируется официальными международными стандартами. Тем не менее, эти методологии можно условно считать, стандартами, так как они признаны большим количеством специалистов в области системной индустрии и постоянно совершенствуются, а сфера их применения неуклонно расширяется. Данное семейство методологий основано на принципах RAD.

Выбор стандарта для реализации того или иного проекта АС является неформальной задачей и зависит от конкретной ситуации на объекте заказчика, а также квалификации и предпочтений разработчика.

***2.2.2  Основы канонического проектирования согласно ГОСТ 34.601-90***

***2.2.2.1  Перечень и состав стадий создания АС согласно отечественным стандартам***

Организационные процессы создания и развития АС подразделяются на стадии (с этапами внутри стадии и работами внутри этапа) и очереди.

***Стадии*** определяют получение промежуточных результатов, когда результаты работы на предыдущих стадиях являются исходным материалом для последующих, и лишь стадия ввода системы в действие дает конечный результат.

Стадия – основная структурная единица процесса создания АС для планирования и контроля хода работ посредством календарных, сетевых графиков и других средств управления работой.

***Очередью АС*** является относительно самостоятельная часть системы, пригодная для эксплуатации независимо от разработки и ввода в действие последующих очередей. Очередями система как бы наращивается, но эксплуатируется уже после первой очереди.

Начальным условием и результатом каждой работы (этапа, стадии, очереди) является документация АС. Состав, объем, содержание документации должны быть необходимы и достаточны для фиксации и последующего использования результатов проектных работ. Оформление документации должно обеспечивать доступное усвоение и однозначное понимание содержащейся в ней информации.

Таким образом, любая проектная информационная работа находится в единстве с двумя документами:

-        в начале каждой работы – документ, обусловливающий работу;

-        в конце – документ, обусловливающий результат.

Без этих документов процесс создания АС не может быть объектом фиксации, а, следовательно, организации, планирования и отслеживания.

Стадии и этапы работы описаны в стандарте ГОСТ 34.601-90: «Автоматизированная система. Стадии создания». Общие требования и совокупности работ на всех стадиях и этапах регламентируется ГОСТ 24.602-86 «Состав и содержание работ по стадиям создания».

В зависимости от сложности объекта автоматизации и набора задач, требующих решения при создании конкретной АС, стадии и этапы работ могут иметь различную трудоемкость. Допускается объединять последовательные этапы и даже исключать некоторые из них на любой стадии проекта. Допускается также начинать выполнение работ следующей стадии до окончания предыдущей.

Конкретное содержание работ зависит от вида объекта управления и регламентируется ОРММ (Общеотраслевые руководящие методические материалы) для соответствующей отрасли.

Организация канонического проектирования АС ориентирована на использование главным образом каскадной модели ЖЦ АС и включает стадии и этапы работ, представленные в таблице 2.12. В самом общем случае процесс создания АС может включать восемь стадий: При этом выделяются три предпроектные стадии («Формирование требований к АС»; «Разработка концепции АС»; «Техническое задание на АС»), три стадии разработки («Эскизный проект»; «Технический проект»; «Рабочая документация») и две заключительные стадии («Ввод в действие»; «Сопровождение»).

Таблица 2.12

Стадии и этапы канонического проектирования АС

|  |  |
| --- | --- |
| Стадии | Этапы работ |
| Стадия 1.  **Формирование требований к АС** | ·     *обследование* объекта и обоснование необходимости создания АС;  ·     формирование требований пользователей к АС;  ·     оформление отчета о выполненной работе и тактико-*технического задания* на разработку |
| Стадия 2.  **Разработка концепции АС** | ·     изучение объекта автоматизации;  ·     проведение необходимых научно-исследовательских работ;  ·     разработка вариантов концепции АС, удовлетворяющих требованиям пользователей;  ·     оформление отчета и утверждение концепции. |
| Стадия 3.  **Техническое задание** | ·     разработка и утверждение *технического задания* на создание АС. |
| Стадия 4.  **Эскизный проект** | ·     разработка предварительных проектных решений по системе и ее частям;  ·     разработка эскизной документации на АС и ее части. |
| Стадия 5.  **Технический проект** | ·     разработка проектных решений по системе и ее частям;  ·     разработка документации на АС и ее части;  ·     разработка и оформление документации на поставку комплектующих изделий;  ·     разработка заданий на проектирование в смежных частях проекта. |
| Стадия 6. **Рабочая документация** | ·     разработка *рабочей документации* на АС и ее части;  ·     разработка и адаптация программ. |
| Стадия 7.  **Ввод в действие** | ·   подготовка объекта автоматизации;  ·   подготовка персонала;  ·   комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями);  ·   строительно-монтажные работы;  ·   пусконаладочные работы;  ·   проведение *предварительных испытаний* ;  ·   проведение *опытной эксплуатации* ;  ·   проведение *приемочных испытаний*. |
| Стадия 8.  **Сопровождение АС** | ·   выполнение работ в соответствии с гарантийными обязательствами;  ·   послегарантийное обслуживание |

Конкретный состав, последовательность и сроки реализации стадий и этапов работ устанавливаются в каждом случае в техническом задании и договорах на выполнение работ. При этом обязательными являются стадии технического задания, технического проекта, рабочей документации и ввода в действие. Для относительно простых систем и систем, разрабатываемых на основе типовых проектных решений, стадии технического проекта и рабочей документации объединяют в стадию техно-рабочего проектирования.

***2.2.2.2  Виды и структура документации на АС***

Вся документация, формируемая в процессе создания АС, по характеру может быть разделена на четыре вида, указанные в таблице 2.13.

Таблица 2.13

Виды документации на АС

|  |  |
| --- | --- |
| № | Вид документации |
| 1 | Предпроектная документация, формируемая на предпроектных стадиях и обязательно содержащая техническое задание на создание системы со всеми приложениями к нему |
| 2 | Техническая документация, формируемая на стадии проектирования системы |
| 3 | Проектно-сметная документация архитектурно-строительной и смежной частей проекта АС, включая проект монтажа и наладки комплекса технических средств (КТС) системы |
| 4 | Организационно-распорядительная документация, определяющая организационно-правовые и хозяйственные взаимоотношения участников создания АС |

Общая структура технической документации представлена на рис. 2.2.

Наиболее общие требования к документации состоят в том, чтобы ее объем и детализация обеспечивали возможность понимания и оценки предпроектных предложений и проектных решений, изучения предлагаемых проектом методов управления, выполнения строительно-монтажных и пусконаладочных работ, освоения и эксплуатации созданной системы, координированного взаимодействия участников работ в процессе создания системы.

***2.2.2.3  Документ «Технико-экономическое обоснование создания АС»***

Этот документ формируется по итогам работ по обследованию объекта автоматизации и определению стратегии внедрения АС на первой предпроектной стадии**«**Формирование требований к АС». Ориентировочное содержание тематики этого документа укрупненно представлено в таблице 2.14.

Таблица 2.14.

Содержание технико-экономического обоснования

|  |  |
| --- | --- |
| № темы | Содержимое темы |
| 1 | ограничения, риски, критические факторы, которые могут повлиять на успешность проекта |
| 2 | совокупность условий, при которых предполагается эксплуатировать будущую систему: архитектура системы, аппаратные и программные ресурсы, условия функционирования, обслуживающий персонал и пользователи системы |
| 3 | сроки завершения отдельных этапов, форма приемки/сдачи работ, привлекаемые ресурсы, меры по защите информации |
| 4 | описание выполняемых системой функций |
| 5 | возможности развития системы |
| 6 | информационные объекты системы |
| 7 | интерфейсы и распределение функций между человеком и системой |
| 8 | требования к программным и информационным компонентам ПО, требования к СУБД |
| 9 | что не будет реализовано в рамках проекта |

В нем должно быть четко сформулировано, что получит заказчик, если согласится финансировать проект, когда он получит готовый продукт (график выполнения работ) и сколько это будет стоить (для крупных проектов должен быть составлен график финансирования на разных этапах работ). В документе желательно отразить не только затраты, но и выгоду проекта, например время окупаемости проекта, ожидаемый экономический эффект (если его удается оценить).

***2.2.2.4  Документ «Техническое задание на создание АС»***

***Техническое задание (ТЗ)*** - это документ, определяющий цели, требования и основные исходные данные, необходимые для разработки АС (табл. 2.15).

Таблица 2.15.

Состав и содержание технического задания (ГОСТ 34.602- 89)

|  |  |  |
| --- | --- | --- |
| **№** | **Раздел** | **Содержание** |
| 1 | Общие сведения | ·     полное наименование системы и ее условное обозначение  ·     шифр темы или шифр (номер) договора;  ·     наименование предприятий разработчика и заказчика системы, их реквизиты  ·     перечень документов, на основании которых создается АС  ·     плановые сроки начала и окончания работ  ·     сведения об источниках и порядке финансирования работ  ·     порядок оформления и предъявления заказчику результа-тов работ по созданию системы, ее частей и отдельных средств |
| 2 | Назначение и цели создания (развития) системы | ·     вид автоматизируемой деятельности  ·     перечень объектов, на которых предполагается использование системы  ·     наименования и требуемые значения технических, технологических, производственно-экономических и др. показателей объекта, которые должны быть достигнуты при внедрении АС |
| 3 | Характеристика объектов автоматизации | ·     краткие сведения об объекте автоматизации  ·     сведения об условиях эксплуатации и характеристиках окружающей среды |
| 4 | Требования к системе | Требования к системе в целом:  ·     требования к структуре и функционированию системы (перечень подсистем, уровни иерархии, степень централизации, способы информационного обмена, режимы функционирования, взаимодействие со смежными системами, перспективы развития системы)  ·     требования к персоналу (численность пользователей, квалификация, режим работы, порядок подготовки)  ·     показатели назначения (степень приспособляемости системы к изменениям процессов управления и значений параметров)  ·     требования к надежности, безопасности, эргономике, транспортабельности, эксплуатации, техническому обслуживанию и ремонту, защите и сохранности информации, защите от внешних воздействий, к патентной чистоте, по стандартизации и унификации  Требования к функциям (по подсистемам) :  ·     перечень подлежащих автоматизации задач  ·     временной регламент реализации каждой функции  ·             требования к качеству реализации каждой функции, к форме представления выходной информации, характеристики точности, достоверности выдачи результатов  ·     перечень и критерии отказов  Требования к видам обеспечения:  ·         математическому (состав и область применения матем. моделей и методов, типовых и разрабатываемых алгоритмов)  ·     информационному (состав, структура и организация данных, обмен данными между компонентами системы, информационная совместимость со смежными системами, используемые классификаторы, СУБД, контроль данных и ведение информационных массивов, процедуры придания юридической силы выходным документам)  ·     лингвистическому (языки программирования, языки взаимодействия пользователей с системой, системы кодирования, языки ввода - вывода)  ·     программному (независимость программных средств от платформы, качество программных средств и способы его контроля, использование фондов алгоритмов и программ)  ·     техническому  ·     метрологическому  ·     организационному (структура и функции эксплуатирующих подразделений, защита от ошибочных действий персонала)  ·     методическому (состав нормативно-технической документации) |
| 5 | Состав и содержание работ по созданию системы | ·     перечень стадий и этапов работ  ·     сроки исполнения  ·     состав организаций — исполнителей работ  ·     вид и порядок экспертизы технической документации  ·     программа обеспечения надежности  ·     программа метрологического обеспечения |
| 6 | Порядок контроля и приемки системы | ·     виды, состав, объем и методы испытаний системы  ·     общие требования к приемке работ по стадиям  ·     статус приемной комиссии |
| 7 | Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие | ·     преобразование входной информации к машиночитаемому виду  ·     изменения в объекте автоматизации  ·     сроки и порядок комплектования и обучения персонала |
| 8 | Требования к документированию | ·     перечень подлежащих разработке документов  ·     перечень документов на машинных носителях |
| 9 | Источники разработки | §  документы и информационные материалы, на основании которых разрабатывается ТЗ и система |

***2.2.2.5  Состав комплексов документов стадий проектирования АС***

Эскизный проект (аван-проект, генеральная схема) предусматривает разработку предварительных проектных решений по системе и ее частям.

Выполнение стадии эскизного проектирования не является строго обязательной. Если основные проектные решения определены ранее или достаточно очевидны для конкретной АС и объекта автоматизации, то эта стадия может быть исключена из общей последовательности работ.

Содержание эскизного проекта задается в ТЗ на систему и, как правило, определяет характеристики АС, перечисленные в таблице 2.16.

Таблица 2.16.

Содержание эскизного проекта

|  |  |
| --- | --- |
| № | Определяемая характеристика |
| 1 | Функции АС |
| 2 | Функции подсистем, их цели и ожидаемый эффект от внедрения |
| 3 | Состав комплексов задач и отдельных задач |
| 4 | Концепция информационной базы и ее укрупненная структура |
| 5 | Функции СУБД |
| 6 | Состав вычислительной системы и других технических средств |
| 7 | Функции и параметры основных программных средств |

На основе технического задания (и эскизного проекта) разрабатывается технический проект АС. В техническом проекте в обязательном порядке должны быть раскрыты вопросы, указанные в таблице 2.17.

Таблица 2.17.

Вопросы, раскрываемые в техническом проекте

|  |  |
| --- | --- |
| № | Определяемая характеристика |
| 1 | Обоснование принятых проектных решений по АС в целом, в т.ч. в части совместимости локальных подсистем в составе АС |
| 2 | Обоснование выбора задач, описания постановки задач и алгоритмов их реализации |
| 3 | Описание организационной и функциональной структуры АС |
| 4 | Описание решений по информационному обеспечению АС |
| 5 | Описание решений по техническому обеспечению АС |
| 6 | Описание основных решений по программному обеспечению АС |
| 7 | Уточненный расчет ожидаемой экономической эффективности |

Более детализированный состав разделов и содержание технического проекта приведены в табл. 2 18.

Таблица 2.18

Содержание технического проекта

|  |  |  |
| --- | --- | --- |
| **№** | **Раздел** | **Содержание** |
| 1 | Пояснительная записка | * основания для разработки системы * перечень организаций разработчиков * краткая характеристика объекта с указанием основных технико-экономических показателей его функционирования и связей с другими объектами * краткие сведения об основных проектных решениях по функциональной и обеспечивающим частям системы |
| 2 | Функциональная и организационная структура системы | * обоснование выделяемых подсистем, их перечень и назначение * перечень задач, решаемых в каждой подсистеме, с краткой характеристикой их содержания * схема информационных связей между подсистемами и между задачами в рамках каждой подсистемы |
| 3 | Постановка задач и алгоритмы решения | * организационно-экономическая сущность задачи (наименование, цель решения, краткое содержа-ние, метод, периодичность и время решения зада-чи, способы сбора и передачи данных, связь задачи с другими задачами, характер использования результатов решения, в которых они используются) * экономико-математическая модель задачи (структурная и развернутая форма представления) * входная оперативная информация ( характеристика показателей, диапазон изменения, формы представления) * нормативно-справочная информация ( НСИ) (содержание и формы представления) * информация, хранимая для связи с другими задачами * информация, накапливаемая для последующих решений данной задачи * информация по внесению изменений ( система внесения изменений и перечень информации, подвергающейся изменениям) * алгоритм решения задачи ( последовательность этапов расчета, схема, расчетные формулы) * контрольный пример (набор заполненных данными форм входных документов, условные документы с накапливаемой и хранимой информацией, формы выходных документов, заполненные по результатам решения экономико-технической задачи и в соответствии с разработанным алгоритмом расчета) |
| 4 | Организация информационной базы | * источники поступления информации и способы ее передачи * совокупность показателей, используемых в системе * состав документов, сроки и периодичность их поступления * основные проектные решения по организации фонда НСИ * состав НСИ, включая перечень реквизитов, их определение, диапазон изменения и перечень документов НСИ * перечень массивов НСИ, их объем, порядок и частота корректировки информации * структура фонда НСИ с описанием связи между его элементами; требования к технологии создания и ведения фонда * методы хранения, поиска, внесения изменений и контроля * определение объемов и потоков информации НСИ * контрольный пример по внесению изменений в НСИ * предложения по унификации документации |
| 5 | Альбом форм документов |  |
| 6 | Система математического обеспечения | * обоснование структуры математического обеспечения * обоснование выбора системы программирования * перечень стандартных программ |
| 7 | Принцип построения комплекса технических средств | * описание и обоснование схемы технологического процесса обработки данных * обоснование и выбор структуры комплекса технических средств и его функциональных групп * обоснование требований к разработке нестандартного оборудования * комплекс мероприятий по обеспечению надежности функционирования технических средств |
| 8 | Расчет экономической эффективности системы | * сводная смета затрат, связанных с эксплуатацией систем * расчет годовой экономической эффективности, источниками которой являются оптимизация производственной структуры хозяйства (объединения), снижение себестоимости продукции за счет рационального использования производственных ресурсов и уменьшения потерь, улучшения принимаемых управленческих решений |
| 9 | Мероприятия по подготовке объекта к внедрению системы | * перечень организационных мероприятий по совершенствованию бизнес-процессов * перечень работ по внедрению системы, которые необходимо выполнить на стадии рабочего проектирования, с указанием сроков и ответственных лиц |
| 10 | Ведомость документов |  |

На стадии «Рабочая документация» осуществляется создание программного продукта и разработка всей сопровождающей документации.

Рабочая документация включает разделы, указанные в таблице 2.19.

Таблица 2.19

Разделы рабочей документации

|  |  |
| --- | --- |
| № | Определяемая характеристика |
| 1 | Общее описание автоматизированной системы и ее компонентов |
| 2 | Описание решений, обеспечивающих сопровождение, изготовление и эксплуатацию программ |
| 3 | Инструктивные материалы по эксплуатации оборудования и выполнению операций технологических процессов обработки данных |
| 4 | Схемы монтажа и наладки КТС, включая чертежи и схемы общего вида щитов, пультов, панелей, соединений, устройств контроля, сигнализации и других элементов системы, связанных с аппаратным управлением технологией обработки данных |
| 5 | Расчет экономической эффективности |

***2.2.2.6  Разновидности и общая характеристика схем в документации АС***

Схемы используются для отображения различных видов задач автоматизации и средств их решения.

***Схема*** – графическое представление определения, анализа или метода решения задачи, в котором используются символы для отображения операций, данных, потоков, оборудования и т.д.

Применяемые на практике разновидности схем приведены в таблице 2.20.

Таблица 2.20

Виды схем документации АС

|  |  |
| --- | --- |
| Вид | Определение |
| Схемы данных | отображают путь данных при решении задачи и определяют этапы обработки, а также применяемые носители данных. Схема должна начинаться и заканчиваться символами данных; символы данных предшествуют и следуют за символами процесса |
| Схемы программ | отображают последовательность операций в программе |
| Схемы работы системы | отображают управление операциями и поток данных в системе |
| Схемы взаимодействия программ | отображают путь активации программ и взаимодействие с соответствующими данными (каждая программа в схеме показывается только один раз в отличие от схемы работы системы, где она может показываться более, чем в одном потоке управления) |
| Схемы ресурсов системы | отображают конфигурацию блоков данных и обрабатывающих блоков, требуемую для решения задачи или комплекса задач |

Множество применяемых в схемах символов с точки зрения функциональной специфики делят на следующие подмножества*:*символы данных; символы процесса; символы линий; специальные символы.

В рамках этих подмножеств выделяют основные и специфические символы. Первые используются в тех случаях, когда точный тип процесса или носителя данных несущественен или отсутствует необходимость в описании фактического носителя данных. Вторые - в противном случае.

Описание и применение символов в схемах представлено в табл. 2.21.

Таблица 2.21

Описание и применение символов в схемах

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Символы* | *Наименование символов* | *Описание отображаемого объекта* | *Применение в схемах* | | | | |
| *1* | *2* | *3* | *4* | *5* |
| **Символы данных** | | | | | | | |
| **Основные** |  | | | | | | |
|  | Данные | Данные, носитель которых не определен. | + | + | + | + | + |
|  | Запоминаемые данные | Хранимые данные, имеющие пригодный для обработки вид, носитель которых не определен | + | - | + | + | + |
| **Специфические** |  | | | | | | |
|  | ОЗУ | Данные, хранящиеся в ОЗУ | + | - | + | + | + |
|  | ЗУ с последовательным доступом | Магнитная лента, магнитофонная кассета | + | - | + | + | + |
|  | ЗУ с прямым доступом | Магнитный диск, магнитный барабан, ГМД | + | - | + | + | + |
|  | Документ | Данные представленные на носителе в удобочитаемой форме (машинограмота, документ для оптического или магнитного считывания, микрофильм, рулон ленты с итоговыми данными, бланки ввода данных и т.п.) | + | - | + | + | + |
|  | Ручной ввод | Данные вводимые вручную во время обработки с устройств любого типа (клавиатура, переключатели, кнопки, световое перо, полоски со штрих-кодом и т.д.) | + | - | + | + | + |
|  | Карта | Данные на носителях в виде карты (перфокарты, магнитные карты, карты со считываемыми метками (ярлыками) и т.д.) | + | - | + | + | + |
|  | Лента бумажная | Данные на носителе в виде бумажной ленты | + | - | + | + | + |
|  | Дисплей | Данные, представленные в человекочитаемой форме на носителе в виде отображающего устройства (экран, индикатор ввода информации) | + | - | + | + | + |
| **Символы процесса** | | | | | | | |
| **Основные** |  | | | | | | |
|  | Процесс | Функция обработки данных любого вида | + | + | + | + | + |
| **Специфические** |  | | | | | | |
|  | Предопределенный процесс | Процесс, состоящий из одной или нескольких операций, или шагов программы, которые определены в другом месте | - | + | + | + | - |
|  | Ручная операция | Любой процесс, выполняемый человеком | + | - | + | + | - |
|  | Подготовка | Модификация команды или группы команд с целью воздействия на некоторую последовательную функцию (установка переключателей, модификация индексного регистра или инициализация программы). | + | + | + | + | - |
|  | Решение | Решение или функции переключающего типа, имеющие один вход и ряд альтернативных выходов, один и только один из которых может быть активирован после выполнения условий, описанных внутри этого символа | - | + | + | - | - |
|  | Параллельные действия | Синхронизация двух или более параллельных операций  Пример:   |  | | --- | |  | |  |  |                       Процессы C,D,E не могут начинаться, пока не завершится процесс A. Процесс F должен ожидать завершения процессов B,C,D. Но процесс C может начинаться и завершаться прежде, чем начнется и завершится процесс D. | - | + | + | + | - |
|  | Граница цикла | Начало и конец цикла. Обе части могут содержать один и тот же идентификатор. Условие начала, приращения, завершения и т.д. помещается внутри, в начале или конце, в зависимости от проверяющей операции | - | + | + | - | - |
| **Символы линий** | | | | | | | |
| **Основные** |  | | | | | | |
|  | Линия | Поток данных или управление (для удобочитаемости могут быть добавлены стрелки-указатели) | + | + | + | + | + |
| **Специфические** |  | | | | | | |
|  | Передача управления | Непосредственная передача управления от одного процесса к другому, иногда с возможностью прямого возвращения к инициирующему процессу после завершения функций инициированного процесса. Тип передачи управления указывается внутри символа | - | - | - | + | - |
|  | Канал связи | Передача данных по каналу связи | + | - | + | + | + |
|  | Пунктирная линия | Альтернативная связь между двумя и более символами. Применяется также для обведения аннотируемого участка | + | + | + | + | + |
| **Специальные символы** | | | | | | | |
|  | Соединитель | Выход в часть схемы и вход из другой части этой схемы. Используется для обрыва линии и продолжения её в другом месте. Соответствующие символы должны иметь одно и тоже уникальное обозначение | + | + | + | + | + |
|  | Терминатор | Выход во внешнюю среду и вход из внешней среды (начало и конец программы, внешнее использование, источник данных) | + | + | + | - | - |
|  | Комментарий | Для добавления описательных комментариев или пояснительных записей | + | + | + | + | + |
|  | Пропуск | Пропуск символа (группы символов) в схемах, в которых не определён ни тип, ни число символов. Используется только в символах линий или между ними. Применяется обычно в схемах, изображающих общее решение с неизвестным числом повторений | + | + | + | + | + |

***2.2.2.7  Правила выполнения схем в документации АС***

Символы в схеме должны быть расположены равномерно при минимальном числе длинных соединительных линий. Символы должны быть по возможности одного размера без изменения углов и других параметров, влияющих на форму символа. Допустима любая ориентация, но предпочтительнее - горизонтальная.

Если пояснительный текст внутри символов не помещается, его выносят в комментарий сбоку или на отдельный лист, давая перекрестную ссылку.

В схемах можно использовать идентификатор символа (для удобства ссылки на него из других элементов документации). Идентификатор должен располагаться слева над символом в разрыве линий символа. Какое-либо дополнительное описание символа может быть расположено справа над символом.

Потоки данных или потоки управления в схемах показываются линией. Направление потока слева направо и сверху вниз считается стандартным и может не показываться стрелками. В остальных случаях стрелки указываются обязательно. Следует избегать пересечения линий. Недопустимо менять направление потока в точке пересечения. Две или более входящие линии могут объединяться в одну исходящую, причем места объединения должны быть смещены. Линии в схемах должны подходить к символу либо слева, любо сверху, а исходить - либо справа, либо снизу.

В случае разрыва линий соединитель в начале разрыва называется  внешним соединителем, а в конце - внутренним. При этом ссылки к странице могут быть приведены совместно с символом комментария для соответствующих соединителей, размещенных на разных страницах.

Несколько выходов из символа решения следует показывать:

-      Несколькими линиями из данного символа к другим;

-      Одной линией из данного символа, которая затем разветвляется в соответствующее число линий.

Каждый выход из символа должен сопровождаться соответствующими значениями условий, чтобы понять логический путь, который он представляет. Вместо одного символа с соответствующим текстом могут быть использованы несколько символов с перекрытием изображений, каждое из которых содержит описательный текст.

Когда несколько символов представляют упорядоченное множество, это упорядочение должно располагаться от переднего (первого) к заднему (последнему). Линии могут входить или исходить из любой точки перекрытых символов, не меняя при этом их приоритет или последовательный порядок.

Обычно считается, что первым выполняется действие с символом, расположенным на переднем плане.

***2.2.2.8  Правила оформления текстовых документов***

Правила оформления текстовых документов регламентирует ГОСТ 2.105-95, руководящий документ РД 50-34.698 и стандарт предприятия, для которого формируется документ (рис. 2.3).

*Общие требования*:

-        Титульный лист.

-        Нумерация арабскими цифрами через точку, при этом точка в конце не ставится (разделы: 1, 2, 3 и т.д.; подразделы - 1.1, 1.2 и т.д., пункты – 1.1.1, 1.1.2 и т.д.).

-        Заголовки разделов – прописными буквами, не подчеркивая, без выделения цветом; заголовки других уровней – с большой буквы строчными буквами.

-        Нумерация формул, рисунков, таблиц двойная через точку (номер раздела и номер в разделе).

-        Заголовки рисунков – над или под ними, а номер – под ними.

-        Номер таблицы – справа сверху.

-        Номер формулы записывается в скобках, справа от формулы.

-        Нумерация страниц сверху – по центру или справа.

-        Каждый раздел начинается с нового листа.

-        Подразделы и ниже могут начинаться на том же листе, что и предыдущий, если после заголовка на странице помещается не менее трех строк.

-        Поля на листе: сверху/снизу – 2 см, слева – 2,5 см, справа – 1,0 см. Размер шрифта - 12÷14.

*Требования к содержанию*:

ТИТУЛЬНЫЙ ЛИСТ

БЛАНК ЗАДАНИЯ

АННОТАЦИЯ

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1       СИСТЕМНО-АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1  Описание и анализ объекта

1.2  Постановка задачи

1.3  Обзор существующих аналогов

1.4  Концептуальная модель системы

1.5  Функциональная декомпозиция системы

2       ПРОЕКТНО-РАСЧЕТНАЯ ЧАСТЬ

2.1  Информационное обеспечение

*2.1.1  Внемашинное информационное обеспечение (формы документов)*

*2.1.2  Внутримашинная информационная база (описанием сущностей, их связей, логической и физической модели базы данных и т.д.)*

2.2  Математическое и алгоритмическое обеспечение

2.3  Выбор инструментальной платформы и комплекса технических средств (КТС)

3       РЕАЛИЗАЦИОННАЯ ЧАСТЬ

3.1  Программное обеспечение

*3.1.1  Структура программного обеспечения в целом*

*3.1.2  Описание программы на уровне исходного текста*

3.2  Организационное обеспечение

*3.2.1  Организация работы системы*

*3.2.2  Руководство пользователя*

*3.2.3  Описание результатов тестирования системы и ее апробации на контрольном примере*

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ

ЧЕРТЕЖ формата А1 (диаграммы и схемы, раскрывающие суть проектных решений)

Рис. 2.3  Требования кафедры ИТАС БГУИР к оформлению расчетно-пояснительной записки по курсовой работе по дисциплине ПАС

***2.2.3  Международный стандарт ISO/IEC 12207***

В соответствии с базовым международным стандартом ISO/IEC 12207 все процессы ЖЦ ПО АС делятся на три группы: основные, вспомогательные и организационные. Их состав приведен в таблице 2.22.

Таблица 2.22

Группы процессов ЖЦ АС согласно ISO/IEC 12207

|  |  |
| --- | --- |
| Группа процессов | Наименования процессов |
| **1. Основные процессы** | o         приобретение;  o         поставка;  o         разработка;  o         эксплуатация;  o         сопровождение. |
| **2. Вспомогательные процессы**: | o   документирование;  o   управление конфигурацией;  o   обеспечение качества;  o   разрешение проблем;  o   аудит;  o   аттестация;  o   совместная оценка;  o   верификация. |
| **3. Организационные процессы**: | o   создание инфраструктуры;  o   управление;  o   обучение;  o   усовершенствование. |

В таблице 2.23 приведены укрупненные описания основных процессов ЖЦ. АС согласно стандарту ISO/IEC 12207.

     Таблица 2.23

Содержание основных процессов ЖЦ АС согласно ISO/IEC 12207

|  |  |  |  |
| --- | --- | --- | --- |
| **Процесс (исполнитель)** | **Действия** | **Вход** | **Результат** |
| Приобретение (заказчик) | · Инициирование  · Подготовка заявочных предложений  · Подготовка договора  · Контроль деятельности поставщика  · Приемка АС | · Решение о начале работ по внедрению АС  · Результаты обследо-вания деятельности заказчика  · Результаты анализа рынка АС/ тендера  · План поставки/ разработки  · Комплексный тест АС | · Технико-экономическое обоснование внедрения АС  · Техническое задание на АС  · Договор на поставку/ разработку  · Акты приемки этапов работы  · Акт приемно-сдаточных испытаний |
| Поставка (разработчик АС) | · Инициирование  · Ответ на заявочные предложения  · Подготовка договора  · Планирование исполнения  · Поставка АС | · Техническое задание на АС  · Решение руковод-ства об участии в разработке  · Результаты тендера  · Техническое задание на АС  · План управления проектом  · Разработанная АС и документация | · Решение об участии в разработке  · Коммерческие предложения/ конкурсная заявка  · Договор на поставку/ разработку  · План управления проектом  · Реализация/ корректировка  · Акт приемно-сдаточных испытаний |
| Разработка (разработчик АС) | · Подготовка  · Анализ требований к АС  · Проектирование архитектуры АС  · Разработка требований к ПО  · Проектирование архитектуры ПО  · Детальное проектирование ПО  · Кодирование и тестирование ПО  · Интеграция ПО и квалификационное тестирование ПО  · Интеграция АС и квалификационное тестирование АС | · Техническое задание на АС  · Техническое задание на АС, модель ЖЦ  · Подсистемы АС  · Спецификации требования к компонентам ПО  · Архитектура ПО  · Материалы детального проектирования ПО  · План интеграции ПО, тесты  · Архитектура АС, ПО, документация на АС, тесты | · Используемая модель ЖЦ, стандарты разработки  · План работ  · Состав подсистем, компоненты оборудования  · Спецификации требований к компонентам ПО  · Состав компонентов ПО, интерфейсы с БД, план интеграции ПО  · Проект БД, спецификации интерфейсов между компонентами ПО, требования к тестам  · Тексты модулей ПО, акты автономного тестирования  · Оценка соответствия ком-плекса ПО требованиям ТЗ  · Оценка соответствия ПО, БД, технического комплекса и комплекта документации требованиям ТЗ |

Вспомогательные процессы предназначены для поддержки выполнения основных процессов, обеспечения качества проекта, организации верификации, проверки и тестирования ПО. Организационные процессы определяют действия и задачи, выполняемые как заказчиком, так и разработчиком проекта для управления своими процессами.

Стандарт не содержит описания фаз, стадий и этапов, т.е. подразумевает, что разработчик самостоятельно принимает решение о компоновке и взаимной увязке проектных работ.

Для поддержки практического применения стандарта ISO/IEC 12207 разработан ряд технологических документов: Руководство для ISO/IEC 12207 (ISO/IEC TR 15271:1998 Information technology - Guide for ISO/IEC 12207) и Руководство по применению ISO/IEC 12207 к управлению проектами (ISO/IEC TR 16326:1999 Software engineering - Guide for the application of ISO/IEC 12207 to project management).

***2.2.4  Международный стандарт ISO/IEC 15288***

Позднее был разработан и в 2002 г. опубликован стандарт на процессы жизненного цикла систем (ISO/IEC 15288 System life cycle processes). К разработке стандарта были привлечены специалисты различных областей: системной инженерии, программирования, управления качеством, человеческими ресурсами, безопасностью и пр. Был учтен практический опыт создания систем в правительственных, коммерческих, военных и академических организациях. Стандарт применим для широкого класса систем, но его основное предназначение - поддержка создания компьютеризированных систем.

Согласно стандарту ISO/IEC серии 15288 в структуру ЖЦ следует включать следующие группы процессов, перечисленные в табл. 2.24.

Таблица 2.24

Группы процессов ЖЦ АС согласно ISO/IEC серии 15288

|  |  |
| --- | --- |
| **Группа процессов** | Наименования процессов |
| **1. Договорные процессы** | o       приобретение (внутренние решения или решения внешнего поставщика);  o   поставка (внутренние решения или решения внешнего поставщика) |
| **2. Процессы предприятия**: | o   управление окружающей средой предприятия;  o   инвестиционное управление;  o   управление ЖЦ АС;  o   управление ресурсами;  o   управление качеством |
| **3. Проектные процессы**: | o   планирование проекта;  o   оценка проекта;  o   контроль проекта;  o   управление рисками;  o   управление конфигурацией;  o   управление информационными потоками;  o   принятие решений |
| **4. Технические процессы** | o   определение требований;  o   анализ требований;  o   разработка архитектуры;  o   внедрение;  o   интеграция;  o   верификация;  o   переход;  o   аттестация;  o   эксплуатация;  o   сопровождение;  o   утилизация. |
| **5. Специальные процессы** | o   определение и установка взаимосвязей исходя из задач и целей. |

Стадии создания системы, предусмотренные в стандарте ISO/IEC 15288, несколько отличаются от рассмотренных выше. Перечень стадий и основные результаты, которые должны быть достигнуты к моменту их завершения, приведены в табл. 2.25.

Таблица 2.25.

Стадии создания систем (ISO/IEC 15288)

|  |  |  |
| --- | --- | --- |
| **№** | **Стадия** | **Описание** |
| 1 | Формирование концепции | Анализ потребностей, выбор концепции и проектных решений |
| 2 | Разработка | Проектирование системы |
| 3 | Реализация | Изготовление системы |
| 4 | Эксплуатация | Ввод в эксплуатацию и использование системы |
| 5 | Поддержка | Обеспечение функционирования системы |
| 6 | Снятие с эксплуатации | Прекращение использования, демонтаж, архивирование системы |

***2.2.5  Корпоративный стандарт CDM***

CDM (Custom Development Method - методика Oracle по разработке прикладных информационных систем) - технологический материал, детализированный до уровня заготовок проектных документов, рассчитанных на использование в проектах с применением Oracle. Применяется для классической каскадной модели ЖЦ (предусмотрены все работы/задачи и этапы), а также для технологий "быстрой разработки" (Fast Track) или "облегченного подхода", рекомендуемых в случае малых проектов.

***2.2.6  Корпоративный стандарт RUP***

RUP(Rational Unified Process) — методология разработки АС и их программного обеспечения, созданная компанией Rational Software. В табл. 2.26 представлены принципы, лежащие в основе RUP, а на рис. 2.4 – схема RUP.

Таблица 2.26

Принципы RUP

|  |  |
| --- | --- |
| № | Описание |
| 1 | Ранняя идентификация и непрерывное (до окончания проекта) устранение основных рисков |
| 2 | Концентрация на выполнении требований заказчиков к исполняемой программе (анализ и построение модели прецедентов (вариантов использования)) |
| 3 | Ожидание изменений в требованиях, проектных решениях и реализации в процессе разработки |
| 4 | Компонентная архитектура, реализуемая и тестируемая на ранних стадиях проекта |
| 5 | Постоянное обеспечение качества на всех этапах разработки проекта (продукта) |
| 6 | Работа над проектом в сплочённой команде, ключевая роль в которой принадлежит архитекторам |

RUP использует итеративную модель разработки. В конце каждой итерации (в идеале продолжающейся от 2 до 6 недель) проектная команда должна достичь запланированных на данную итерацию целей, создать или доработать проектные артефакты и получить промежуточную, но функциональную версию конечного продукта. Итеративная разработка позволяет быстро реагировать на меняющиеся требования, обнаруживать и устранять риски на ранних стадиях проекта, а также эффективно контролировать качество создаваемого продукта.

Рис. 2.4 Графическое представление процесса разработки по RUP

Полный жизненный цикл разработки продукта состоит из четырех фаз, каждая из которых включает в себя одну или несколько итераций. Схема, приведенная на рисунке 2.4, позволяет получить графическое представление процесса разработки по RUP. Краткая характеристика фаз RUP дана в табл. 2.27.

Таблица 2.27

Краткая характеристика фаз процесса разработки по RUP

|  |  |
| --- | --- |
| Фаза | Описание |
| 1. Начало (Inception) | В фазе «Начало»:   * Формируются видение и границы проекта * Создается экономическое обоснование (business case) * Определяются основные требования, ограничения и ключевая функциональность продукта * Создается базовая версия модели прецедентов * Оцениваются риски   При завершении фазы оценивается достижение вехи целей жизненного цикла (англ. Lifecycle Objective Milestone), которое предполагает соглашение заинтересованных сторон о продолжении проекта. |
| 2. Уточнение (Elaboration) | В фазе «Уточнение» производится анализ предметной области и построение исполняемой архитектуры. Это включает в себя:   * Документирование требований (включая детальное описание для большинства прецедентов). * Спроектированную, реализованную и оттестированную исполняемую архитектуру. * Обновленное экономическое обоснование и более точные оценки сроков и стоимости. * Сниженные основные риски.   Успешное выполнение фазы разработки означает достижение вехи архитектуры жизненного цикла (англ. Lifecycle Architecture Milestone). |
| 3. Построение (Construction) | В фазе «Построение» происходит реализация большей части функциональности продукта. Фаза завершается первым внешним релизом системы и вехой начальной функциональной готовности (Initial Operational Capability). |
| 4. Внедрение (Transition) | В фазе «Внедрение» создается финальная версия продукта и передается от разработчика к заказчику. Это включает в себя программу бета-тести-рования, обучение пользователей, а также определение качества продук-та. В случае, если качество не соответствует ожиданиям пользователей или критериям, установленным в фазе «Начало», фаза «Внедрение» пов-торяется снова. Выполнение всех целей означает достижение вехи гото-вого продукта (Product Release) и завершение полного цикла разработки. |

***2.2.7  Корпоративный стандарт MSF***

**Microsoft Solutions Framework** (**MSF**) — методология разработки АС и их программного обеспечения, предложенная корпорацией Microsoft. MSF описывает управление людьми и рабочими процессами при разработке решения. MSF представляет собой согласованный набор концепций, моделей и правил. Благодаря своей гибкости, масштабируемости и отсутствию жестких инструкций MSF способна удовлетворить нужды организации или проектной группы любого размера.

Методология MSF состоит из принципов, моделей и дисциплин по управлению персоналом, процессами, технологическими элементами и связанными со всеми этими факторами вопросами, характерными для большинства проектов. MSF состоит из двух моделей и трех дисциплин (табл. 2.28). Их краткая характеристика дана в табл. 2.29 - табл. 2.31.

Таблица 2.28

Перечень моделей и дисциплин MSF

|  |  |
| --- | --- |
| Объект | Наименование |
| Модели | модель проектной группы  модель процессов |
| Дисциплины | дисциплина *управление проектами*  дисциплина *управление рисками*  дисциплина *управление подготовкой* |

Таблица 2.29

Краткая характеристика модели проектной группы MSF

|  |  |
| --- | --- |
| Объект | Описание |
| Сущность модели проектной группы | Модель проектной группы MSF (MSF Team Model) описывает подход Майкрософт к организации работающего над проектом персонала и его деятельности в целях максимизации успешности проекта. Данная модель определяет ролевые кластеры, их области компетенции и зоны ответственности, а также рекомендации членам проектной группы, позволяющие им успешно осуществить свою миссию по воплощению проекта в жизнь.  В соответствии с моделью MSF проектные группы строятся как небольшие многопрофильные команды, члены которых распределяют между собой ответственность и дополняют области компетенций друг друга. Это дает возможность четко сфокусировать внимание на нуждах проекта. Проектную группу объединяет единое видение проекта, стремление к воплощению его в жизнь, высокие требования к качеству работы и желание самосовершенствоваться.  Модель проектной группы MSF предлагает разбиение больших команд (более 10 человек) на малые многопрофильные группы направлений (feature teams). Эти малые коллективы работают параллельно, регулярно синхронизируя свои усилия. Кроме того, когда ролевому кластеру требуется много ресурсов, формируются т.н. функциональные группы (functional teams), которые затем объединяются в ролевые кластеры.  Минимальный коллектив по MSF может состоять всего из трех человек. |
| Основные принципы MSF, имею-щие отноше-ние к успеш-ной работе команды | 1.     Распределение ответственности при фиксации отчетности  2.     Наделяйте членов команды полномочиями  3.     Концентрируйтесь на бизнес-приоритетах  4.     Единое видение проекта  5.     Проявляйте гибкость — будьте готовы к переменам  6.     Поощряйте свободное общение |
| Ключевые концепции модели проектной группы MSF | 1.     Команда соратников  2.     Сфокусированность на нуждах заказчика  3.     Нацеленность на конечный результат  4.     Установка на отсутствие дефектов  5.     Стремление к самосовершенствованию  6.     Заинтересованные команды работают эффективно |
| Ролевые кластеры проектной группы по MSF и их зоны ответственности | ·        управление программой (program manager) — разработка архитектуры решения, административные службы;  ·        разработка (developer) — разработка приложений и инфраструктуры, технологические консультации;  ·        тестирование (QAE) — планирование, разработка тестов и отчетность по тестам;  ·        управление выпуском (release manager) — инфраструктура, сопровождение, бизнес-процессы, выпуск готового продукта;  ·        удовлетворение заказчика (user experience) — обучение, эргономика, графический дизайн, техническая поддержка;  ·        управление продуктом (product manager) — бизнес-приоритеты, маркетинг, представительство интересов заказчика. |
| Подход к управлению проектами | ·       ответственность за управление проектом распределенная между лидерами ролевых кластеров внутри команды — каждый член проектной группы отвечает за общий успех проекта и качество создаваемого продукта.  ·       профессиональные менеджеры выступают в качестве консультантов и наставников команды, а не выполняют функции контроля над ней — в эффективно работающей команде каждый её член имеет необходимые полномочия для выполнения своих обязанностей и уверен, что получит от коллег все необходимое.        Одна из характерных особенностей MSF — отсутствие должности менеджера проекта. |

Таблица 2.30

Краткая характеристика модели процессов MSF

|  |  |
| --- | --- |
| Объект | Описание |
| Сущность модели процессов | **Модель процессов MSF** (MSF process model) представляет общую методологию разработки и внедрения IT решений. Особенность этой модели состоит в том, что благодаря своей гибкости и отсутствию жестко навязываемых процедур она может быть применена при разработке весьма широкого круга IT проектов. Эта модель сочетает в себе свойства двух стандартных производственных моделей: каскадной (waterfall) и спиральной (spiral). Модель процессов в MSF 3.0 была дополнена ещё одним инновационным аспектом: она покрывает весь жизненный цикл создания решения, начиная с его отправной точки и заканчивая непосредственно внедрением. Такой подход помогает проектным группам сфокусировать свое внимание на бизнес-отдаче (business value) решения, поскольку эта отдача становится реальной лишь после завершения внедрения и начала использования продукта.  Процесс MSF ориентирован на «**вехи**» (milestones) — ключевые точки проекта, характеризующие достижение в его рамках какого-либо существенного (промежуточного либо конечного) результата. Этот результат может быть оценен и проанализирован, что подразумевает ответы на вопросы: «Пришла ли проектная группа к однозначному пониманию целей и рамок проекта?», «В достаточной ли степени готов план действий?», «Соответствует ли продукт утвержденной спецификации?», «Удовлетворяет ли решение нужды заказчика?» и т. д.  Модель процессов MSF учитывает постоянные изменения проектных требований. Она исходит из того, что разработка решения должна состоять из коротких циклов, создающих поступательное движение от простейших версий решения к его окончательному виду. |
| Особенности модели процессов MSF | ·        Подход, основанный на фазах и вехах.  ·        Итеративный подход.  ·        Интегрированный подход к созданию и внедрению решений. |
| Основные фазы процесса разработки | ·        Выработка концепции (Envisioning)  ·        Планирование (Planning)  ·        Разработка (Developing)  ·        Стабилизация (Stabilizing)  ·        Внедрение (Deploying)  Кроме этого существует большое количество *промежуточных вех*, которые показывают достижение в ходе проекта определенного прогресса и расчленяют большие сегменты работы на меньшие, обозримые участки. Для каждой фазы модели процессов MSF определяет:  ·       что (какие артефакты) является результатом этой фазы  ·       над чем работает каждый из ролевых кластеров на этой фазе |
| Итеративность разработки приложений | В рамках MSF программный код, документация, дизайн, планы и другие рабочие материалы создаются, как правило, итеративными методами. MSF рекомендует начинать разработку решения с построения, тестирования и внедрения его базовой функциональности. Затем к решению добавляются все новые и новые возможности. Такая стратегия именуется стратегией версионирования. Несмотря на то, что для малых проектов может быть достаточным выпуск одной версии, рекомендуется не упускать возможности создания для одного решения ряда версий. С созданием новых версий эволюционирует функциональность решения.  Итеративный подход к процессу разработки требует использования гибкого способа ведения документации. «Живые» документы (living documents) должны изменяться по мере эволюции проекта вместе с изменениями требований к конечному продукту. В рамках MSF предлагается ряд шаблонов стандартных документов, которые являются артефактами каждой стадии разработки продукта и могут быть использованы для планирования и контроля процесса разработки.  Решение не представляет бизнес-ценности, пока оно не внедрено. Именно по этой причине модель процессов MSF содержит весь жизненный цикл создания решения, включая его внедрение — вплоть до момента, когда решение начинает давать отдачу. |

Таблица 2.31

Краткая характеристика дисциплин MSF

|  |  |
| --- | --- |
| Дисциплина | Характеристика |
| Управление проектами  (project management) | Управление проектами (project management) — это область знаний, навыков, инструментария и приемов, используемых для достижения целей проектов в рамках согласованных параметров качества, бюджета, сроков и прочих ограничений.  Очевидна взаимозависимость между ресурсами проекта (людскими и финансовыми), его календарным графиком (временем) и реализуемыми возможностями (рамками). Эти три переменные образуют так называемый «**треугольник компромиссов**». Нахождение верного баланса между ресурсами, временем разработки и возможностями — ключевой момент в построении решения, должным образом отвечающего нуждам заказчика.  Другое весьма полезное средство для управления проектными компромиссами — **матрица компромиссов** проекта (project tradeoff matrix). Она отражает достигнутое на ранних этапах проекта соглашение между проектной группой и заказчиком о выборе приоритетов в возможных в будущем компромиссных решениях. В определенных случаях из этой приоритезации могут делаться исключения, но в целом следование ей облегчает достижение соглашений по спорным вопросам.  Для иллюстрации использования матрицы компромиссов MSF предлагает использовать следующее предложение (вместо пропущенных слов могут быть вставлены «календарный график», «ресурсы» и «функциональность»): «Зафиксировав \_\_\_\_\_\_\_\_\_\_\_, мы согласовываем \_\_\_\_\_\_\_\_\_\_\_ и принимаем результирующий \_\_\_\_\_\_\_\_\_\_\_».  В MSF нет роли «менеджер проекта». Деятельность по управлению проектом распределяется между лидерами групп и ролевым кластером «Управление программой». Для лидеров групп и ролевого кластера «Управление программой» инструментом управления проектом, облегчающим создание планов и календарных графиков, является **WBS**. Иерархическая структура работ (Work Breakdown Structure — WBS) — это структуризация работ проекта, отражающая его основные результаты и определяющая его рамки. Работа, не описанная в WBS, находится вне границ проекта. В MSF создание WBS является коллективной деятельностью, в которую вовлекаются все ролевые кластеры. Каждая роль ответственна за предоставление детального описания собственной работы. |
| Управление рисками  (MSF risk management discipline | MSF видит в изменениях и возникающей из-за них неопределенности неотъемлемые части жизненного цикла информационных технологий. Дисциплина отстаивает превентивный подход к работе с рисками в условиях такой неопределенности, непрерывное оценивание рисков и использование информации о рисках в рамках процесса принятия решений на протяжении всего жизненного цикла проекта. Дисциплина предлагает принципы, идеи и рекомендации, подкрепленные описанием пошагового процесса для успешного активного управления рисками. Этот процесс включает в себя выявление и анализ рисков; планирование и реализацию стратегий по их профилактике и смягчению возможных последствий; отслеживание состояния рисков и извлечение уроков из обретенного опыта. Девиз MSF — мы не боремся с рисками — мы ими управляем. |
| Управление подготовкой | Дисциплина посвящена управлению знаниями, профессиональными умениями и способностями, необходимыми для планирования, создания и сопровождения успешных решений. Она описывает фундаментальные принципы MSF и дает рекомендации по применению превентивного подхода к управлению знаниями на протяжении всего жизненного цикла информационных технологий. Эта дисциплина также рассматривает планирование процесса управления подготовкой. Будучи подкрепленной испытанными практическими методиками, дисциплина управления подготовкой предоставляет проектным группам и отдельным специалистам базу для осуществления этого процесса. |

***2.2.8  Корпоративный стандарт DATARUN***

**2.2.8.1 Методологии проектирования АС как программные продукты**

Современные методологии описания предметной области и реализующие их технологии поставляются в электронном виде вместе с CASE-средствами и включают библиотеки процессов, шаблонов, методов, моделей и других компонентов, предназначенных для построения того класса АС, на который ориентирована методология. Такие комплексные методологии выделяют в самостоятельный класс электронных методологий, называемых еще электронными методологиями системного проектирования. В целом они представляют собой комплекс согласованных инструментальных средств, образующих специализированную среду разработки АС.

Электронные методологии включают также средства, которые должны обеспечивать их адаптацию для конкретных пользователей и развитие методологии по результатам выполнения конкретных проектов.

Процесс адаптации заключается в удалении ненужных процессов, действий ЖЦ и других компонентов методологии, в изменении неподходящих или в добавлении собственных процессов и действий, а также методов, моделей, стандартов и руководств. Настройка методологии может осуществляться также по следующим аспектам: этапы и операции ЖЦ, участники проекта, используемые модели ЖЦ, поддерживаемые концепции и др.

Одной из наиболее распространенных в мире электронных методологий является методология DATARUN.

**2.2.8.2 Сущность методологии DATARUN**

В соответствии с методологией DATARUN ЖЦ АС разбивается на стадии, которые связываются с результатами выполнения основных процессов, определяемых стандартом ISO 12207. Каждую стадию кроме ее результатов должен завершать план работ на следующую стадию. В табл. 2.32 описаны работы, выполняемые на каждой стадии ЖЦ согласно методологии DATARUN.

Таблица 2.32

Работы, выполняемые на каждой стадии согласно методологии DATARUN

|  |  |
| --- | --- |
| Стадия | Выполняемые на стадии работы |
| Стадия форми-рования требова-ний и планиро-вания | Определение начальных оценок объема и стоимости проекта. Формулируются требования и экономическое обоснование для разработки АС, строятся функциональные модели (модели бизнес-процессов организации) и исходная концептуальная модель данных, которые дают основу для оценки технической реализуемости проекта. Основные результаты стадии: модели деятельности организации (исходные модели процессов и данных организации), требования к системе, включая требования по сопряжению с существующими АС, исходный бизнес-план. |
| Стадия концеп-туально-го про-ектиро-вания | Детальный анализ первичных данных и уточнение концептуальной модели данных. Проектирование архитектуры системы. Архитектура включает в себя разделение концептуальной модели на обозримые подмодели. Оценивается возможность использования существующих АС и выбирается соответствую-щий метод их преобразования. После построения проекта уточняется исходный бизнес-план. Выходные компоненты стадии: концептуальная модель данных, модель архитектуры системы и уточненный бизнес-план. |
| Стадия специфи-кации приложе-ний | Продолжается процесс создания и детализации проекта. Концептуальная модель данных преобразуется в реляционную модель данных. Определяется структура приложения, необходимые интерфейсы приложения в виде экранов, отчетов и пакетных процессов вместе с логикой их вызова. Модель данных уточняется бизнес-правилами и методами для каждой таблицы. Принимается окончательное решение о способе реализации приложений. По результатам стадии должен быть построен проект АС, включающий модели архитектуры АС, данных, функций, интерфейсов (с внешними системами и с пользователями), требований к разрабатываемым приложениям (модели данных, интерфейсов и функций), требований к доработкам существующих АС, требований к интеграции приложений, а также сформирован окончательный план создания АС. |
| Стадия разработ-ки, инте-грации и тестиро-вания | Создается тестовая база данных, частные и комплексные тесты. Проводится разработка, прототипирование и тестирование баз данных и приложений в соответствии с проектом. Отлаживаются интерфейсы с существующими системами. Описывается конфигурация текущей версии ПО. На основе результатов тестирования проводится оптимизация базы данных (БД) и приложений. Приложения интегрируются в систему, проводится тестирование приложений в составе системы и испытания системы. Основные результаты стадии: готовые приложения, проверенные в составе системы на комплексных тестах, текущее описание конфигурации ПО, скорректированная по результатам испытаний версия системы и ее эксплуатационная документация. |
| Стадия внедре-ния | Установка и внедрение БД и приложений. Основные результаты стадии: готовая к эксплуатации и перенесенная на программно-аппаратную платформу заказчика версия системы, документация сопровождения и акт приемочных испытаний по результатам опытной эксплуатации. |
| Стадии сопрово-ждения и развития | Регистрация, диагностика и локализация ошибок, внесение изменений и тестирование, проведение доработок, тиражирование и распространение новых версий ПО в места его эксплуатации, перенос приложений на новую платформу и масштабирование системы. Стадия развития фактически является повторной итерацией стадии разработки. |

Методология DATARUN базируется на системном подходе к описанию деятельности организации и опирается на две модели или на два представления: модель организации и модель АС. Построение моделей начинается с описания процессов, из которых затем извлекаются первичные данные (стабильное подмножество данных, которые организация должна использовать для своей деятельности). Первичные данные описывают продукты или услуги организации, выполняемые операции (транзакции) и потребляемые ресурсы. К первичным относятся данные, которые описывают внешние и внутренние сущности, такие как служащие, клиенты или агентства, а также данные, полученные в результате принятия решений, как например, графики работ, цены на продукты.

Основной принцип DATARUN заключается в том, что первичные данные, если они должным образом организованы в модель данных, становятся основой для проектирования архитектуры АС. Архитектура АС будет более стабильной, если она основана на первичных данных, тесно связанных с основными деловыми операциями, определяющими природу бизнеса, а не на традиционной функциональной модели.

Любая АС (рис. 2.5) представляет собой набор модулей, исполняемых процессорами и взаимодействующих с базами данных.

Рис. 2.5 Модель АС с точки зрения  DATARUN

Базы данных и процессоры могут располагаться централизованно или быть распределенными. События в системе могут инициироваться внешними сущностями, такими как клиенты у банкоматов или временные события (конец месяца или квартала). Все транзакции осуществляются через объекты или модули интерфейса, которые взаимодействуют с одной или более базами данных.

Подход DATARUN преследует две цели:

определить стабильную структуру, на основе которой будет строиться АС. Такой структурой является модель данных, полученная из первичных данных, представляющих фундаментальные процессы организации;

спроектировать АС на основании модели данных.

Объекты, формируемые на основании модели данных, являются объектами базы данных, обычно размещаемыми на серверах в среде клиент/сервер. Объекты интерфейса, определенные в архитектуре компьютерной системы, обычно размещаются на клиентской части. Модель данных, являющаяся основой для спецификации совместно используемых объектов базы данных и различных объектов интерфейса, обеспечивает сопровождаемость АС. На рисунке 2.6 представлена последовательность шагов проектирования АС.

Рис. 2.6  Последовательность шагов проектирования системы

**2.2.8.3 Модели, создаваемые в рамках подхода DATARUN**

На рисунке 2.7 определены модели, создаваемые в процессе разработки АС. Для их создания используется CASE-средство Silverrun. Silverrun обеспечивает автоматизацию проведения проектных работ в соответствии с методологией DATARUN. Предоставляемая этими средствами среда проектирования дает возможность руководителю проекта контролировать проведение работ, отслеживать выполнение работ, вовремя замечать отклонения от графика. Каждый участник проекта, подключившись к этой среде, может выяснить содержание и сроки выполнения порученной ему работы, детально изучить технику ее выполнения в гипертексте по технологиям, и вызвать инструмент (модуль Silverrun) для реального выполнения работы. АС создается последовательным построением ряда моделей, начиная с модели бизнес-процессов и заканчивая моделью программы, автоматизирующей эти процессы.

**BPM** (**B**usiness **P**rocess **M**odel) - модель бизнес-процессов.  
**PDS** (**P**rimary **D**ata **S**tructure) - структура первичных данных.  
**CDM** (**C**onceptual **D**ata **M**odel) - концептуальная модель данных.  
**SPM** (**S**ystem **P**rocess **M**odel) - модель процессов системы.  
**ISA** (**I**nformation **S**ystem **A**rchitecture) - архитектура информационной системы.  
**ADM** (**A**pplication **D**ata **M**odel) - модель данных приложения.  
**ISM** (**I**nterface **S**pecification **M**odel) - модель спецификации интерфейса**.**

**IPM** (**I**nterface **P**resentation **M**odel) - модель представления интерфейса.

Рис. 2.7  Модели, создаваемые в рамках подхода DATARUN

Создаваемая АС должна основываться на функциях, выполняемых организацией. Поэтому первая создаваемая модель - это модель бизнес-процессов, построение которой осуществляется в модуле Silverrun BPM. Характеристика BPM и других моделей, а также их взаимодействие описаны в табл. 2.33.

Таблица 2.33

Характеристика моделей, создаваемых в рамках подхода DATARUN

|  |  |
| --- | --- |
| Модель | Характеристика |
| **BPM B**usiness **P**rocess **M**odel (модель бизнес-процессов | Для этой модели используется специальная нотация BPM. В процессе анализа и спецификации бизнес-функций выявляются основные информационные объекты, которые документируются как структуры данных, связанные с потоками и хранилищами модели. Источниками для создания структур являются используемые в организации документы, должностные инструкции, описания производственных операций. Эти данные вводятся в том виде, как они существуют в деятельности организации. Нормализация и удаление избыточности производится позже при построении концептуальной модели данных в модуле Silverrun ERX. После создания модели бизнес-процессов информация сохраняется в репозитории проекта. |
| **PDS**  **P**rimary **D**ata **S**tructure - структура первичных данных | В процессе обследования работы организации выявляются и документируются структуры первичных данных. Эти структуры заносятся в репозиторий модуля BPM при описании циркулирующих в организации документов, сообщений, данных. В модели бизнес-процессов первичные структуры данных связаны с потоками и хранилищами информации. |
| **CDM** **C**onceptual **D**ata **M**odel – концепту-альная модель данных. | На основе структур первичных данных в модуле Silverrun ERX создается концептуальная модель данных (ER-модель). От структур первичных данных концептуальная модель отличается удалением избыточности, стандартизацией наименований понятий и нормализацией. Эти операции в модуле ERX выполняются при помощи встроенной экспертной системы. Цель концептуальной модели данных - описать используемую информацию без деталей возможной реализации в базе данных, но в хорошо структурированном нормализованном виде. |
| **SPM**  **S**ystem **P**rocess **M**odel – мо-дель процес-сов системы | С помощью модели системных процессов детально документируется поведение каждого приложения. В модуле BPM создается модель системных процессов, определяющая, каким образом реализуются бизнес-процессы. Эта модель создается отдельно для каждого приложения и тесно связана с моделью данных приложения. |
| **ISA**  **I**nformation **S**ystem **A**rchitecture- архитектура информаци-онной системы. | На основе модели бизнес-процессов и концептуальной модели данных проектируется архитектура АС. Определяются входящие в систему приложения, для каждого приложения специфицируются используемые данные и реализуемые функции. Архитектура АС создается в модуле Silverrun BPM с использованием специальной нотации ISA. Основное содержание этой модели - структурные компоненты системы и навигация между ними. Концептуальная модель данных разбивается на части, соответствующие входящим в состав системы приложениям. |
| **ADM A**pplication **D**ata **M**odel - модель данных приложения | Приложение состоит из интерфейсных объектов (экранных форм, отчетов, процедур обработки данных). Каждый такой интерфейс системы имеет дело с подмножеством базы данных. В модели данных приложения (созданной в модуле RDM) создается подсхема базы данных для каждого интерфейса этого приложения. Уточняются также правила обработки данных, специфичные для каждого интерфейса. Интерфейс работает с данными в ненормализованном виде, поэтому спецификация данных, как ее видит интерфейс, оформляется как отдельная подсхема модели данных интерфейса. |
| **ISM** **I**nter-face **S**pecifi-cation **M**odel – модель специфика-ции интер-фейса | После создания подсхем реляционной модели для приложений проектируется детальная структура каждого приложения в виде схемы навигации экранов, отчетов, процедур пакетной обработки. На данном шаге эта структура детализируется до указания конкретных столбцов и таблиц базы данных, правил их обработки, вида экранных форм и отчетов. Полученная модель детально документирует приложение и непосредственно используется для программирования специфицированных интерфейсов. |
| **IPM** **I**nterface **P**resentation **M**odel- модель представ-ления интерфейса | Модель представления интерфейса - это описание внешнего вида интерфейса, как его видит конечный пользователь системы. Это может быть как документ, показывающий внешний вид экрана или структуру отчета, так и сам экран (отчет), созданный с помощью одного из средств визуальной разработки приложений - так называемых языков четвертого поколения (4GL - Fourth Generation Languages). Так как большинство языков 4GL позволяют быстро создавать работающие прототипы приложений, пользователь имеет возможность увидеть работающий прототип системы на ранних стадиях проектирования. |

Перед разработкой приложений должна быть спроектирована структура корпоративной базы данных. DATARUN предполагает использование базы данных, основанной на реляционной модели. Концептуальная модель данных после нормализации переносится в модуль реляционного моделирования Silverrun RDM с помощью специального моста ERX-RDM. Преобразование модели из формата ERX в формат RDM происходит автоматически без вмешательства пользователя. После преобразования форматов получается модель реляционной базы данных. Эта модель детализируется в модуле Silverrun RDM определением физической реализации (типов данных СУБД, ключей, индексов, триггеров, ограничений ссылочной целостности). Правила обработки данных можно задавать как непосредственно на языке программирования СУБД, так и в декларативной форме, не привязанной к реализации. Мосты Silverrun к реляционным СУБД переводят эти декларативные правила на язык требуемой системы, что снижает трудоемкость программирования процедур сервера базы данных, а также позволяет из одной спецификации генерировать приложения для разных СУБД.

После формирования всех моделей, с помощью средств разработки приложений происходит физическое создание системы: приложения программируются и интегрируются в АС.

***2.2.9 Семейство гибких методологий Agile***

**2.2.9.1 Общая характеристика гибких методологий разработки семейства Agile**

Гибкая методология разработки (англ. *Agile software development*) — это концептуальный каркас, в рамках которого выполняется разработка АС и их программного обеспечения.

Agile — семейство процессов разработки, а не единственный подход в разработке программного обеспечения, и определяется международным соглашением Agile Manifesto, которое разработано и принято 11-13 февраля 2001 года на лыжном курорте The Lodge at Snowbird в горах Юты (США). Манифест подписали представители методологий, перечисленных в табл. 2.34.

Таблица 2.34

Представители семейства гибких методологий Agile

|  |  |
| --- | --- |
| № | Частная методология |
| 1. | Adaptive Software Development, |
| 2. | Agile Modeling |
| 3. | Agile Unified Process  (AUP) |
| 4. | Agile Data Method |
| 5 | Crystal Clear |
| 6 | DSDM |
| 7. | Essential Unified Process  (EssUP) |
| 8. | Feature-Driven Development |
| 9. | Экстремальное программирование (Extreme programming, XP) |
| 10. | Feature Driven Development (FDD) |
| 11. | Getting Real |
| 12. | Open Unified Process (OpenUP) |
| 13. | Pragmatic Programming |
| 14. | Scrum |
| 15. | Бережливая разработка программного обеспечения (Lean Software Development) |

Примечательно что, Agile Manifesto не содержит практических советов, т.е. Agile не включает практик, а определяет ценности и принципы, которыми руководствуются успешные команды. Сущность Agile, содержащиеся в Agile Manifesto 4 основные идеи и 12 принципов, а также объекты критики представлены в табл. 2.35.

Таблица 2.35

Сущность, идеи, принципы и объекты критики Agile

|  |  |
| --- | --- |
| Объект | Описание |
| Сущ-ность | Большинство гибких методологий нацелены на минимизацию рисков, путём сведения разработки к серии коротких циклов, называемых *итерациями*, которые обычно длятся две-три недели. Каждая итерация сама по себе выглядит как программный проект в миниатюре, и включает все задачи, необходимые для выдачи мини-прироста по функциональности: планирование, анализ требований, проектирование, кодирование, тестирование и документирование. Хотя отдельная итерация, как правило, недостаточна для выпуска новой версии продукта, подразумевается, что гибкий программный проект готов к выпуску в конце каждой итерации. По окончании каждой итерации, команда выполняет переоценку приоритетов разработки.  Agile-методы делают упор на непосредственное общение лицом к лицу. Большинство agile-команд расположены в одном офисе, иногда называемом bullpen. Как минимум, она включает и «заказчиков» (product owner – заказчик или его полномочный представитель, определяющий требования к продукту; эту роль может выполнять менеджер проекта, бизнес-аналитик или клиент). Офис может также включать тестировщиков, дизайнеров интерфейса, технических писателей и менеджеров.  Основной метрикой agile-методов является рабочий продукт. Отдавая предпочтение непосредственному общению, agile-методы уменьшают объем письменной документации по сравнению с другими методами. Это привело к критике этих методов, как недисциплинированных. |
| Идеи | ·    Личности и их взаимодействия важнее, чем процессы и инструменты;  ·    Работающее программное обеспечение важнее, чем полная документация;  ·    Сотрудничество с заказчиком важнее, чем контрактные обязательства;  ·    Реакция на изменения важнее, чем следование плану. |
| Прин-ципы | ·   удовлетворение клиента за счёт ранней и бесперебойной поставки ценного ПО;  ·   приветствие изменений требований, даже в конце разработки (это может повысить конкурентоспособность полученного продукта);  ·   частая поставка рабочего ПО (каждый месяц или неделю или ещё чаще);  ·   тесное, ежедневное общение заказчика с разработчиками на протяжении всего проекта;  ·   проектом занимаются мотивированные личности, которые обеспечены нужными условиями работы, поддержкой и доверием;  ·   рекомендуемый метод передачи информации — личный разговор (лицом к лицу);  ·   работающее ПО — лучший измеритель прогресса;  ·   спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок;  ·   постоянное внимание на улучшение технического мастерства и удобный дизайн;  ·   простота — искусство НЕ делать лишней работы;  ·   лучшие технические требования, дизайн и архитектура получаются у самоорганизованной команды;  ·   постоянная адаптация к изменяющимся обстоятельствам. |
| Объек-ты крити-ки | Основной пункт критики: при Agile-подходе часто пренебрегают созданием "дорожной карты" развития проекта, равно как и управлением требованиями, в процессе которого и формируется оная "карта". Подход Agile к управлению требованиями не подразумевает далеко идущих планов (по сути, управления требованиями просто не существует в данной методологии), а подразумевает возможность заказчика вдруг и неожиданно, в конце каждой итерации, выставлять новые требования, часто противоречащие архитектуре уже созданного и поставляемого продукта. Такое иногда приводит к тяжелым "авралам" с массовыми переделками практически на каждой итерации.  Кроме того, считается, что работа в Agile мотивирует разработчиков решать все поступившие задачи простейшим и быстрейшим возможным способом, зачастую не обращая внимания на правильность кода с точки зрения требований нижележащей платформы (подход - "работает, и ладно", при этом не учитывается, что созданная система может перестать работать при малейшем изменении или же дать тяжелые к воспроизводству дефекты после реального развертывания у клиента). Это приводит к снижению качества продукта и накоплению дефектов. |

**2.2.9.2 Экстремальное программирование как пример гибкой методологии разработки**

Экстрема́льное программи́рование (англ. *Extreme Programming*, *XP*) — одна из гибких методологий разработки АС и их программного обеспечения. Авторы методологии — Кент Бек, Уорд Каннингем, Мартин Фаулер и другие.

Экстремальное программирование включает двенадцать основных приёмов, которые могут быть объединены в четыре группы (табл. 2.36).

Таблица 2.36

Группы приемов экстремального программирования

|  |  |
| --- | --- |
| Группа | Приемы |
| Короткий цикл обратной связи | o   Разработка через тестирование (тестирование модулей и приемочное тестирование)  o   Игра в планирование  o   Заказчик всегда рядом  o   Парное программирование |
| Непрерывный, а не пакетный процесс | o   Непрерывная интеграция  o   Рефакторинг  o   Частые небольшие релизы |
| Понимание, разделяемое всеми | o   Простота дизайна  o   Метафора системы  o   Коллективное владение кодом или выбранными шаблонами проектирования  o   Стандарт кодирования |
| Социальная защищенность программиста | o   40-часовая рабочая неделя |

Краткая характеристика приемов экстремального программирования дана в табл. 2.37.

Таблица 2.37

Краткая характеристика приемов экстремального программирования

|  |  |
| --- | --- |
| Прием | Характеристика |
| Тестирова-ние | Разработчик не может быть уверен в правильности написанного им кода до тех пор, пока не сработают абсолютно все тесты модулей разрабатываемой им системы. Тесты модулей позволяют разработчикам убедиться в том, что их код работает корректно. Они также помогают другим разработчикам понять, зачем нужен тот или иной фрагмент кода и как он функционирует. Тесты модулей также позволяют разработчику без каких-либо опасений выполнять рефакторинг (refactoring).  Приемочные тесты позволяют убедиться в том, что система действительно обладает заявленными возможностями. Кроме того, приемочные тесты позволяют проверить корректность функционирования разрабатываемого продукта.  Для XP более приоритетным является подход, называемый TDD (Test Driven Development) - сначала пишется тест, который не проходит, затем пишется код, чтобы тест прошел, а после этого делается рефакторинг кода. |
| Игра в планирова-ние | Основная цель игры в планирование — быстро сформировать приблизительный план работы и постоянно обновлять его по мере того, как условия задачи становятся все более четкими. Артефактами игры в планирование является набор бумажных карточек, на которых записаны пожелания заказчика (customer stories), и приблизительный план работы по выпуску следующих одной или нескольких небольших версий продукта. Критическим фактором, благодаря которому такой стиль планирования оказывается эффективным, является то, что в данном случае заказчик отвечает за принятие бизнес-решений, а команда разработчиков отвечает за принятие технических решений. Если не выполняется это правило, весь процесс распадается на части. |
| Заказчик всегда рядом | «Заказчик» в XP — это не тот, кто оплачивает счета, а тот, кто на самом деле использует систему. XP утверждает, что заказчик должен быть всё время на связи и доступен для вопросов. |
| Парное программи-рование | *Парное программирование* предполагает, что весь код создается парами программистов, работающих за одним компьютером. Один из них работает непосредственно с текстом программы, другой просматривает его работу и следит за общей картиной происходящего. При необходимости клавиатура свободно передается от одного к другому. В течение работы над проектом пары не фиксированы: рекомендуется их перемешивать, чтобы каждый программист в команде имел хорошее представление о всей системе. Т.о. парное программирование усиливает взаимодействие внутри команды. |
| Непрерыв-ная интеграция | Если выполнять интеграцию разрабатываемой системы достаточно часто, можно избежать большей части связанных с этим проблем. В традиционных методиках интеграция, как правило, выполняется в самом конце работы над продуктом, когда считается, что все составные части разрабатываемой системы полностью готовы. В XP интеграция кода всей системы выполняется несколько раз в день, после того, как разработчики убедились в том, что все тесты модулей корректно срабатывают. |
| Рефакторинг | Рефакторинг (refactoring) — это методика улучшения кода, без изменения его функциональности. XP подразумевает, что однажды написанный код в процессе работы над проектом почти наверняка будет неоднократно переделан. Разработчики XP безжалостно переделывают написанный ранее код для того, чтобы улучшить его. Этот процесс называется рефакторингом. Отсутствие тестового покрытия провоцирует отказ от рефакторинга, в связи с боязнью поломать систему, что приводит к постепенной деградации кода. |
| Частые небольшие релизы | Версии (releases) продукта должны поступать в эксплуатацию как можно чаще. Работа над каждой версией должна занимать как можно меньше времени. При этом каждая версия должна быть достаточно осмысленной с точки зрения полезности для бизнеса. Чем раньше выпущена первая рабочая версия продукта, тем раньше заказчик начнет получать за счет нее дополнительную прибыль. Следует помнить, что деньги, заработанные сегодня, стоят дороже, чем деньги, заработанные завтра. Чем раньше заказчик приступит к эксплуатации продукта, тем раньше разработчики получат от него информацию о том, что соответствует требованиям заказчика, а что — нет. Эта информация может оказаться чрезвычайно полезной при планировании следующего выпуска. |
| Простота дизайна | XP исходит из того, что в процессе работы условия задачи могут неоднократно измениться, а значит, разрабатываемый продукт не следует проектировать заблаговременно целиком и полностью. Если в самом начале работы вы пытаетесь от начала и до конца детально спроектировать систему, вы напрасно тратите время. XP предполагает, что проектирование — это настолько важный процесс, что его необходимо выполнять постоянно в течение всего времени работы над проектом. Проектирование должно выполняться небольшими этапами, с учетом постоянно изменяющихся требований. В каждый момент времени мы пытаемся использовать наиболее простой дизайн, который подходит для решения текущей задачи. При этом мы меняем его по мере того, как условия задачи меняются. |
| Метафора системы | Архитектура — это некоторое представление о компонентах системы и о том, как они взаимосвязаны между собой. Разработчики используют архитектуру для того, чтобы понять, в каком месте системы добавляется некоторая новая функциональность и с чем будет взаимодействовать некоторый новый компонент.  Метафора системы (system metaphor) — это аналог того, что в большинстве методик называется архитектурой. Метафора системы дает команде представление о том, каким образом система работает в настоящее время, в каких местах добавляются новые компоненты и какую форму они должны принять.  Подобрав хорошую метафору, вы облегчаете команде понимание того, каким образом устроена ваша система. Иногда сделать это не просто. |
| Стандарт кодирования | Все члены команды в ходе работы должны соблюдать требования общих стандартов кодирования. Благодаря этому:  ·         члены команды не тратят время на глупые споры о вещах, которые фактически никак не влияют на скорость работы над проектом;  ·         обеспечивается эффективное выполнение остальных практик.  Если в команде не используются единые стандарты кодирования, разработчикам становится сложнее выполнять рефакторинг; при смене партнеров в парах возникает больше затруднений; в общем и целом, продвижение проекта затрудняется. В рамках XP необходимо добиться того, чтобы было сложно понять, кто является автором того или иного участка кода, — вся команда работает унифицированно, как один человек. Команда должна сформировать набор правил, а затем каждый член команды должен следовать этим правилам в процессе кодирования. Перечень правил не должен быть исчерпывающим или слишком объемным. Задача состоит в том, чтобы сформулировать общие указания, благодаря которым код станет понятным для каждого из членов команды. Стандарт кодирования поначалу должен быть простым, затем он будет эволюционировать по мере того, как команда обретает опыт. Вы не должны тратить на предварительную разработку стандарта слишком много времени. |
| Коллектив-ное владение кодом | *Коллективное владение* означает, что каждый член команды несёт ответственность за весь исходный код. Таким образом, каждый вправе вносить изменения в любой участок программы. Парное программирование поддерживает эту практику: работая в разных парах, все программисты знакомятся со всеми частями кода системы. Важное преимущество коллективного владения кодом — в том, что оно ускоряет процесс разработки, поскольку при появлении ошибки её может устранить любой программист.  Давая каждому программисту право изменять код, мы получаем риск появления ошибок, вносимых программистами, которые считают, что знают то, что делают, но не рассматривают некоторые зависимости. Хорошо определённые тесты модулей решают эту проблему: если нерассмотренные зависимости порождают ошибки, то следующий запуск тестов модулей будет неудачным. |

**2.2.9.3  Обзор гибкой методологии разработки систем OpenUP**

OpenUP - это экономичный унифицированный процесс, использующий принципы итеративности и инкрементальности в рамках структурированного жизненного цикла. OpenUP использует прагматичную философию гибкой разработки, которая имеет в своей основе коллективный подход к разработке программного обеспечения. Это независимый от инструментов, мало регламентированный процесс, который можно расширить для адаптации к широкому диапазону типов проектов.

Работа в проекте OpenUP организована по принципу микрошагов (см. рис. 2.8). Микрошаги представляют собой небольшие единицы работы, которые формируют постоянное измеряемое приращение выполнения проекта (обычно их продолжительность составляет от нескольких часов до несколько дней). Процесс предполагает интенсивную совместную работу, выполняемую заинтересованным коллективом, построенным по принципу самоорганизации. Разработка системы осуществляется постепенно, а микрошаги обеспечивают исключительно короткий цикл обратной связи, который делает возможным принятие адаптивных решений в ходе каждой итерации.

Рис. 2.8 Уровни OpenUP: микрошаги, жизненный цикл итерации и жизненный цикл проекта

**2.2.9.4  Гибкая методология разработки систем SCRUM**

Кратко и упрощенно методологиюScrum можно представить так: это итеративный процесс разработки программного обеспечения небольшой командой специалистов (до десяти человек), предполагающий создание серии последовательных выпусков программного продукта, в которых постепенно добавляется требуемая функциональность. Такой подход позволяет по завершению текущей итерации продемонстрировать заказчику работоспособный программный продукт, возможно с ограниченной функциональностью, получить отзыв, замечания и дополнительные требования, которые будут учтены в следующих итерациях.

Определение основных терминов (понятий и артефактов), используемых в Scrum,  приведено в таблице 2.38.

Таблица 2.38

Краткое определение терминов (понятий и артефактов) методологии Scrum

|  |  |
| --- | --- |
| Термин | Определение |
| Спринт | набор задач, запланированных на выполнение за определенный период времени. |
| Владелец продукта | член Scrum-команды, который отвечает за всё, что связано с потребительскими качествами программного продукта. |
| Руководитель | член Scrum-команды, который отвечает за состояние и координацию проекта, продуктивность команды и устранение препятствий, мешающих проекту. |
| Член команды | член Scrum-команды, который наравне с другими членами команды отвечает за разработку программного продукта высокого качества. |
| Элементы задела работы продукта (ЭЗРП) | список пользовательских требований, которые определяют функциональность продукта. |
| Элемент работы (ЭР) | краткое описание функций продукта. |
| Рабочий элемент (РЭ) | запись, которая создается в Visual Studio Team Foundation Server (TFS) для задания определения, назначения, приоритета и состояния элемента работы. |
| Невыполненная работа по продукту (НВР) | список элементов работы (пользовательских описаний функциональности), которые необходимо выполнить при создании программного продукта. |
| Незаконченная работа (НЗР) | список элементов работы планируемого спринта. |

Итерации, называемые спринтами, продолжаются от 1 до 4 недель (до 30 дней) каждая. В составе Scrum-команды выделяют владельца продукта (Product owner), руководителя (ScrumMaster) и членов команды (Team members). Выполнямые ими основные задачи представлены в таблице 2.39. Рядовые члены команды отвечают за разработку программного продукта высокого качества. Они должны обладать навыками в проектировании и архитектуре программного продукта, бизнес-анализе, программировании, тестировании, настройке баз данных и проектировании пользовательского интерфейса. Члены команды участвуют в планировании спринтов. Команда может включать опытных разработчиков и новичков, которые в процессе работы должны совершенствоваться при обмене знаниями с другими членами команды.

Таблица 2.39

Задачи членов Scrum-команды

|  |  |
| --- | --- |
| Роль члена команды | Задачи |
| Владелец продукта | ·     определение и приоритезация требований/функций, то есть элементов работ и задач;  ·     планирование спринтов и выпусков;  ·     тестирование требований/функций. |
| Руководитель | ·     проведение ежедневных Scrum-собраний;  ·     привлечение сотрудников вне команды:  ·     стимулирование эффективного общения членов команды;  ·     определение размера команды. |
| Член команды | ·     обязательное выполнение элементов работ, включенных в текущий спринт;  ·     акцент на взаимосвязанных задачах спринта;  ·     совершенствование команды. |

Организационно-методическую поддержку Scrum-процесса командной работы по созданию ПО АС (The Team Software Process) обеспечивают инструментальные средства, реализованные в VisualStudio 2012, в частности Visual Studio Team Foundation Server (TFS). При этом жизненный цикл проекта ПО может быть описан схемой, представленной на рисунке 2.9.

Рис. 2.9  Жизненный цикл проекта ПО на основе Scrum

Жизненный цикл проекта ПО включает формирование рабочих элементов, управление невыполненной работой, планирование и выполнение спринта, развертывание ПО у заинтересованных лиц, получение отзывов для улучшения программного продукта. Рабочий элемент - это запись, которая создается в TFS для задания определения, назначения, приоритета и состояния элемента работы. Возможны следующие типы рабочих элементов:

· невыполненная работа по продукту;

· ошибка;

· задача;

· препятствие;

· тестовый случай.

При управлении рабочим процессом элементов невыполненной работы по продукту постоянно отслеживаются связи и изменяется состояние этих элементов.

Методология Scrum имеет следующие положительные стороны:

·      пользователи начинают видеть систему спустя всего несколько недель и могут выявлять проблемы на ранних стадиях разработки программного продукта;

·      интеграция технических компонентов происходит в ходе каждого спринта и поэтому проблемы проекта (если они возникают) выявляются практически сразу;

·      в каждом спринте команда фокусируется на контроле качества;

·      выполняется гибкая работа с изменениями в проекте на уровне спринта.

Ниже методология Scrum описывается более детально.

В начале проектирования владелец продукта и заказчик формируют концепцию программного продукта, которая показывает, для кого предназначен продукт, какие преимущества получат пользователи и какие существуют конкуренты. Концепция продукта связывается с областью проекта и ограничениями. Детальные пользовательские требования, которые определяют функциональность продукта, задаются «*элементами задела работы продукта» (ЭЗРП)* (ProductBacklogItem - PBI). На основе детальных ЭЗРП владелец продукта и руководитель формируют «*элементы работы» (ЭР)*, которые представляют собой краткое описание функций продукта и оформляются в произвольной форме в виде кратких заметок. Вначале задаются наиболее важные и понятные всем пользовательские требования - ЭР. Элементы работы могут детализироваться в виде составляющих их задач. В процессе создания программного продукта ЭР могут уточняться, добавляться или удаляться из списка требований.

Далее владелец продукта создает список всех потенциальных функций продукта – «Невыполненная работа по продукту» (ProductBacklog). *Невыполненная работа по продукту*, которую в дальнейшем будем называть *невыполненная работа - НВР*, содержит список *элементов работы* - пользовательских описаний функциональности.

Управление *невыполненной работой* по проекту сводится к поддержанию *элементов работ* в актуальном состоянии. Отдельные элементы работ списка НВР могут добавляться или удаляться в процессе создания ПО. Это является результатом того, что команда получает дополнительную информацию о новых требованиях заказчика к проектируемому программному продукту, а заказчик выясняет, как реализуются его ожидания.

Для элементов работ владелец продукта совместно с командой проекта расставляет приоритеты. При планировании спринта в него включают наиболее значимые, с точки зрения владельца продукта, пользовательские требования - ЭР, которые характеризуются наибольшей потребительской ценностью. Выбранные элементы работ перемещают в список «*Незаконченная работа*» (SprintBacklog). Список «*Незаконченная работа» (НЗР)* отражает состав работ планируемого спринта. Список НЗР является результатом процесса планирования спринта.

Координацией работ в спринте занимается руководитель спринта (ScrumMaster). Он организовывает процесс приоритезации задач спринта, распределения задач между членами команды. Руководитель спринта проводит собрание по планированию работ, ежедневные собрания для краткого обсуждения результатов работы и проблем, обзорные собрания в конце спринта и выпуска.

Ежедневные Scrum-собрания имеют продолжительность 15 - 30 минут. Целью таких собраний является выявление проблем, которые тормозят процесс разработки, и определение действий по их нейтрализации. Для простых проблем сразу принимается решение по их устранению, а сложные проблемы откладываются на последующие спринты. В ходе ежедневного Scrum-собрания руководитель задает темп спринта, акцентирует команду на наиболее важных элементах списка невыполненных работ. Каждый член команды сообщает, что было сделано вчера, что будет делать сегодня и какие имеются препятствия в работе. Если на ежедневном Scrum-собрании возникают вопросы, для решения которых необходимы специалисты, которых в команде нет, тогда руководитель берет на себя анализ и возможные пути разрешения данного вопроса.

Результатом спринта является работоспособное ПО, возможно обладающее только частью необходимых функций программного продукта. Выпуск спринта может быть развернут у заинтересованных лиц для предварительного анализа соответствия ожиданиям заказчика. Итогом анализа является формирование "Отзыва" (FeedBack), что может привести к изменения содержания списка «Элемент задела работы продукта». После выполнения всех работ по программному продукту, то есть обнуления списка требований в списке «Элемент задела работы продукта» подготавливается финальный выпуск программного продукта.

Список «*Невыполненная работа по продукту»* является одним из ключевых артефактов в методологии Scrum. Успех Scrum-команды во многом определяется качественным содержанием данного списка. Список НВР, кроме ЭР, может включать нефункциональные требования. Для создания списка НВР в TFS - могут применяться различные клиентские сервисы: командный обозреватель Visual Studio; веб-доступ через Team Web Access; MS Office Excel; MS Office Project.

Итак, владелец продукта на основе требований и пожеланий клиентов формирует список функций продукта в виде ЭЗРП, которые помещаются в список НВР (рис. 2.9). При создании нового элемента невыполненной работы для него устанавливается состояние "Новый" (рис. 2.10).

Рис. 2.10  Рабочий процесс элемента невыполненной работы

После установки элементу работы приоритета его состояние изменяют на "Утверждено". На собрании по планированию спринта команда просматривает наиболее приоритетные элементы работы и выбирает те, которые будут выполняться в текущем спринте. Для элементов невыполненной работы, которые попали в текущий спринт, устанавливается состояние "Зафиксировано". Это означает тот факт, что рабочие элементы спринта не подлежат изменению до конца спринта. При завершении работы по элементу его состояние устанавливается "Выполнено".

Если для элемента работы, находящегося в состоянии "Выполнено", выявляется дополнительная работа, то этот элемент может быть переведен в состояние "Зафиксировано". Для элемента работы, находящегося в состоянии "Зафиксировано", при возникновении проблем, препятствующих его завершению в спринте, может быть работа приостановлена и установлено состояние "Утверждено". Элемент невыполненной работы может быть удален из списка *Невыполненная работа по продукту* по решению Владельца продукта.

Это может произойти как из состояния "Новый", так и состояния "Утверждено", что соответствует установке для элемента работы состояния "Удалено". В результате пересмотра элемента работы, имеющего состояние "Удалено", для него вновь возможен перевод в состояние "Новый".

Список НВР является главным документом для Scrum-команды. На основе данного списка команда создает другие рабочие элементы, составляющие спринты и выпуски. Для *Элементов невыполненной работы* команда создает задачи и тестовые случаи. Задачи детализируют элемент работы и определяют конкретную реализацию требований пользователя. Тестовые случаи необходимы для проверки соответствия функциональности кода требованиям пользователя. Если тестовый случай не проходит, то создается рабочий элемент "Ошибка". При блокировании задачи из-за невозможности её выполнения в текущем спринте создают рабочий элемент "Препятствие".

Scrum-команда может создавать вспомогательные рабочие элементы (ошибки и препятствия) в отношении элементов, на которые они влияют (задачи и тестовые случаи) и связывать эти элементы (рис. 2.11).

Рис. 2.11Связь между рабочими элементами

Для отслеживания хода выполнения проекта, можно создавать отчеты, отражающие наиболее важные данные для текущего проекта. В процессе создания ПО можно пользоваться стандартными отчетами или создавать собственные отчеты. Отчеты можно создавать, настраивать и просматривать с помощью Excel, Project или служб Reporting Services SQL Server.

[2 ИНДУСТРИАЛЬНЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627443)

[2.1 Классификация и описание методов проектирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627444)

[*2.1.1 Схема* *классификации методов проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627445)

[*2.1.2 Метод оригинального проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627446)

[*2.1.3 Методы типового проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627447)

[*2.1.4 Метод автоматизированного проектирования*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627448)

[2.2 Регламентация процессов проектирования АС отечественными и международными стандартами](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627449)

[*2.2.1 Реализуемые концептуальные подходы* *и общая характеристика стандартов проектирования АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627450)

[2.2.1.1 Cleanroom и RAD - концептуальныеподходы к организации процесса проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627451)

[2.2.1.2 Перечень и общая характеристика стандартов проектирования АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627452)

[*2.2.2  Основы канонического проектирования согласно ГОСТ 34.601-90*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627453)

[*2.2.2.1  Перечень и состав стадий создания АС согласно отечественным стандартам*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627454)

[*2.2.2.2  Виды и структура документации на АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627455)

[*2.2.2.3  Документ «Технико-экономическое обоснование создания АС»*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627456)

[*2.2.2.4  Документ «Техническое задание на создание АС»*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627457)

[*2.2.2.5  Состав комплексов документов стадий проектирования АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627458)

[*2.2.2.6  Разновидности и общая характеристика схем в документации АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627459)

[*2.2.2.7  Правила выполнения схем в документации АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627460)

[*2.2.2.8  Правила оформления текстовых документов*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627461)

[2.2.3  Международный стандарт ISO/IEC 12207](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627462)

[2.2.4  Международный стандарт ISO/IEC 15288](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627463)

[2.2.5  Корпоративный стандарт CDM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627464)

[2.2.6  Корпоративный стандарт RUP](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627465)

[2.2.7  Корпоративный стандарт MSF](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627466)

[2.2.8  Корпоративный стандарт DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627467)

[2.2.8.1 Методологии проектирования АС как программные продукты](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627468)

[2.2.8.2 Сущность методологии DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627469)

[2.2.8.3 Модели, создаваемые в рамках подхода DATARUN](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627470)

[2.2.9 Семейство гибких методологий Agile](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627471)

[2.2.9.1 Общая характеристика гибких методологий разработки семейства Agile](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627472)

[2.2.9.2 Экстремальное программирование как пример гибкой методологии разработки](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627473)

[2.2.9.3  Обзор гибкой методологии разработки систем OpenUP](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627474)

[2.2.9.4  Гибкая методология разработки систем SCRUM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk2\2%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456627475)

[3 МЕТОДЫ ОБСЛЕДОВАНИЯ И АНАЛИЗА ПРЕДМЕТНОЙ ОБЛАСТИ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805200)

[3.1 Общая характеристика процесса и методов обследования и анализа объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805201)

[3.1.1 Общие положения  по обследованию объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805202)

[3.1.2 Сущность, роль и место консалтинга при проведении обследования объекта автоматизации и создании АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805203)

[3.1.3  Процессный подход к организации деятельности предприятия](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805204)

[3.1.4 Выделение и классификация процессов на предприятии](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805205)

[3.1.5 Сущность реинжиниринга объектов автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805206)

[3.1.5.1 Понятие «реинжиниринг бизнеса» и его отличие от совершенствования бизнеса](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805207)

[3.1.5.2 Условия успешного реинжиниринга и факторы риска](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805208)

[3.2 Краткий обзор формализованных методов обследования и анализа объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805209)

[3.2.1  Методы, основанные на построении графовых моделей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805210)

[3.2.1.1 Графовая модель структурно-функциональных связей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805211)

[3.2.1.2 Метод сетевого моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805212)

[3.2.1.3 Графоаналитический метод](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805213)

[3.2.2 Методы оперограмм  и структурно-информационно- временных схем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805214)

[3.2.3  Сущность методов разработки спецификаций](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805215)

[3.3 Анализ и моделирование функциональной области внедрения АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805216)

[3.3.1  Основные понятия организационного бизнес-моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805217)

[3.3.2  Шаблоны организационного бизнес-моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805218)

[3.3.2.1 Шаблон разработки миссии](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805219)

[3.3.2.2 Шаблон формирования бизнесов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805220)

[3.3.2.3 Шаблон формирования функционала компании (основных бизнес-функций)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805221)

[3.3.2.4 Шаблон формирования зон ответственности за функционал компании](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805222)

[3.3.2.5 Шаблон потокового процессного описания](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805223)

[3.3.3  Построение организационно-функциональной модели компании](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805224)

[3.4 Общая характеристика методологий описания предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805225)

[3.4.1 Сущность структурного моделирования предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805226)

[3.4.2 Принципы структурного анализа предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805227)

[3.4.3 Сопоставление функционально-ориентированных и объектно-ориентированных методологий описания предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805228)

[3.4.4 Референтная модель бизнес-процесса](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805229)

**3 МЕТОДЫ ОБСЛЕДОВАНИЯ И АНАЛИЗА ПРЕДМЕТНОЙ ОБЛАСТИ**

**3.1 Общая характеристика процесса и методов обследования и анализаобъекта автоматизации**

**3.1.1 Общие положения  по обследованию объекта автоматизации**

Как выше указано, процесс создания АС представляет собой процесс построения и последовательного преобразования ряда согласованных моделей. В основе проектирования АС лежит моделирование объекта автоматизации или предметной области (предприятие, организация, фирма, компания и т.п.), для которой создается АС. При этом под моделью предметной области понимается некоторая система, имитирующая структуру и/или функционирование исследуемой предметной области и отвечающая основному требованию – быть адекватной этой области.

Предварительное моделирование предметной области позволяет сократить время и сроки проведения проектировочных работ и получить более эффективный и качественный проект. Без проведения моделирования предметной области велика вероятность допущения большого количества ошибок в решении стратегических вопросов, приводящих к экономическим потерям и высоким затратам на последующее перепроектирование системы. Вследствие этого все современные технологии проектирования АС основываются на использовании методологии моделирования предметной области.

Чтобы построить адекватную модель любого объекта, нужно провести его обследование.

***Oбследование*** - это изучение и анализ организационной структуры предприятия, его деятельности и существующей системы обработки информации. Например, в случае канонического проектирования материалы, полученные в результате *обследования*, используются для:

·   технико-экономического обоснования разработки и внедрения АС;

·   составления *технического задания* на разработку АС;

·   разработки технического и рабочего проектов АС.

От того, насколько глубоко предпроектное обследование вскроет внутреннюю сущность исходной системы, во многом зависят эффективность разработки и реализации последующих проектных решений.

*Обследование* содержит две составляющие: определение стратегии внедрения АС и детальный анализ деятельности организации. Основное назначениесоставляющих обследования приведено в таблице 3.1.

Первая составляющая обследования реализуется или заказчиком АС самостоятельно, или привлеченной консалтинговой организацией. Необходимо тесное взаимодействие с основными потенциальными пользователями системы и бизнес-экспертами, чтобы получить полное и однозначное понимание требований заказчика. В результате появляется возможность определить вероятные технические подходы к созданию системы и оценить затраты на ее реализацию.

Таблица 3.1

Основное назначениесоставляющих обследования

|  |  |
| --- | --- |
| Составляющая | Назначение |
| 1. Определение стратегии внедрения АС | Оценка реального объема проекта, его целей и задач на основе выявленных функций и информационных элементов автоматизируемого объекта. |
| 2. Детальный анализ деятельности организации | 1) Изучение задач, обеспечивающих реализацию функций объекта,  2) Изучение организационной структуры, штатов и содержания работ по управлению объектом,  3) Изучение характера подчиненности вышестоящим органам управления.  4) Выявление инструктивно-методических и директивных материалов, на основании которых определяются состав подсистем и перечень задач,  5) Выявление возможности применения новых методов решения задач. |

В ходе обследования аналитики собирают и фиксируют информацию в двух взаимосвязанных формах: функции (информация о событиях и процессах, которые происходят в бизнесе) и сущности (информация о вещах, имеющих значение для организации и о которых что-то известно).

Каналы получения сведений об исходной системе:

-        Изучение документов системы;

-        Беседы, опросы работников системы;

-        Непосредственное наблюдение информационных процессов в системе в их взаимосвязи с производственным процессом.

Все каналы равноправные и на практике требуется их разумное совместное использование.

В начале работ по обследованию рекомендуется подготовить документ, называемый «Программа обследования». В нем определяются состав, последовательность, методы обследования и обработки его результатов, представляются конкретные формы для заполнения в ходе сбора, обработки и анализа информации об исходной системе. Совокупность таких документов называется «Документы описания и анализа системы» (ДОАС).

На основе заполненных при обследовании объекта первичных ДОАС составляются вторичные ДОАС – сводные схемы, таблицы и т.д., необходимые для выработки и обоснования выводов и для иллюстрации положений технико-экономического обоснования в качестве приложений.

Выводы по результатам обследования должны аргументировано и в концентрированном виде отражать то, насколько сформулированная первоначально в общих чертах цель создания системы соответствует ее новому пониманию в результате обследования и анализа. Есть данные, что на исследование, описание и анализ процессов и информационных потоков на объекте приходится до 30-40% времени, затачиваемого на проектирование АС.

С целью сокращения затрат времени на этапах обследований при одновременном повышении качества проектных решений осуществляют внедрение формализованных методов представления результатов обследования и анализа. Используемые методы обычно определяются выбранной методологией проектирования и могут варьироваться в широком диапазоне: от методов, основанных на построении графовых моделей и разного рода структурных схем, до методов, основанных на построении стандартной системы документов - системных спецификаций (см. п. 3.2).

В связи с большой сложностью обследуемых объектов на данной стадии приходится иметь дело с большим разнообразием сочетаний подходов, методов, приемов, средств, выбираемых в зависимости от объекта и конкретных целей и задач исследования. Здесь большое значение имеют квалификация, творческие способности, искусство и талант исследователя или, коротко, его способности в области консалтинга.

**3.1.2 Сущность, роль и место консалтинга при проведении обследования объекта автоматизации и создании АС**

Консалтинг – это деятельность специалиста или целой фирмы, занимающихся стратегическим планированием проекта, анализом и формализацией требований к АС, созданием системного проекта, иногда – проектированием приложений.

В ходе консалтинга выполняются два вида работ:

1. Элементарное наведение порядка в организации: бизнес-анализ и реструктуризация (или даже реинжиниринг бизнес процессов).

2. Собственно системный анализ и проектирование.

Иначе говоря, бизнес-консалтинг – это деятельность, направленная на то, чтобы разобраться в функционировании предприятия, построить соответствующие модели и на их основе выдвинуть некоторые предложения по поводу улучшения работы некоторых звеньев, а еще лучше – бизнес-процессов. Также выявляются и согласовываются требования заказчика, что приводит к пониманию того, что в действительности необходимо сделать. За этим следует проектирование или выбор готовой системы так, чтобы она в итоге как можно в большей степени удовлетворяла требованиям заказчика.

Кроме того, важный элемент консалтинга – формирование и обучение рабочих групп (подробнее см. ниже).

Консалтинг осуществляется до этапа собственно программирования или настройки каких-то уже имеющихся компонентов систем управления (КСУ) предприятием, выбор которых и проводится на основе системного проекта. Тем более в консалтинг не входит системная интеграция.

В конечном итоге все это направлено на будущую автоматизацию и призвано устранить существующий хаос, царящий на многих предприятиях. Одна из основных целей бизнес-консалтинга – избежать автоматизации хаоса.

**3.1.3  Процессный подход к организации деятельности предприятия**

Современное состояние экономики характеризуется переходом от традиционной функциональной модели деятельности компании, построенной на принципах разделения труда, узкой специализации и жестких иерархических управленческих структурах, к процессной модели, основанной на интеграции работ вокруг бизнес-процессов. В связи с этим при проведении консалтинга целесообразно применять процессный подход.

Основной принцип процессного подхода определяет структурирование предприятия в соответствии с его деятельностью и *бизнес-процессами*, а не в соответствии с его организационно-штатной структурой. При этом любое предприятие рассматривается как бизнес-система – система, которая представляет собой связанное множество бизнес-процессов. Под бизнес-процессом понимают совокупность различных видов деятельности, которые создают результат, имеющий ценность для потребителя. Бизнес-процесс – это цепочка работ (функций), результатом которой является какой-либо продукт или услуга. Именно *бизнес-процессы*, обеспечивающие значимый для потребителя результат, представляют ценность и для специалистов, проектирующих АС.

Каждый бизнес-процесс имеет свои границы и роли. При этом одним из основных элементов *процессного подхода* является команда. Подготовка и формирование команд является важной задачей *процессного подхода*. В табл. 3.2 приведено описание ключевых ролей, а в табл. 3.3 – описание типов процессных команд.

Таблица 3.2

Описание ключевых ролей, используемых в *процессном подходе*

|  |  |
| --- | --- |
| Название роли | Описание роли |
| Владелец процесса | Человек, отвечающий за ход и результаты процесса в целом. Он должен знать бизнес-процесс, следить за его выполнением и совершенствовать его эффективность. *Владельцу бизнес-процесса* необходимо обладать коммуникативностью, энтузиазмом, способностью влиять на людей и производить изменения. |
| Лидер команды | Работник, обладающий знаниями о бизнес-процессе и имеющий позитивные личные качества |
| Коммуникатор | Работник, обучающий команду различным методам работы, подготавливающий совместно с *лидером* совещания и анализирующий их результат. |
| Координатор процесса | Работник, отвечающий за согласованную работу всех частей бизнеса и обеспечивающий связь с другими бизнес-процессами. *Координатор* должен обладать административными способностями и пониманием стратегических целей предприятия |
| Участники команды | Специалисты различных уровней иерархии. *Участники команды* получают поддержку и методическое обеспечение от консультанта и *коммуникатора*, вместе с *лидером* проводят моделирование, анализ и оценку бизнес-процесса |

Таблица 3.3

Типы процессных команд

|  |  |
| --- | --- |
| Тип команды | Описание |
| Ситуационная команда | Обычно работает на постоянной основе и выполняет периодически повторяющуюся работу. |
| Виртуальная команда | Создается для разработки нового продукта или услуги. |
| Ситуационный менеджер | Высококвалифицированный специалист, способный самостоятельно выполнить до 90% объема работ |

В табл. 3.4 приведена сводка ряда других важных элементов процессного подхода

Таблица 3.4

Описание элементов процессного подхода

|  |  |
| --- | --- |
| Название | Определение и описание |
| Бизнес-функция | Вид деятельности предприятия |
| Дерево целей | Достижение определенной совокупности целей за счет выполнения бизнес-процессов. Дерево целей имеет, как правило, иерархический вид. Каждая цель имеет свой вес и критерий (количественный или качественный) достижимости. |
| Дерево функций | Множество бизнес-функций, которое представляет иерархическую декомпозицию функциональной деятельности |
| Дерево показателей | Связанные с бизнес-функциями показатели деятельности предприятия, на основании которых строится система показателей оценки эффективности выполнения процессов. *Владельцы процессов* контролируют свои бизнес-процессы с помощью данной системы показателей. Наиболее общими показателями оценки эффективности бизнес-процессов являются: количество производимой продукции заданного качества за определенный интервал времени; количество потребляемой продукции; длительность выполнения типовых операций и др |

Однако чисто "процессная компания" является скорее иллюстрацией правильной организации работ. В действительности все бизнес-процессы компании протекают в рамках организационной структуры предприятия, описывающей функциональные компетентности и отношения.

Управление всей текущей деятельностью компании ведется по двум направлениям — управление функциональными областями, которые поддерживают множество унифицированных бизнес-процессов, разделенных на операции, и управление интегрированными бизнес-процессами, задачей которого является маршрутизация и координация унифицированных процессов для выполнения как оперативных заказов потребителей, так и глобальных проектов самой организации.

Таким образом, процессная ориентация ведет к перестройке организационной структуры, делает организационную структуру компании более "плоской", что иллюстрирует тесную связь между "вертикальным" описанием организации (как структуры распределения ответственности, полномочий и взаимоотношений) и ее "горизонтальным" описанием, как системы процессов.

**3.1.4 Выделение и классификация процессов на предприятии**

Важнейшим шагом при обследовании и структуризации любой компании является выделение и классификация процессов. Классы процессов, на которых целесообразно основываться, перечислены в табл. 3.5.

Таблица 3.5

Классы процессов, на которых целесообразно основываться

|  |  |
| --- | --- |
| № | Описание подпроцесса |
| 1 | основные |
| 2 | управленческие (процессы управления) |
| 3 | обеспечивающие (процессы обеспечения) |
| 4 | сопутствующие |
| 5 | вспомогательные |
| 6 | процессы развития |

Рассмотрим упрощенную модель деятельности компании (рис.3.1), при описании которой используют основные бизнес-процессы, процессы управления, и процессы обеспечения.

**Основные бизнес-процессы** — это процессы, ориентированные на производство товаров и услуг, представляющие ценность для клиента и обеспечивающие получение дохода.

Рис. 3.1  Упрощенная модель деятельности компании

Основные процессы образуют "жизненный цикл" продукции компании. Все они описываются по производственно-коммерческим цепочкам: "первичное взаимодействие с клиентом и определение его потребностей реализация запроса послепродажное сопровождение и мониторинг удовлетворения потребностей". Эти этапы цепочки достаточно стандартны. Процесс "реализации (запроса клиента)" может быть декомпозирован на подпроцессы (табл. 3.6).

Таблица 3.6

Подпроцессы процесса реализации запроса клиента

|  |  |
| --- | --- |
| № | Описание подпроцесса |
| 1 | Разработка (проектирование) продукции |
| 2 | Закупка (товаров, материалов, комплектующих изделий) |
| 3 | Транспортировка (закупленного) |
| 4 | Разгрузка, приемка на склад и хранение (закупленного) |
| 5 | Производство (со своим технологическим циклом и внутренней логистикой) |
| 6 | Приемка на склад и хранение (готовой продукции) |
| 7 | Отгрузка (консервация и упаковка, погрузка, доставка) |
| 8 | Пуско-наладка |
| 9 | Оказание услуг (предусмотренных контрактом на поставку или имеющих самостоятельное значение) и т.п. |

Для оценки этапов работы с любым документом можно использовать также анализ "жизненного цикла документа", который может выглядеть так, как  показано в табл. 3.7.

Таблица 3.7

Подпроцессы процесса работы с документом

|  |  |
| --- | --- |
| № | Описание подпроцесса |
| 1 | предоставление исходных данных |
| 2 | подготовка, разработка |
| 3 | заполнение |
| 4 | корректировка |
| 5 | оформление |
| 6 | подписание |
| 7 | контроль соответствия установленным требованиям |
| 8 | визирование |
| 9 | согласование |
| 10 | утверждение |
| 11 | акцептирование (принятие к сведению, использование) |
| 12 | хранение |
| 13 | снятие копии (копирование) |

***Процессы управления*** – это процессы, охватывающие весь комплекс функций управления на уровне каждого бизнес-процесса и бизнес-системы в целом. *Процессы управления* имеют своей целью выработку и принятие управленческого решения. Данные управленческие решения и, соответственно, функции управления можно классифицировать по уровню объекта управления или по элементам управленческого цикла (табл. 3.8).

Таблица 3.8

Варианты классификации процессов (функций) управления

|  |  |
| --- | --- |
| № | Описание подпроцесса |
| Классификация по уровню объекта управления | |
| 1 | стратегическое управление (решения относительно всей организации в целом) |
| 2 | организационное проектирование (структуризация) |
| 3 | маркетинг |
| 4 | финансово-экономическое управление |
| 5 | логистика и организация процессов |
| 6 | менеджмент качества |
| 7 | персонал (и другие отдельные функциональные области или отдельные процессы организации) |
| Классификация по элементам управленческого цикла | |
| 1 | сбор информации |
| 2 | выработка решения |
| 3 | реализация |
| 4 | контроль |
| 5 | учет |
| 6 | анализ |
| 7 | регулирование |

В табл. 3.9 приведены наиболее часто встречающиеся варианты детализации управленческого цикла.

Таблица 3.9

Варианты детализации управленческого цикла

|  |  |
| --- | --- |
| № | Элемент цикла (этап) |
| 1 | **сбор информации:** |
| 1.1 | определение состава собираемой информации; |
| 1.2 | определение форм отчетности. |
| 2 | **выработка решения:** |
| 2.1 | анализ альтернатив; |
| 2.2 | подготовка вариантов решения; |
| 2.3 | принятие решения; |
| 2.4 | выработка критериев оценки. |
| 3 | **реализация:** |
| 3.1 | планирование; |
| 3.2 | организация; |
| 3.3 | мотивация; |
| 3.4 | координация. |
| 4 | **контроль исполнения** |
| 5 | **учет результатов** |
| 6 | **анализ:** |
| 6.1 | сравнение по принятым критериям; |
| 6.2 | анализ дополнительной информации; |
| 6.3 | диагностика возможных причин отклонений. |
| 7 | **регулирование:** |
| 7.1 | регулирование на уровне реализации (возврат к п.3); |
| 7.2 | регулирование на уровне выработки решения (возврат к п.2.2). |

Каждый из этих этапов имеет своих характерных для него исполнителей — управленцев, которых можно отнести к трем основным категориям:

·   руководитель (ответственный за принятие и организацию выполнения решений);

·   специалист-аналитик (ответственный за подготовку решения и анализ отклонений);

·   технические исполнители (сбор информации, учет, коммуникации).

***Процессы обеспечения*** – это процессы, предназначенные для жизнеобеспечения основных, управленческих и сопутствующих процессов и ориентированные на поддержку их универсальных средств. Например, процесс финансового обеспечения, *процесс обеспечения* кадрами, процесс юридического обеспечения — это вторичные процессы. Они создают и поддерживают необходимые условия для выполнения основных функций и функций менеджмента. Клиенты обеспечивающих процессов находятся внутри компании.

Например, можно выделить следующие стандартные *процессы обеспечения*, указанные в табл. 3.10.

Таблица 3.10

Стандартные процессы обеспечения на верхнем уровне детализации

|  |  |
| --- | --- |
| № | Название процесса |
| 1 | обеспечение производства |
| 2 | техобслуживание и ремонт оборудования |
| 3 | обеспечение теплоэнергоресурсами |
| 4 | обслуживание и ремонт зданий и сооружений |
| 5 | технологическое обеспечение |
| 6 | метрологическое обеспечение |
| 7 | техника безопасности |
| 8 | экологический контроль и т.п. |
| 9 | обеспечение управления |
| 10 | информационное обеспечение |
| 11 | обеспечение документооборота |
| 12 | коммуникационное обеспечение |
| 13 | юридическое обеспечение |
| 14 | обеспечение безопасности |
| 15 | материально-техническое обеспечение управления |
| 16 | хозяйственное обеспечение |
| 17 | обеспечение коммунальными услугами |
| 18 | транспортное обслуживание и т.п. |

Для каждого из выделенных выше подпроцессов также следует определить, какой основной или управленческий процесс является потребителем этих "внутренних" услуг.

Разбиение данных процессов производится по индивидуальным технологическим цепочкам. Многие из обеспечивающих процессов стандартны для всех компаний или определенных сфер деятельности: промышленность, торговля, предоставление услуг и т.п..

**3.1.5 Сущность реинжиниринга объектов автоматизации**

**3.1.5.1 Понятие «реинжиниринг бизнеса» и его отличие от совершенствования бизнеса**

Инжиниринг бизнеса — это набор приемов и методов, которые компания использует для проектирования и реализации бизнеса в соответствии со своими целями.

Реинжиниринг — это радикальное переосмысление и перепроектирование фундаментальных деловых процессов для достижения резких, скачкообразных улучшений главных современных показателей деятельности компании, таких, как стоимость, качество, сервис и темпы (термин «реинжиниринг» ввел М. Хаммер).

Это определение содержит четыре ключевых слова: «фундаментальный», «радикальный», «резкий (скачкообразный)» и «процесс» (наиболее важное слово), поясняемых табл. 3.11.

Таблица 3.11

Ключевые слова в определении реинжиниринга

|  |  |
| --- | --- |
| Слово | Пояснение |
| Фундамен-тальный | На начальной стадии реинжиниринга необходимо ответить на такие основные вопросы:     почему компания делает то, что она делает?     почему компания делает это таким способом?     какой хочет стать компания?  Отвечая на эти вопросы, специалисты должны переосмыслить текущие правила и положения (зачастую не сформулированные в письменной форме) ведения бизнеса и часто оказывающиеся устаревшими, ошибочными или неуместными. |
| Радикаль-ный | Радикальное перепроектирование — это изменение всей существующей системы, а не только поверхностные преобразования, т.е. в ходе радикального перепроектирования предлагаются совершенно новые способы выполнения работы. |
| Резкий (скачкооб-разный) | Реинжиниринг не применяется, если необходимо улучшение либо увеличение показателей деятельности компании на 10—100%, т.е. могут использоваться более традиционные методы (от произнесения речей перед сотрудниками до проведения программ повышения качества), применение которых не сопряжено со значительным риском. Реинжиниринг целесообразен только в тех случаях, когда требуется достичь резкого (скачкообразного) улучшения показателей деятельности компании (500—1000% и более) путем замены старых методов управлении новыми. |
| Процесс | Бизнес-процесс — это множество «внутренних шагов» предприятия, заканчивающихся созданием продукции, необходимой потребителю. Назначение каждого бизнес-процесса состоит в том, чтобы предложить потребителю продукцию (услугу), удовлетворяющую его по стоимости, сервису и качеству. Итак, бизнес-процесс — это действия по достижению цели компании. При этом оптимизируется результативность бизнес-процесса путем его организации на основе упорядочения горизонтальных связей в структуре управления компанией. |

Различия между совершенствованием и реинжинирингом бизнеса представлены в. табл. 3.12

Таблица 3.12

Сравнительная характеристика совершенствования

и реинжиниринга бизнеса

|  |  |  |
| --- | --- | --- |
| Параметр | Совершенствование | Реинжиниринг |
| Уровень изменений | Наращиваемый | Радикальный |
| Начальная точка | Существующий процесс | «Чистая доска» |
| Частота изменений | Непрерывно/единовременно | Единовременно |
| Длительность изменений | Малая | Большая |
| Направление изменений | Снизу вверх | Сверху вниз |
| Охват | Узкий — на уровне функций (функциональный подход) | Широкий — межфункциональный |
| Риск | Умеренный | Высокий |
| Основное средство | Стратегическое управление | Информационные технологии |
| Тип изменений | Изменение корпоративной культуры | Культурный/структурный |

Можно выделить три типа компаний, для которых реинжиниринг необходим и целесообразен (табл. 3.13).

Таблица 3.13

Типы компаний для реинжиниринга

|  |  |
| --- | --- |
| Тип | Пояснение |
| 1 | Компании, находящиеся на грани краха в связи с тем, что цены на товары заметно выше и (или) их качество (сервис) заметно ниже, чем у конкурентов. Если эти компании не предпримут решительных шагов, они неизбежно разорятся |
| 2 | Компании, не имеющие в текущий момент затруднений, но предвидящие неизбежность возникновения трудноразрешимых проблем, связанных, например, с появлением новых конкурентов, изменением требований клиентов, изменением экономического окружения и пр. |
| 3 | Компании, не имеющие проблем сейчас, не прогнозирующие их в обозримом будущем. Это компании-лидеры, проводящие агрессивную маркетинговую политику, не удовлетворяющиеся хорошим текущим состоянием и желающие с помощью реинжиниринга добиться лучшего. |

**3.1.5.2 Условия успешного реинжиниринга и факторы риска**

Проведение реинжиниринга связано с перепроектированием бизнес-процессов. Существуют определенные требования, которым все они должны отвечать, поэтому можно выделить следующие принципы организации бизнес-процессов, формируемых в ходе проведения реинжиниринга (табл. 3.14).

Таблица 3.14

Принципы перепроектирования бизнес-процессов

|  |  |
| --- | --- |
| № | Принцип и пояснение |
| 1. | *Интегрирование бизнес-процессов.* Наиболее характерное свойство перепроектированных процессов — отсутствие сборочных конвейеров как способа координации работы персонала с относительно простыми трудовыми функциями. При выполнении сложных трудовых функций требуется иная организация работ. На практике, конечно, не всегда удается свести все этапы процесса к работе, выполняемой одним человеком. В этом случае создается команда, которая несет ответственность за данный процесс. Возможны сбои и ошибки, но потери будут значительно меньше, чем при традиционной организации работ. |
| 2. | *Горизонтальное сжатие бизнес-процессов* Сравнительные оценки, выполненные компаниями, которые провели реинжиниринг, показывают, что переход от традиционной организации работ к выполнению процесса одним человеком позволяет снизить численность персонала и ускорить выполнение процесса примерно в 10 раз. Уменьшается количество ошибок и отпадает необходимость держать специалистов для устранения этих ошибок. За счет уменьшения численности работающих и четкого распределения ответственности между ними улучшается управляемость. |
| *3.* | *Децентрализация ответственности (вертикальное сжатие бизнес-процессов).* Исполнители принимают самостоятельные решения в случаях, в которых раньше они традиционно должны были обращаться к руководству. |
| *4.* | *Логика реализации бизнес-процессов.* Линейное выполнение работ заменяется логическим порядком (т.е. часто работы осуществляются параллельно). Это экономит время, которое тратилось на взаимоувязку работ на разных участках. |
| *5.* | *Диверсификация бизнес-процессов.* Существуют различные варианты процессов выполнения. Традиционный процесс, ориентированный на производство массовой продукции, должен выполняться одинаково для всех входов, приводя к согласованным выходам. Традиционные процессы обычно оказываются очень сложными, так как они весьма детализированные и во многом рассчитаны на исключения и частные случаи. |
| *6.* | *6. Разработка различных версий бизнес-процессов* в условиях постоянно меняющегося рынка необходима, чтобы процессы имели различные варианты в зависимости от ситуаций, входов и состояния рынка. Новые процессы, имеющие различные версии, начинаются с проверочного шага, на котором определяется, какая версия процесса наиболее подходит для текущей ситуации. Поэтому новые процессы в отличие от традиционных проще и понятнее, так как каждый вариант ориентирован только на одну, соответствующую ему ситуацию. |
| *7.* | *Рационализация горизонтальных связей.* Создание линейных функциональных подразделений. Работа выполняется в том месте, где это наиболее целесообразно. Раньше в компаниях работа была организована по «тематическому» принципу в соответствующих подразделениях: расчетный отдел, транспортный отдел, отдел снабжения и т.д., поэтому если расчетному отделу требовались карандаши, то он обращался в отдел снабжения с заявкой. Этот отдел находил производителя, догова-ривался о цене, размещал заказ, осматривал товар, оплачивал его и передавал в рас-четный отдел. Этот процесс длителен и неэкономичен. Анализ, проведенный в одной из компаний, показал, что затраты на приобретение батарейки за 3 дол. составили 100 дол. При реинжиниринге чаще всего создаются горизонтальные управленческие связи между подразделениями. Это позволяет устранить излишнюю интеграцию. |
| *8.* | *Рационализация управленческого воздействия.* Речь идет об уменьшении числа проверок и снижении степени управленческого воздействия, которые не приводят непосредственно к получению материальных ценностей. Поэтому задача реинжиниринга — осуществлять их только в той мере, в которой это экономически целесообразно. |
| *9.* | *Культура решения задачи.* Предполагается минимизация согласований, так как они тоже не имеют материальной ценности. Задача реинжиниринга — минимизировать согласования в ходе исполнения процесса путем сокращения внешних контактов. |
| *10.* | *Рационализация связей «компания — заказчик».* Совершенствование оргструктуры фирмы должно создать условия, при которых уполномоченный менеджер обеспечивает единый канал связей. |
| *11.* | *Уполномоченный менеджер.* Этот принцип применяется в тех случаях, когда шаги процесса либо сложны, либо распределены таким образом, что их интеграция силами небольшой команды невозможна. Уполномоченный менеджер является буфером между сложным процессом и заказчиком. Менеджер во взаимоотношениях с заказчиком выступает ответственным за весь процесс. Чтобы сыграть эту роль, менеджер должен быть способен отвечать на вопросы заказчика и решать его проблемы. Содержание задачи обусловливает необходимость обеспечения доступа менеджера ко всем информационным системам, используемым в этом процессе, а также к его исполнителям. |
| *12.* | *Сохранение положительных моментов централизации управления.* На практике это достигается путем совершенствования информационного обеспечения дивизиональной организации управления. Современные ИТ дают возможность подразделениям компании действовать автономно, сохраняя возможность пользования централизованными данными. Таким образом, компания может устранить бюрократические региональные структуры, необходимые для обслуживания территориально разобщенной клиентуры, и одновременно повысить качество обслуживания. |

Экспертные оценки показывают, что до сих пор около 50% проектов реинжиниринга заканчиваются неудачей. Специальные исследования позволили выявить ряд факторов, которые оказывают существенное влияние на процесс реинжиниринга, определяя условия его проведения и соответствующие риски (т.е. возможность успеха или неудачи). Эти факторы представлены в табл.. 3.15.

Таблица 3.15

Факторы, влияющие на реинжиниринг бизнес-процессов

|  |  |
| --- | --- |
| № | Фактор и пояснение |
| 1. | *Мотивация.* Мотив осуществления проекта реинжиниринга должен быть ясно определен и зафиксирован. При этом высшее руководство должно быть абсолютно убеждено, что этот проект действительно даст значительный результат, и понимать, что полученный результат вызовет изменение структуры компании. Чтобы обеспечить успех, руководство должно верить в необходимость реинжиниринга, проводимого в масштабах всей компании, и предоставить в распоряжение команды по реинжинирингу лучшие силы. |
| 2. | *Руководство.* Проект должен выполняться под управлением руководителей компании; руководитель, возглавляющий проект реинжиниринга, должен иметь большой авторитет и нести за него ответственность. Для успеха проекта очень важно твердое и умелое управление. Руководитель проекта должен понимать, что возникнут трудности, неизбежные при построении новой компании: он должен сопротивляться «давлению» старых порядков и убедить своих сотрудников в том, что проект не только выполним, но и необходим для выживания компании. Он обязан прилагать все усилия для продвижения проекта и своевременного его завершения. |
| 3. | *Сотрудники.* В команде, выполняющей проект реинжиниринга и контролирующей его проведение, необходимо участие сотрудников, наделенных соответствующими полномочиями и способных создать атмосферу сотрудничества. Сотрудники должны понимать, почему проект приведен в действие (другими словами, они должны оценивать проблемы, которые мешают бизнесу), принимать свои новые обязанности, быть способными выполнять их, посвящать реинжинирингу необходимое время и обоснованно двигаться к успеху. По сути, все работники должны освоить и устойчиво реализовывать новый набор образцов поведения. Опыт показывает, что относительно просто объяснить новый способ работы персоналу нижнего уровня, но людям, занимающим должности менеджеров, намного труднее понять то, что предлагает новая компания. Группа, на которую следует обратить особое внимание специалистов,— менеджеры среднего уровня. |
| 4. | *Коммуникации.* Новые задачи компании должны быть четко сформулированы и доведены до каждого сотрудника в понятном ему виде. Успешность реинжиниринга зависит от того, насколько руководство и рядовые сотрудники понимают, как достичь стратегических целей компании. |
| 5. | *Бюджет.* Проект должен иметь свой бюджет, особенно если планируется интенсивное использование ИТ. Часто ошибочно считают, что реинжиниринг возможен на условиях самофинансирования. Поэтому реинжиниринг нужно рассматривать как венчурный по характеру проект. |
| 6. | *Технологическая поддержка.* Для проведения работ по реинжинирингу необходима поддержка — соответствующие методики и инструментальные средства. Реинжиниринг обычно включает в себя построение информационной системы для поддержки нового бизнеса. |
| 7. | *Консультации.* Эксперты (консультанты) могут оказать существенную помощь исполнителям, впервые осуществляющим реинжиниринг. Важно, чтобы консультанты исполняли поддерживающую, а не управляющую роль, и не входили в штат компании. Поэтому руководитель проекта реинжиниринга должен быть грамотным заказчиком услуг консультантов. К факторам, способствующим успеху реинжиниринга, можно отнести и такие, как оценка и контроль рисков, четкое определение ролей и обязанностей исполнителей, нацеливание их на достижение осязаемых результатов. |

Риск реинжиниринга достаточно велик, однако причины неудач заключаются не в загадочности реинжиниринга, а в нарушении правил его проведения. Американские исследователи М. Хаммер и Дж. Чампи указывают, что с точки зрения риска реинжиниринг подобен игре в шахматы, а не в рулетку, т.е. участники реинжиниринга, как игроки в шахматы, в меру своих знаний и умения могут влиять на результат. Иными словами, величину результата невозможно гарантировать. Главное в стратегии управления реинжинирингом — избегать глобальных ошибок.

При проведении реинжиниринга встречаются следующие характерные ошибки, представленные в табл. 3.16.

Таблица 3.16

Ошибки при проведении реинжиниринга бизнес-процессов

|  |  |
| --- | --- |
| № | Ошибка и пояснение |
| 1. | *Компания пытается улучшить существующий процесс вместо того чтобы перепроектировать его.* |
| 2. | *Несистемный подход к обновлению.* Компании концентрируются только на перепроектировании процессов, игнорируя все остальное. |
| 3. | *Неправильная оценка уровня корпоративной культуры компании.* Для того чтобы персонал успешно выполнял перепроектированные процессы, он должен иметь побудительные причины, |
| 4. | *Непоследовательность освоения новации.* Заключается в преждевременном завершении реинжиниринга и ограниченной постановке задачи. |
| 5. | *Нерациональное распределение задач по освоению инновации.* Попытки осуществить реинжиниринг не сверху вниз, а снизу вверх, не могут быть успешно завершены менеджерами нижнего и среднего уровни |
| 6. | *Недостаточное ресурсное обеспечение инновации.* Существенное повышение эффективности деятельности компании, являющееся следствием реинжиниринга, невозможно без значительных инвестиций в программу его проведения. |
| 7. | *Планирование момента начала мотивации.* Например, шансы на успешный реинжиниринг заметно снижаются, если известно, что исполнительный директор компании через год или два уходит в отставку. |
| 8. | *Личностные проблемы обновления.* Попытка провести реинжиниринг, не ущемив ничьих прав, не может привести к положительному результату. Выражение «нельзя приготовить омлет, не разбив яиц» весьма точно отражает суть реинжиниринга. |

Компания, руководители которой понимают основы реинжиниринга и привержены его идее, имеет практически стопроцентный шанс добиться успеха.

**3.2 Краткий обзор формализованных методов обследования и анализаобъекта автоматизации**

**3.2.1  Методы, основанные на построении графовых моделей**

**3.2.1.1 Графовая модель структурно-функциональных связей**

Модель используется при анализе организационной и функциональной структур, а также информационных потоков и документооборота. Объекты анализируемой системы представляются в виде графовой модели (схемы) структурно-функциональных связей, реально существующих в системе на данный период. При этом показываются основные материальные и информационные потоки с указанием их параметров. Детализация представления определяется рамками задач, охватываемых разрабатываемой автоматизированной системой.

Модель выполняется в виде стратифицированного ориентированного графа. Страта – уровень или слой. На графе выделяют (рис. 3.2):

Рис. 3.2  Графовая модель структурно-функциональных связей

1.    Производственную страту, на которой отображаются основные производственно-технические подразделения объекта, связанные материальными потоками;

2.    Управленческую страту – основные управленческие подразделения различных иерархических уровней;

3.    Страту поставщиков и заказчиков;

4.    Руководство своей и взаимодействующими с ней организациями основных поставщиков и заказчиков с показом информационного взаимодействия с главным ИВЦ и основными управленческими подразделениями.

Указанные подразделения отображаются вершинами ориентированного графа. Вершины связаны между собой дугами, отображающими материальные потоки (двойные линии) и информационные потоки (одинарные линии).

Управление материальными потоками при возникновении различных ситуаций осуществляется информационными потоками через узлы принятия решений, которые изображаются на материальных потоках в виде вентилей.

Достоинства модели:

1.      Наглядность отображения связей, управляющих воздействий и видов информации;

2.      Позволяет путем эвристических имитационных экспериментов выявить недостатки, узкие места существующей организации системы управления и на основе этих экспериментов наметить и отработать предложения по изменению в структурно-функциональной схеме организации в условиях автоматизации.

Недостаток: громоздкость для систем большой размерности.

**3.2.1.2 Метод сетевого моделирования**

Базируется на традиционных моделях сетевого анализа с построением сетевых графиков. При этом под ***событием*** понимается определенный документ, являющийся конечным событием, если он представляет собой результат выполнения какой-либо работы, или начальными, если будут использоваться в дальнейшем ходе выполнения работ. Под ***работой*** понимается определенная задача или функция, выполняемая сотрудниками органа управления.

**3.2.1.3 Графоаналитический метод**

Метод основан на анализе матриц смежности *и*достижимостиинформационного графа. Исходными данными являются сведения о парных отношениях между наборами информационных элементов.

***Информационные элементы*** – различные типы входящих, выходящих и промежуточных данных.

Под ***матрицей смежности B*** понимают квадратную бинарную матрицу, проиндексированную по обеим координатным осям множеством информационных  элементов  *D*{*d1, d2,. d3,…, dn*}.  В позиции *(i,j)* такой матрицы записывается *1*, если для получения элемента*d*j требуется обращение непосредственно к элементу *di.*

***Матрицей достижимости M*** называют квадратную бинарную матрицу, подобную   матрице смежности, но отличающуюся тем, чтозапись *1* в позицию *(i,j)* матрицы соответствует наличию для упорядоченной пары информационных элементов *(di,dj)* смыслового отношения достижимости, означающего, что в процессе получения элемента *dj* используется значение элемента *di* (не обязательно непосредственно).

В результате анализа описанных матриц и соответствующих им графов можно выделить основные этапы обработки данных, их последовательность и циклы обработки на каждом уровне.

**3.2.2 Методы оперограмм  и структурно-информационно- временных схем**

***Метод оперограмм*** позволяет представить изучаемый процесс работы с информацией в виде цепей состояния документа, поступающего с одной операции на другую, и операций, отображенных в схеме между этими состояниями.

***Метод структурно-информационо-временных схем (СИВС)*** является развитием метода оперограмм. Он предполагает построение схемы в виде прямоугольной сетки, разбитой на три горизонтальные зоны (рис. 3.3).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Подразделения и организации | ВРЕМЯ | | | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Рис. 3.3 Построение схемы по методу СИВС

Основной частью СИВС является средняя зона, в которой по строкам перечислены все структурные подразделения изучаемой организации. По строкам верхней зоны перечислены вышестоящие организации, с которыми изучаемая организация связана при выполнении своих функций. По строкам нижней зоны – подчиненные организации, а также поставщики исходных материалов и потребители продукции.

В левой части схемы выделен столбец для записи наименований подразделений и организаций. В оставшейся части схемы, используя стандартные условные обозначения, составляют схему материальных потоков.

Слева направо показано движение во времени или по ходу технологического процесса. В зависимости от степени детализации время откладывается в часах, днях, неделях и т.д. Строгий масштаб времени не обязателен. Просто операции, изображенные на схеме правее, происходят позже, чем помещенные левее.

Схему рисуют сначала очень укрупненной, постепенно (по мере надобности) дополняя и детализирую. В дальнейшем на схеме аналогичным образом отображают потоки информации.

Сопоставление информационных потоков с материальными позволяет проанализировать их синхронность, выявить излишние процедуры, связанные со сбором, движением и переработкой информации. При глубоком анализе СИВС можно выявить степень использования данных, действительную необходимость их передачи в то или иное подразделение, количественные характеристики информационных потоков и др.

Представляя собой простое и наглядное отображение рабочего процесса, СИВС позволяет всем участникам проектирования АС достаточно быстро и легко ознакомится с деятельностью объекта.

**3.2.3  Сущность методов разработки спецификаций**

Под ***спецификацией*** понимают достаточно полное и точное описание объекта исследований и соответствующей задачи автоматизации, которую необходимо решить. Спецификация представляет собой описание в терминах, характерных для задачи, а не для ее реализации и служит основой дальнейшей детализации и разработки.

В отличие от вышеописанных классических методов, которые можно назвать традиционными, методы данной группы являются сравнительно новыми и интенсивно развиваются.

На основе анализа спецификаций разрабатывается структура АС в целом и ее важнейших обеспечивающих подсистем (в первую очередь программное обеспечение). Хорошие спецификации обеспечивают построение требуемых программ, их проверку и модификацию, сокращая затраты на разработку и сопровождение.

В спецификации выделяют *две части*: функциональную и эксплуатационную.

***Функциональная часть*** описывает объекты, участвующие в задаче, деление задачи на подзадачи, входные и выходные данные, связи между ними, процессы и действия, реакции на исключительные ситуации и т.д.

***Эксплуатационная часть*** определяет скорость решения задачи, используемые ресурсы, характеристики требуемой аппаратуры, требования надежности и безопасности решения задачи и т.д.

Языки спецификации делятся на два класса: графические и текстовые.

***Графический язык*** представляет спецификацию в виде графов и диаграмм (например, языки методов  структурного анализа SA, операционно-графического метода HIPO, метода объектно-ориентированного моделирования и др.)

***Текстовый язык*** представляет спецификацию в виде последовательности символов (псевдокоды, PSL, PDL и др.).

Разработка спецификаций –  это самостоятельная область технологий и методологий проектирования систем вообще и их программирования, в частности. Сами спецификации стали важной частью системной и программной документации.

Более детально методы этой группы рассмотрены в следующих разделах курса.

***3.3 Анализ и моделирование функциональной области внедрения АС***

**3.3.1  Основные понятия организационного бизнес-моделирования**

На практике при проведении организационного анализа наибольшее распространение получил инжиниринговый подход. Организационный анализ компании при таком подходе проводится по определенной схеме с помощью *полной бизнес-модели компании*. Компания рассматривается как целеустремленная, открытая, социально-экономическая система, принадлежащая иерархической совокупности открытых внешних надсистем (рынок, государственные учреждения и пр.) и внутренних подсистем (отделы, цеха, бригады и пр.). Возможности компании определяются характеристиками ее структурных подразделений и организацией их взаимодействия. На рис. 3.4 представлена обобщенная схема организационного бизнес-моделирования, а в табл. 3.17 – определены элементы этой схемы.

Рис. 3.4  Обобщенная схема организационного бизнес- моделирования

Построение *бизнес-модели компании* начинается с описания модели взаимодействия с внешней средой по закону единства и борьбы противоположностей, и с определения *миссии компании*.

Таблица 3.17

Определение элементов схемы организационного бизнес-моделирования

|  |  |
| --- | --- |
| Элемент схемы | Определение |
| ***Миссия*** | - деятельность, осуществляемая предприятием для того, чтобы выполнить функцию, для которой оно было учреждено,  - механизм, с помощью которого предприятие реализует свои цели и задачи |
| ***Дерево целей*** | Иерархические списки уточнения и детализации миссии |
| ***Дерево стратегий*** | Иерархические списки уточнения и детализации способов достижения целей |
| Б***изнес-потенциал*** | Набор видов коммерческой деятельности, направленный на удовлетворение потребностей конкретных сегментов рынка |
| ***функционал компании*** | перечень бизнес-функций, функций менеджмента и функций обеспечения, требуемых для поддержания на регулярной основе указанных видов коммерческой деятельности |
| ***Матрица проекций*** | Модель, представленная в виде матрицы, задающей систему отношений между классификаторами в любой их комбинации |
| ***Матрица коммерческой ответственности*** | Закрепляет ответственность структурных подразделений за получение дохода в компании от реализации коммерческой деятельности. Ее дальнейшая детализация (путем выделения центров финансовой ответственности) обеспечивает построение финансовой модели компании, что, в свою очередь, позволяет внедрить систему бюджетного управления |
| ***Матрица функциональной ответственности*** | Закрепляет ответственность структурных звеньев (и отдельных специалистов) за выполнение бизнес-функций при реализации процессов коммерческой деятельности (закупка, производство, сбыт и пр.), а также функций менеджмента, связанных с управлением этими процессами (планирование, учет, контроль в области маркетинга, финансов, управления персоналом и пр.). Дальнейшая детализация матрицы (до уровня ответственности отдельных сотрудников) позволит получить функциональные обязанности персонала, что в совокупности с описанием прав, обязанностей, полномочий обеспечит разработку пакета должностных инструкций |
| ***Процессные потоковые модели*** | Модели, описывающие процесс последовательного во времени преобразования материальных и информационных потоков компании в ходе реализации какой-либо бизнес-функции или функции менеджмента. Сначала (на верхнем уровне) описывается логика взаимодействия участников процесса, а затем (на нижнем уровне) - технология работы отдельных специалистов на своих рабочих местах. |
| ***Модель структур данных*** | Определяет перечень и форматы документов, сопровождающих процессы в компании, а также задает форматы описания объектов внешней среды, компонентов и регламентов самой компании. При этом создается система справочников, на основании которых получают пакеты необходимых документов и отчетов. |

Определение *миссии* позволяет сформировать ***дерево целей компании***, которое в свою очередь формирует ***дерево стратегий***. Их анализ дает возможность сформировать б***изнес-потенциал компании.*** При этом исходя из специфики каналов сбыта, формируется первоначальное представление об организационной структуре (определяются центры коммерческой ответственности). Возникает понимание основных ресурсов, необходимых для воспроизводства товарной номенклатуры.

*Бизнес-потенциал*, в свою очередь, определяет ***функционал компании.*** Кроме того, уточняются необходимые для реализации функций ресурсы (материальные, человеческие, информационные) и структура компании.

Построение *бизнес-потенциала* и *функционала компании* позволяет с помощью *матрицы проекций* определить зоны ответственности менеджмента:***матрицу коммерческой ответственности***и***матрицу функциональной ответственности***

Описание *бизнес-потенциала*, *функционала* и соответствующих матриц ответственности представляет собой **статическое описание компании**. При этом процессы, протекающие в компании пока в свернутом виде (как функции), идентифицируются, классифицируются и, что особенно важно, закрепляются за исполнителями (будущими хозяевами этих процессов).

На этом этапе бизнес-моделирования формируется общепризнанный набор основополагающих внутрифирменных регламентов (табл. 3.18).

Таблица 3.18

Набор внутрифирменных регламентов

|  |  |
| --- | --- |
| № | Набор регламентов |
| 1 | Базовое Положение об организационно-функциональной структуре компании |
| 2 | Пакет Положений об отдельных видах деятельности (финансовой, маркетинговой и т.д.) |
| 3 | Пакет Положений о структурных подразделениях (цехах, отделах, секторах, группах и т.п.) |
| 4 | Должностные инструкции |

Это вносит прозрачность в деятельность компании за счет четкого разграничения и документального закрепления зон ответственности менеджеров.

Дальнейшее развитие (детализация) бизнес-модели происходит на этапе динамического описания компании на уровне ***процессных потоковых моделей***. Эти модели отвечают на вопросы **что—кто—как—кому**.

Процессная модель компании должна строиться с учетом следующих положений.

Верхний уровень модели должен отражать только контекст диаграммы – взаимодействие представляемого единственным контекстным процессом предприятия с внешним миром.

На втором уровне должны быть отражены тематически сгруппированные  по видам деятельности предприятия *бизнес-процессы* и их взаимосвязи. Детализация *бизнес-процессов* осуществляется посредством бизнес – функций, которые состоят из бизнес–операций. Описание элементарной бизнес–операции осуществляется с помощью миниспецификации.

*Процессный подход* требует комплексного изучения различных сторон жизни организации. В результате создается модель деятельности "**как есть**" (англ. "**As is**"). В табл. 3.19 показано, что позволяет анализ этой модели.

Таблица 3.19

Перечень того, что позволяет анализ модели деятельности "как есть"

|  |  |
| --- | --- |
| № | Анализ модели позволяет: |
| 1 | проверить, насколько деловые процессы рациональны |
| 2 | определить, является ли та или иная операция ориентированной на общественно значимый конечный результат или излишней бюрократической процедурой |
| 3 | установить адреса *владельцев* деловых процессов |
| 4 | подойти к обоснованию стандартов выполнения деловых процессов, допустимых рисков и диапазонов свободы принятия решений исполнителями, предельных нормативов затрат ресурсов на единицу эффекта |

Фактически, основной задачей организационного проектирования является выбор оптимального соотношения между эффективностью использования ресурсов и эффективностью процессов. Жесткая специализация подразделений экономит ресурсы организации, но снижает качество реализации процессов. Создание "процессных" команд, включающих собственных специалистов по всем ключевым операциям, обходится достаточно дорого, но при этом значительно сокращается время и повышается точность выполнения процесса.

Завершается организационное бизнес-моделирование разработкой **модели структур данных**.

Такой подход позволяет описать деятельность компании с помощью универсального множества управленческих регистров (цели, стратегии, продукты, функции, организационные звенья и др.).

Управленческие регистры по своей структуре представляют собой иерархические классификаторы. Объединяя классификаторы в функциональные группы и закрепляя между собой элементы различных классификаторов с помощью матричных проекций, можно получить **полную бизнес-модель компании** (рис. 3.5).

При этом происходит процессно-целевое описание компании, позволяющее получить взаимосвязанные ответы на следующие вопросы: **зачем-что-где-кто-как-когда-кому-сколько-в каком виде** (рис. 3.6).

Рис. 3.5  Полная бизнес-модель компании

Рис. 3.6  Основные этапы процессно-целевого описания компании

В целом организационный анализ предполагает построение комплекса взаимосвязанных информационных моделей компании, описанных в табл. 3.20.

Таблица 3.20

Определение элементов комплекса взаимосвязанных информационных моделей компании

|  |  |
| --- | --- |
| Элемент | Определение |
| ***Полная бизнес-модель компании*** | Совокупность функционально ориентированных информационных моделей, обеспечивающая взаимосвязанные ответы на следующие вопросы: "зачем" - "что" - "где" - "кто" - " |
| **Стратегическая модель целеполагания** | Отвечает на вопросы: зачем компания занимается именно этим бизнесом, почему предполагает быть конкурентоспособной, какие цели и стратегии для этого необходимо реализовать |
| **Организационно-функциональная модель** | Отвечает на вопрос кто-что делает в компании и кто за что отвечает |
| **Функционально-технологическая модель** | Отвечает на вопрос что-как реализуется в компании |
| **Процессно-ролевая модель** | Отвечает на вопрос кто-что-как-кому |
| **Количественную модель** | Отвечает на вопрос сколько необходимо ресурсов |
| **Модель структуры данных** | Отвечает на вопрос в каком виде описываются регламенты компании и объекты внешнего окружения |

Представленная совокупность моделей обеспечивает необходимую полноту и точность описания компании и позволяет вырабатывать понятные требования к проектируемой АС.

**3.3.2  Шаблоны организационного бизнес-моделирования**

**3.3.2.1 Шаблон разработки миссии**

Технология организационного бизнес-моделирования предполагает использование типовых шаблонных техник описания компании.

Как было сказано выше, любая компания с ее микро- и макроокружением представляет собой иерархию вложенных друг в друга открытых, субъектно-ориентированных систем. Компания, с одной стороны, является частью рынка, а с другой отстаивает в конкурентной борьбе собственные интересы. ***Миссия*** представляет собой результат позиционирования компании среди других участников рынка. Поэтому *миссию компании* нельзя описывать путем анализа ее внутреннего устройства. Для построения модели взаимодействия компании с внешней средой (определение *миссии компании* на рынке) необходимо:

идентифицировать рынок (надсистему), частью которого является компания;

определить свойства (потребности) рынка;

определить предназначение (*миссию*) компании, исходя из ее роли на рынке.

Кроме этого, *миссия*, как было сказано выше, это компромисс между потребностями рынка, с одной стороны, и возможностями и желанием компании удовлетворить эти интересы, с другой. Поиск компромисса может быть выполнен по шаблону, представленному на рис. 3.7.

Рис. 3.7  Шаблон разработки миссии (матрица проекций)

При разработке модели *миссии компании* рекомендуется выполнить шаги, описанные в табл. 3.21.

Таблица 3.21

Определение шагов разработки  модели миссии компании

|  |  |
| --- | --- |
| № шага | Описание |
| 1 | Описать базис конкурентоспособности компании - совокупность характеристик компании как социально-экономической системы. Например:             для объекта - уникальность освоенных технологий и исключительность имеющихся в компании ресурсов (финансовых, материальных, информационных и др.)             для субъекта - знания и умения персонала и опыт менеджеров.  Это определяет уникальность ресурсов и навыков компании и формирует позицию "могу". |
| 2 | Выяснить конъюнктуру рынка, т.е. определить наличие платежеспособного спроса на предлагаемые товары или услуги и степень удовлетворения рынка конкурентами. Это позволяет понять потребности рынка и сформировать позицию "надо". |
| 3 | Выявить наличие способствующих и противодействующих факторов для выбранного вида деятельности со стороны государственных институтов в области политики и экономики |
| 4 | Оценить перспективу развития технологии в выбранной сфере деятельности |
| 5 | Оценить возможную поддержку или противодействие общественных организаций |
| 6 | Сопоставить результаты вышеперечисленных действий с учетом правовых, моральных, этических и др. ограничений со стороны персонала и сформировать позицию "хочу". |
| 7 | Оценить уровень возможных затрат и доходов |
| 8 | Оценить возможность достижения приемлемого для всех сторон компромисса и сформулировать *Миссию компании* в соответствии с шаблоном |

*Миссия* в широком понимании представляет собой основную деловую концепцию компании, изложенную в виде восьми положений, определяющих взаимоотношения компании с другими субъектами (рис. 3.8).

* что получит Заказчик в части удовлетворения своих потребностей;
* кто, для чего и как может выступать в качестве партнера компании;
* на какой основе предполагается строить отношения с конкурентами (какова, в частности, готовность пойти на временные компромиссы);
* что получит собственник и акционеры от бизнеса;
* что получат от бизнеса компании менеджеры;
* что получит от компании персонал;
* в чем может заключаться сотрудничество с общественными организациями;
* как будут строиться отношения компании с государством (в частности, возможное участие в поддержке государственных программ).

Рис. 3.8  Шаблон разработки миссии

**3.3.2.2 Шаблон формирования бизнесов**

В соответствии с разработанной *Миссией компании* определяются социально значимые потребности, на удовлетворение которых направлен бизнес компании.

Разработка *бизнес-потенциала компании* может быть выполнена по Шаблону формирования бизнесов, представленному на рис. 3.9.

Рис. 3.9  Шаблон формирования бизнесов

В результате формируются базовый рынок и базовый продукт, детализация которых определяет предложения компании глазами покупателей (товарные группы) и однородные по отношению к продуктам компании группы покупателей (сегменты рынка). С помощью матричной проекции (рис. 3.10) устанавливается соответствие между сформированными товарными группами и сегментами рынка и определяется список бизнесов компании (на пересечении строк и столбцов находятся бизнесы компании).

Рис. 3.10  Шаблон формирования бизнесов (матрица проекций)

**3.3.2.3 Шаблон формирования функционала компании (основных бизнес-функций)**

На основании списка бизнесов, с помощью матричной проекции (рис. 3.11) формируется классификатор бизнес-функций компании.

Рис. 3.11  Шаблон формирования основных бизнес-функций

Для формирования основных функций менеджмента компании сначала разрабатываются и утверждаются два базовых классификатора - "Компоненты менеджмента" (перечень используемых на предприятии инструментов/контуров управления) и "Этапы управленческого цикла" (технологическая цепочка операций, последовательно реализуемых менеджерами при организации работ в любом контуре управления). Далее аналогично, с помощью *матрицы проекций*, формируется список основных функций менеджмента. На рис. 3.12 приведены примеры классификаторов, на основании которых построена матрица - генератор основных функций менеджмента.

Рис. 3.12  Шаблон формирования основных функций менеджмента

Представленные матричные проекции позволяют формировать функции любой степени детализации путем более подробного описания как строк, так и столбцов матрицы.

**3.3.2.4 Шаблон формирования зон ответственности за функционал компании**

Формирование зон ответственности за *функционал компании* выполняется с помощью *матрицы организационных проекций* (рис. 3.13).

Рис. 3.13.  Шаблон распределения функций по организационным звеньям

Эта матрица представляет собой таблицу, в строках которой расположен список исполнительных звеньев**,**в столбцах - список функций, выполняемых в компании. Для каждой функции определяется исполнительное звено, отвечающее за эту функцию.

Заполнение такой таблицы позволяет по каждой функции найти исполняющие ее подразделения или сотрудника. Анализ заполненной таблицы позволяет увидеть "пробелы" как в исполнении функций, так и в загруженности сотрудников, а также рационально перераспределить все задачи между исполнителями и закрепить как систему в документе "Положение об организационной структуре".

Положение об организационной структуре - это внутрифирменный документ, фиксирующий: продукты и услуги компании, функции, выполняемые в компании, исполнительные звенья, реализующие функции, распределение функций по звеньям.

Таблица проекций функций на исполнительные звенья может иметь весьма большую размерность. В средних компаниях это, например, 500 единиц - 20 звеньев на 25 функций. В больших компаниях это может быть 5 000 единиц - 50 звеньев на 100 функций.

Аналогично строится *матрица коммерческой ответственности*.

**3.3.2.5 Шаблон потокового процессного описания**

Шаблон потокового процессного описания приведен на рис. 3.14. Такое описание дает представление о процессе последовательного преобразования ресурсов в продукты усилиями различных исполнителей на основании соответствующих регламентов.

Рис. 3.14  Потоковая процессная модель

Методики построения процессных моделей будут приведены ниже.

**3.3.3  Построение организационно-функциональной модели компании**

Организационно-функциональная модель компании строится на основе функциональной схемы деятельности компании рис. 3.15.

Рис. 3.15   Функциональная схема компании

Для построения организационно-функциональной модели используется всего два типа элементарных моделей.

***Древовидные модели (классификаторы)*** - точные иерархические списки выделенных объектов управления (организационных звеньев, функций, ресурсов, в том числе исполнительных механизмов для бизнес-процессов, документов и их структуры, и т.п.). Каждый элемент классификатора может быть дополнительно охарактеризован рядом атрибутов: тип, шкала, комментарий и т.п. Фактически, классификаторы представляют собой набор управленческих регистров, содержащих, в основном, неколичественную информацию, совокупность которых задает систему координат для описания деятельности компании. Количество таких списков-классификаторов определяется целью построения модели.

***Матричные модели*** - это проекции, задающие систему отношений между классификаторами в любой их комбинации. Связи могут иметь дополнительные атрибуты (направление, название, индекс, шкала и вес). В начальной модели применяется всего несколько классификаторов предметной области:

·         основные группы продуктов и услуг компании;

·         ресурсы, потребляемые компанией в ходе своей деятельности;

·         функции (процессы), поддерживаемые в компании;

·         организационные звенья компании.

В классификаторе функций обычно выделяют три базовых раздела:

·         основные функции - непосредственно связанные с процессом преобразования внешних ресурсов в продукцию и услуги предприятия;

·         функции менеджмента - или функции управления предприятием;

·         функции обеспечения - поддерживающие производственную, коммерческую и управленческую деятельность.

Главной функцией компании является предоставление продуктов и услуг, поэтому сначала производится формальное описание, согласование и утверждение руководством предприятия перечня его бизнесов (направлений коммерческой деятельности), продукции и услуг. Из этого классификатора внешним контрагентам должно быть понятно, чем предприятие интересно рынку, а для внутренних целей - для чего нужен тот или иной *функционал компании*.

В результате этих операций производится идентификация *функционала* и создается единая терминология описания функций предприятия, которая должна быть согласована всеми ведущими менеджерами. При составлении классификатора оргзвеньев важно, чтобы уровень детализации функций соответствовал уровню детализации звеньев. После формирования всех базовых классификаторов с помощью матричных проекций производится их закрепление за оргзвеньями предприятия (рис. 3.13).

Процесс формирования *матрицы проекций* функций на оргзвенья на практике напоминает игру в крестики-нолики. По строчкам таблицы указываются подразделения, по столбцам - функции, составляющие содержание процесса управления или бизнес-процесса в данной компании. На пересечениях функций и подразделений, которые ответственны за выполнение функции, ставится крестик. Для проекций большой размерности используется механизм расстановки связей между двумя классификаторами, представленных списками.

Стандартная практика построения моделей организационно-функциональной структуры компаний поддерживает два уровня детализации: 1) агрегированную модель; 2) детализированную модель.

**Агрегированная модель** - модель организационной структуры, учетные регистры которой имеют ограничение по степени детализации до 2-3 уровней.

**Детализированная модель** - модель организационной структуры, детализация учетных регистров которой производится на более глубоких уровнях, чем в агрегированной модели. Степень детализации в модели обусловлена конкретными потребностями компании (создание определенных организационных регламентов). Цели построения моделей указаны в табл. 3.22.

Таблица 3.22

Цели построения моделей организационной структуры

|  |  |
| --- | --- |
| Вид модели | Цель построения |
| Агрегированная модель | Предоставление информации об организационной структуре высшим руководителям компании для проведения стратегического анализа, анализа соответствия данной структуры стратегии и внешнему окружению компании. Модель может также предоставляться внешним пользователям (например, потенциальным инвесторам как иллюстрация к бизнес-плану, крупным клиентам и др.). |
| Детализированная модель | Предоставление информации о распределении функциональных обязанностей между подразделениями компании, а также об организации бизнес-процессов в компании. |

Построение детализированной модели позволяет создавать различные внутрифирменные регламенты: в первую очередь, Положение об организационной структуре (см. рис. 3.16).

Рис. 3.16  Схема создания Положения об организационно-функциональной структуре компании

Ниже приведен пример описания фрагментов организационно-функциональной модели торгового предприятия (рис. 3.17). Приведенные *матрицы проекций* являются основой для выделения бизнес-процессов предприятия и их владельцев на последующих этапах создания АС.

Рис. 3.17  Распределение функций по подразделениям торгового предприятия

***3.4 Общая характеристика методологий описания предметной области***

**3.4.1 Сущность структурного моделирования предметной области**

В данном пункте приводятся сведения, необходимые для более четкого понимания процесса перехода от первичных моделей, построенных по итогам обследования объекта (и, возможно, по результатам консалтинга, включающего в себя реинжиниринг бизнес-процессов), к моделям, на основании которых непосредственно  создается АС.

Основные требования к структурной модели предметной области следующие:

формализация, обеспечивающая однозначное описание структуры предметной области;

понятность для заказчиков и разработчиков на основе применения графических средств отображения модели;

реализуемость, подразумевающая наличие средств физической реализации модели предметной области в АС;

обеспечение оценки эффективности реализации модели предметной области на основе определенных методов и вычисляемых показателей.

Для реализации перечисленных требований, как правило, строится система моделей, которая отражает структурные и оценочные аспекты функционирования предметной области. Сущность указанных аспектов описана в табл. 3.23.

Таблица 3.23

Сущность структурного и оценочного аспектов функционирования предметной области

|  |  |
| --- | --- |
| Структурные аспекты | Предполагают построение:  объектной структуры, отражающей состав взаимодействующих в процессах материальных и информационных объектов предметной области;  функциональной структуры, отражающей взаимосвязь функций (действий) по преобразованию объектов в процессах;  структуры управления, отражающей события и бизнес-правила, которые воздействуют на выполнение процессов;  организационной структуры, отражающей взаимодействие организационных единиц предприятия и персонала в процессах;  технической структуры, описывающей топологию расположения и способы коммуникации комплекса технических средств.          Для отображения структурного аспекта моделей предметных областей в основном используются графические методы, которые должны гарантировать представление информации о компонентах системы. Главное требование к графическим методам документирования — простота. Графические методы должны обеспечивать возможность структурной декомпозиции спецификаций системы с максимальной степенью детализации и согласований описаний на смежных уровнях декомпозиции. |
| Оценочные аспекты | Связаны с разрабатываемыми показателями эффективности автоматизируемых процессов, к которым относятся:  время решения задач;  стоимостные затраты на обработку данных;  надежность процессов;  косвенные показатели эффективности - объемы производства, производительность труда, оборачиваемость капитала, рентабельность и пр.  Для расчета показателей эффективности, как правило, используются статические методы функционально-стоимостного анализа (ABC) и динамические методы имитационного моделирования. |

С моделированием непосредственно связана проблема выбора языка представления проектных решений, позволяющего как можно больше привлекать будущих пользователей системы к ее разработке. Язык моделирования – это нотация, в основном графическая, которая используется для описания проектов. Нотация представляет собой совокупность графических объектов, используемых в модели. Нотация является синтаксисом языка моделирования. Язык моделирования, с одной стороны, должен делать решения проектировщиков понятными пользователю, с другой стороны, предоставлять проектировщикам средства достаточно формализованного и однозначного определения проектных решений, подлежащих реализации.

Главный критерий адекватности структурной модели предметной области заключается в функциональной полноте разрабатываемой АС.

В основе различных методологий моделирования предметной области АС лежат принципы последовательной детализации абстрактных категорий. Обычно модели строятся на трех уровнях: на внешнем уровне (определении требований), на концептуальном уровне (спецификации требований) и внутреннем уровне (реализации требований). На внешнем уровне модель отвечает на вопрос, что должна делать система, то есть определяется состав основных компонентов системы: объектов, функций, событий, организационных единиц, технических средств. На концептуальном уровне модель отвечает на вопрос, как должна функционировать система, то есть определяется характер взаимодействия компонентов системы. На внутреннем уровне модель отвечает на вопрос: с помощью каких программно-технических средств реализуются требования к системе? С позиции жизненного цикла АС по канонической схеме описанные уровни моделей соответственно строятся на этапах анализа требований, логического (технического) и физического (рабочего) проектирования. В табл. 3.24-3.28 представлены особенности построения моделей предметной области на трех уровнях детализации.

Таблица 3.24

Особенности построения объектной структуры предметной области

на трех уровнях детализации

|  |  |
| --- | --- |
| Основные определения | Объект — это сущность, которая используется при выполнении некоторой *функции* или *операции* (преобразования, обработки, формирования и т.д.). Объекты могут иметь динамическую или статическую природу: динамические объекты используются в одном цикле воспроизводства, например заказы на продукцию, счета на оплату, платежи; статические объекты используются во многих циклах воспроизводства, например, оборудование, персонал, запасы материалов. |
| Внешний  уровень | Выделяются основные виды материальных объектов (например, сырье и материалы, полуфабрикаты, готовые изделия, услуги) и основные виды информационных объектов или документов (например, заказы, накладные, счета и т.д.). |
| Концептуальный уровень | Уточняется состав классов объектов, определяются их атрибуты и взаимосвязи. Таким образом строится обобщенное представление структуры предметной области. |
| Внутренний уровень | Концептуальная модель на этом уровне отображается в виде файлов базы данных, входных и выходных документов АС. Причем динамические объекты представляются единицами переменной информации или документами, а статические объекты — единицами условно-постоянной информации в виде списков, номенклатур, ценников, справочников, классификаторов. Модель базы данных как постоянно поддерживаемого информационного ресурса отображает хранение условно-постоянной и накапливаемой переменной информации, используемой в повторяющихся информационных процессах. |

Таблица 3.25

Особенности построения функциональной структуры предметной области

на трех уровнях детализации

|  |  |
| --- | --- |
| Основные определения | ***Функция*** (*операция*) представляет собой некоторый преобразователь входных объектов в выходные. Последовательность взаимосвязанных по входам и выходам *функций* составляет *бизнес-процесс*. *Функция* *бизнес-процесса* может порождать объекты любой природы (материальные, денежные, информационные). Причем *бизнес-процессы* и информационные процессы, как правило, неразрывны, то есть *функции* материального процесса не могут осуществляться без информационной поддержки. Например, отгрузка готовой продукции осуществляется на основе документа "Заказ", который, в свою очередь, порождает документ "Накладная", сопровождающий партию отгруженного товара.  *Функция* может быть представлена одним действием или некоторой совокупностью действий. В последнем случае каждой *функции* может соответствовать некоторый процесс, в котором могут существовать свои *подпроцессы*, и т.д., пока каждая из подфункций не будет представлять некоторую недекомпозируемую последовательность действий. |
| Внешний  уровень | Определяется список основных бизнес-функций или видов *бизнес-процессов*. Обычно таких *функций* насчитывается 15–20. |
| Концептуальный уровень | Выделенные *функции* декомпозируются и строятся иерархии взаимосвязанных *функций.* |
| Внутренний уровень | Отображается структура информационного процесса в компьютере: определяются иерархические структуры программных модулей, реализующих автоматизируемые *функции.* |

Таблица 3.26

Особенности построения структуры управления предметной области

на трех уровнях детализации

|  |  |
| --- | --- |
| Основные определения | В совокупности *функций* *бизнес-процесса* возможны альтернативные или циклические последовательности в зависимости от различных условий протекания процесса. Эти условия связаны с происходящими событиями во внешней среде или в самих процессах и с образованием определенных состояний объектов (например, заказ принят, отвергнут, отправлен на корректировку). **События** вызывают выполнение *функций*, которые, в свою очередь, изменяют состояния объектов и формируют новые события, и т.д., пока не будет завершен некоторый *бизнес-процесс*. Тогда последовательность событий составляет конкретную реализацию *бизнес-процесса*.  Каждое событие описывается с двух точек зрения: **информационной** и **процедурной**. Информационно событие отражается в виде некоторого сообщения, фиксирующего факт выполнения некоторой *функции* изменения состояния или появления нового. Процедурно событие вызывает выполнение новой *функции*, и поэтому для каждого состояния объекта должны быть заданы описания этих вызовов. Таким образом, события выступают в связующей роли для выполнения *функций* *бизнес-процессов* |
| Внешний  уровень | Определяются список внешних событий, вызываемых взаимодействием предприятия с внешней средой (платежи налогов, процентов по кредитам, поставки по контрактам и т.д.), и список целевых установок, которым должны соответствовать *бизнес-процессы* (регламент выполнения процессов, поддержка уровня материальных запасов, уровень качества продукции и т.д.). |
| Концептуальный уровень | Устанавливаются бизнес-правила, определяющие условия вызова *функций* при возникновении событий и достижении состояний объектов. |
| Внутренний уровень | Выполняется формализация бизнес-правил в виде триггеров или вызовов программных модулей |

Таблица 3.27

Особенности построения организационной структуры предметной области

на трех уровнях детализации

|  |  |
| --- | --- |
| Основные определения | Организационная структура представляет собой совокупность организационных единиц, как правило, связанных иерархическими и процессными отношениями. Организационная единица — это подразделение, представляющее собой объединение людей (персонала) для выполнения совокупности общих *функций* или *бизнес-процессов*. В функционально-ориентированной организационной структуре организационная единица выполняет набор *функций*, относящихся к одной *функции* управления и входящих в различные процессы. В процессно-ориентированной структуре организационная единица выполняет набор *функций*, входящих в один тип процесса и относящихся к разным *функциям* управления. |
| Внешний  уровень | Строится структурная модель предприятия в виде иерархии подчинения организационных единиц или списков взаимодействующих подразделений |
| Концептуальный уровень | Для каждого подразделения задается организационно-штатная структура должностей (ролей персонала). |
| Внутренний уровень | Определяются требования к правам доступа персонала к автоматизируемым *функциям* АС |

Таблица 3.28

Особенности построения технической структуры предметной области

на трех уровнях детализации

|  |  |
| --- | --- |
| Основные определения | Топология определяет территориальное размещение технических средств по структурным подразделениям предприятия, а коммуникация — технический способ реализации взаимодействия структурных подразделений |
| Внешний  уровень | Определяются типы технических средств обработки данных и их размещение по структурным подразделениям |
| Концептуальный уровень | Определяются способы коммуникаций между техническими комплексами структурных подразделений: физическое перемещение документов, машинных носителей, обмен информацией по каналам связи и т.д. |
| Внутренний уровень | Строится модель "клиент-серверной" архитектуры вычислительной сети |

**3.4.2 Принципы структурного анализа предметной области**

Описанные *модели предметной области* нацелены на проектирование отдельных компонентов АС: данных, функциональных программных модулей, управляющих программных модулей, программных модулей интерфейсов пользователей, структуры технического комплекса. Для более качественного проектирования указанных компонентов требуется построение моделей, увязывающих различные компоненты АС между собой. Для правильного отображения взаимодействий компонентов АС важно осуществлять совместное моделирование таких компонентов, особенно с содержательной точки зрения объектов и *функций*. Методология структурного системного анализа существенно помогает в решении таких задач.

В табл. 3.29 представлены ключевые понятия *структурного анализа*.

Таблица 3.29

Ключевые понятияструктурного анализа

|  |  |
| --- | --- |
| Понятие | Определение |
| Операция | элементарное (неделимое) действие, выполняемое на одном рабочем месте |
| Функция | совокупность *операций*, сгруппированных по определенному признаку. |
| Бизнес-процесс | связанная совокупность *функций*, в ходе выполнения которой потребляются определенные ресурсы и создается продукт (предмет, услуга, научное открытие, идея), представляющая ценность для потребителя |
| Подпроцесс | *бизнес-процесс*, являющийся структурным элементом некоторого *бизнес-процесса* и представляющий ценность для потребителя |
| Бизнес-модель | структурированное графическое описание сети процессов и *операций*, связанных с данными, документами, организационными единицами и прочими объектами, отражающими существующую или предполагаемую деятельность предприятия |

***Структурным анализом*** принято называть метод исследования системы, который начинается с ее общего обзора, а затем детализируется, приобретая иерархическую структуру с все большим числом уровней, что иллюстрирует рис.3.18.

Рис. 3.18 Сущность методологии структурного анализа SA

Для таких методов характерно: разбиение на уровни абстракции с ограниченным числом элементов (от 3 до 7); ограниченный контекст, включающий только существенные детали каждого уровня; использование строгих формальных правил записи; последовательное приближение к результату. *Структурный анализ* основан на двух базовых принципах – принципе "разделяй и властвуй" и принципе иерархической упорядоченности. Решение трудных проблем путем их разбиения на множество меньших независимых задач (так называемых "черных ящиков") и организация этих задач в древовидные иерархические структуры значительно повышают понимание сложных систем.

Кроме двух базовых имеются еще ряд важных принципов (табл. 3.30), которые нельзя считать второстепенными, поскольку игнорирование любого из них может привести к непредсказуемым последствиям (в том числе и к провалу всего проекта).

Таблица 3.30

Дополнительные принципы структурного анализа

|  |  |
| --- | --- |
| Название принципа | Описание |
| Принцип абстрагирования. | Выделение существенных аспектов системы и отвлечение от несущественных. |
| Принцип формализации | Необходимость строгого методического подхода к решению проблемы |
| Принцип непротиворечивости | Обеспечение обоснованности и согласованности элементов |
| Принцип структурирования данных | Данные должны быть структурированы и иерархически организованы |

Существуют различные методологии структурного моделирования предметной области, среди которых следует выделить функционально-ориентированные и объектно-ориентированные методологии.

**3.4.3 Сопоставление функционально-ориентированных и объектно-ориентированных методологий описания предметной области**

Процесс бизнес-моделирования может быть реализован в рамках различных методологий, отличающихся прежде всего своим подходом к тому, что представляет собой моделируемая организация. В соответствии с различными представлениями об организации выделяют две основные методологии: функциональную и объектную, каждая из которых может быть реализована на практике в рамках соответствующих прикладных методик моделирования предметной области.

*Функциональная методология* рассматривают организацию как набор *функций*, преобразующий поступающий поток информации в выходной поток. Процесс преобразования информации потребляет определенные ресурсы. Основное отличие от *объектной методологии* заключается в четком отделении *функций* (методов обработки данных) от самих данных.

Осуществляемый в рамках функциональной методологииструктурно-функциональный анализ использует в основном две группы средств, иллюстрирующих функции, выполняемые системой, и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм); наиболее распространенные из них представлены в табл. 3.31.

Таблица 3.31

Виды моделей структурно-функционального анализа

|  |  |  |  |
| --- | --- | --- | --- |
| Краткое название | Полное название | Расшифровка | Ссылка на подраздел, где описана |
| IDEF0 - SADT | Icam DEFinition - Structured Analysis and Design Technique | Структурно-аналитические модели и соответствующие функциональные диаграммы | Подраздел 4.1 |
| DFD | Data Flow Diagrams | Диаграммы потоков данных | Подраздел 4.2 |
| IDEF3 - STD | State Transition Diagrams | Диаграммы переходов состояний | Подраздел 4.3 |
| IDEF1, IDEF1Х - ERD | Entity-Relationship Diagrams | Диаграммы "сущность-связь" | Подраздел 4.4 |

На стадии проектирования АС модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектура ПО, структурные схемы программ и диаграммы экранных форм.

Перечисленные модели в совокупности дают полное описание АС независимо от того, является ли она существующей или вновь разрабатываемой. Состав диаграмм в каждом конкретном случае зависит от необходимой полноты описания системы (раздел 4).

*Объектная методология* рассматривает моделируемую организацию как набор взаимодействующих объектов – производственных единиц. Объект определяется как осязаемая реальность – предмет или явление, имеющие четко определяемое поведение. Целью применения данной методологии является выделение объектов, составляющих организацию, и распределение между ними ответственностей за выполняемые действия (раздел 5).

Проведем краткое сравнение методологий. Достоинства и недостатки функциональных моделей описаны в таблице 3.32.

Таблица 3.32

Достоинства и недостатки функциональных моделей

|  |  |
| --- | --- |
| Достоинства | ·            реализация структурного подхода к проектированию АС по принципу "сверху-вниз", когда каждый функциональный блок может быть декомпозирован на множество подфункций и т.д., выполняя, таким образом, модульное проектирование АС;  ·            процедурная строгость декомпозиции АС;  ·            наглядность представления. |
| Недостатки | ·            процессы и данные существуют отдельно друг от друга — помимо функциональной декомпозиции существует структура данных, находящаяся на втором плане;  ·            не ясны условия выполнения процессов обработки информации, которые динамически могут изменяться. |

Перечисленные недостатки снимаются в объектно-ориентированных моделях, в которых главным структурообразующим компонентом выступает класс объектов с набором функций, которые могут обращаться к атрибутам этого класса. При объектно-ориентированном подходе изменяется и принцип проектирования АС. Сначала выделяются классы объектов, а далее в зависимости от возможных состояний объектов (жизненного цикла объектов) определяются методы обработки (функциональные процедуры), что обеспечивает наилучшую реализацию динамического поведения АС.

Для объектно-ориентированного подхода разработаны графические методы моделирования предметной области, обобщенные в языке унифицированного моделирования UML. Однако по наглядности представления модели пользователю-заказчику объектно-ориентированные модели явно уступают функциональным моделям.

С точки зрения бизнес-моделирования каждый из представленных подходов обладает своими преимуществами. Объектный подход позволяет построить более устойчивую к изменениям систему, лучше соответствует существующим структурам организации. Функциональное моделирование предпочтительнее в тех случаях, когда организационная структура находится в процессе изменения или вообще слабо оформлена, однако решаемые задачи стабильны и хорошо регламентированы.

При выборе методики моделирования предметной области обычно в качестве критерия выступает степень ее динамичности. Для более регламентированных задач больше подходят функциональные модели, для более адаптивных бизнес-процессов (управления рабочими потоками, реализации динамических запросов к информационным хранилищам) — объектно-ориентированные модели. Однако в рамках одной и той же АС для различных классов задач могут требоваться различные виды моделей, описывающих одну и ту же проблемную область. В таком случае должны использоваться комбинированные модели предметной области*.*

Наилучшим способом преодоления недостатков классических методик, построенных в рамках рассмотренных методологий, является формирование **синтетической методики**, объединяющей отдельные этапы разных методик. Идея синтетической методики заключается в последовательном применении функционального и объектного подхода с учетом возможности реинжиниринга существующей ситуации.

**3.4.4 Референтная модель бизнес-процесса**

В заключение отметим еще один подход к построению типовой модели объекта автоматизации. Согласно этому подходу в качестве основного каркаса, объединяющего и систематизирующего все знания по бизнес-модели объекта, используется *референтная модель*. ***Референтная модель*** — это модель эффективного бизнес-процесса, созданная для предприятия конкретной отрасли, внедренная на практике и предназначенная для использования при разработке/реорганизации бизнес-процессов на других предприятиях. По сути, *референтные модели* представляют собой эталонные схемы организации бизнеса, разработанные для конкретных бизнес-процессов на основе реального опыта внедрения в различных компаниях по всему миру. Они включают в себя проверенные на практике процедуры и методы организации управления. *Референтные модели* позволяют предприятиям начать разработку собственных моделей на базе уже готового набора функций и процессов.

*Референтная модель* бизнес-процесса представляет собой совокупность логически взаимосвязанных функций. Для каждой функции указывается исполнитель, входные и выходные документы или информационные объекты. Элементы (функции и документы) *референтной модели* бизнес-процесса содержат ссылки на соответствующие объекты АС, а также документы и другую информацию (пользовательские инструкции, ответственных разработчиков), расположенную в репозитории проекта. Отсюда и название — *референтная модель* (в переводе с английского ссылочная модель).

[3 МЕТОДЫ ОБСЛЕДОВАНИЯ И АНАЛИЗА ПРЕДМЕТНОЙ ОБЛАСТИ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805200)

[3.1 Общая характеристика процесса и методов обследования и анализа объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805201)

[3.1.1 Общие положения  по обследованию объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805202)

[3.1.2 Сущность, роль и место консалтинга при проведении обследования объекта автоматизации и создании АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805203)

[3.1.3  Процессный подход к организации деятельности предприятия](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805204)

[3.1.4 Выделение и классификация процессов на предприятии](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805205)

[3.1.5 Сущность реинжиниринга объектов автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805206)

[3.1.5.1 Понятие «реинжиниринг бизнеса» и его отличие от совершенствования бизнеса](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805207)

[3.1.5.2 Условия успешного реинжиниринга и факторы риска](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805208)

[3.2 Краткий обзор формализованных методов обследования и анализа объекта автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805209)

[3.2.1  Методы, основанные на построении графовых моделей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805210)

[3.2.1.1 Графовая модель структурно-функциональных связей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805211)

[3.2.1.2 Метод сетевого моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805212)

[3.2.1.3 Графоаналитический метод](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805213)

[3.2.2 Методы оперограмм  и структурно-информационно- временных схем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805214)

[3.2.3  Сущность методов разработки спецификаций](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805215)

[3.3 Анализ и моделирование функциональной области внедрения АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805216)

[3.3.1  Основные понятия организационного бизнес-моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805217)

[3.3.2  Шаблоны организационного бизнес-моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805218)

[3.3.2.1 Шаблон разработки миссии](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805219)

[3.3.2.2 Шаблон формирования бизнесов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805220)

[3.3.2.3 Шаблон формирования функционала компании (основных бизнес-функций)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805221)

[3.3.2.4 Шаблон формирования зон ответственности за функционал компании](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805222)

[3.3.2.5 Шаблон потокового процессного описания](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805223)

[3.3.3  Построение организационно-функциональной модели компании](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805224)

[3.4 Общая характеристика методологий описания предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805225)

[3.4.1 Сущность структурного моделирования предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805226)

[3.4.2 Принципы структурного анализа предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805227)

[3.4.3 Сопоставление функционально-ориентированных и объектно-ориентированных методологий описания предметной области](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805228)

[3.4.4 Референтная модель бизнес-процесса](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk3\3%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456805229)

[4 ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608390)

[4.1 Методология IDEF0 (SADT) функционального моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608391)

[4.1.1 Состав функциональной модели и принципы ее построения](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608392)

[4.1.2 Разновидности диаграмм IDEF0](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608393)

[4.1.3 Стрелки и туннелирование стрелок на диаграммах IDEF0](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608394)

[4.1.4 Каркас диаграммы, слияние и расщепление моделей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608395)

[4.2 Методология DFD моделирования потоков данных (процессов)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608396)

[4.2.1 Характеристика методологии и основные символы DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608397)

[4.2.2 Расширения реального времени в DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608398)

[4.2.3 Рекомендации по практическому применению методологии DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608399)

[4.3 Методологии описания процессов  IDEF3, спецификаций управления STD и имитационного моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608400)

[4.3.1 Сущность методологии IDEF3](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608401)

[4.3.2 Методы задания спецификаций процессов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608402)

[4.3.2.1 Структурированный естественный язык - псевдокод](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608403)

[4.3.2.2 Таблицы и деревья решений](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608404)

[4.3.2.3 Визуальные языки проектирования спецификаций](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608405)

[4.3.3 Спецификации управления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608406)

[4.3.3.1 Понятие диаграммы переходов состояний](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608407)

[4.3.3.2 Правила и способы построения STD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608408)

[4.3.3.3 Таблицы и матрицы переходов состояний](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608409)

[4.3.4 Роль и место имитационного моделирования в проектировании АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608410)

[4.4 Методологии моделирования информационного обеспечения IDEF1, IDEF1X, ERD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608411)

[4.4.1 Основные положения, уровни и методологии моделирования данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608412)

[4.4.2 Сущности, отношения и связи в нотации Чена](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608413)

[4.4.3 Диаграммы атрибутов и категоризации сущностей в нотации Чена](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608414)

[4.4.4 Нотация Баркера](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608415)

[4.4.5 Словарь данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608416)

[4.4.5.1 Назначение, определение и содержимое словаря данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608417)

[4.4.5.2 БНФ-нотация](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608418)

# 4 ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ

## *4.1 Методология IDEF0 (SADT) функционального моделирования*

### 4.1.1 Состав функциональной модели и принципы ее построения

Методология IDEF0 (см. табл. 3.31) представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Функциональная модель отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями. Основные элементы этой методологии основываются на принципах (концепциях) представленных в табл.4.1.

Табл. 4.1

Принципы (концепции) методологии IDEF0

|  |  |
| --- | --- |
| Принцип | Описание |
| Графическое представление блочного моделирования | Графика блоков и дуг диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него. Взаимодействие блоков друг с другом описываются посредством интерфейсных дуг, выражающих "ограничения", которые в свою очередь определяют, когда и каким образом функции выполняются и управляются |
| Строгость и точность | Выполнение правил IDEF0 требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика. |
| Ограничение количества блоков на каждом уровне декомпозиции | Правило 3-6 блоков |
| Связность диаграмм | Правило нумерации блоков |
| Уникальность меток и наименований | Отсутствие повторяющихся имен |
| Синтаксические правила для графики | Правила графического отображения блоков и дуг |
| Разделение входов и управлений | Правило определения роли данных |
| Отделение организации от функции | Исключение влияния организационной структуры на функциональную модель |

Методология может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих систем IDEF0 может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Результатом применения методологии IDEF0 является модель, которая состоит из иерархически связанного набора схем (диаграмм), фрагментов текстов и глоссария, имеющих ссылки друг на друга. Диаграммы - главные компоненты модели. На них все функции АС и интерфейсы представлены как блоки и дуги. Каждая схема является детализацией какого-либо объекта и окружающей среды из схемы предыдущего более высокого уровня (рис.3.18). Количество уровней детализации в схеме для различных объектов может быть различно и зависит от сложности объектов.

Анализируемый объект представляется на схеме в виде прямоугольников (функциональных блоков) и связей между ними, показываемых стрелками (интерфейсными дугами). Совокупность стрелок, входящих в схему и выходящих из нее, определяет ***среду схемы***. Среда, изображенная стрелками, состоит из четырех составляющих: Вход (Input); Выход (Output); Управление (Control); Механизм (Mechanism). При этом тип составляющих четко определяются стороной прямоугольника, к которой примыкает стрелка (рис.4.1).

Рис. 4.1 Составляющие среды схемы

Объекты анализа могут быть двух типов: либо предметы, либо операции. Если объект анализа – предмет, то операции образуют его внешнюю среду, и наоборот. Например, при разработке ПО в качестве предметов рассматриваются данные, а в качестве операций – преобразования данных. При этом объектом анализа могут быть данные в среде преобразований или преобразования в среде данных.

Чаще всего объекты анализа представляют собой*работы* **(Activity),** которые обозначают поименованные процессы, функции или задачи, происходящие в течение определенного времени и имеющие распознаваемые результаты. Все работы должны быть названы и определены. Имя работы должно быть выражено отглагольным существительным, обозначающим действие (например, "Деятельность компании", "Прием заказа" и т.д.).

Работы на диаграммах декомпозиции обычно располагаются по диагонали от левого верхнего угла к правому нижнему. Такой порядок называется порядком доминирования. Согласно этому принципу расположения в левом верхнем углу помещается самая важная работа или работа, выполняемая по времени первой. Далее вправо вниз располагаются менее важные или выполняемые позже работы. Такое размещение облегчает чтение диаграмм, кроме того, на нем основывается понятие взаимосвязей работ.

### 4.1.2 Разновидности диаграмм IDEF0

IDEF0-модель может содержать четыре типа диаграмм:

контекстную диаграмму (в каждой модели может быть только одна контекстная диаграмма);

диаграммы декомпозиции;

диаграммы дерева узлов;

диаграммы только для экспозиции (FEO).

Построение модели всей системы начинается с построения *Контекстной диаграммы, которая* является вершиной древовидной структуры диаграмм и представляет собой самое общее описание системы и ее взаимодействия с внешней средой. Система представляется в виде простейшего компонента - одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок представляет всю систему как единое целое, имя, указанное в блоке, является общим. Это верно и для стрелок - они также представляют полный набор внешних интерфейсов системы в целом.

В пояснительном тексте к контекстной диаграмме должна быть указана **цель** (Purpose) построения диаграммы в виде краткого описания и зафиксирована **точка зрения** (Viewpoint).

Определение и формализация цели разработки IDEF0-модели является крайне важным моментом. Фактически цель определяет соответствующие области в исследуемой системе, на которых необходимо фокусироваться в первую очередь.

**Точка зрения определяет основное направление развития модели и уровень необходимой детализации**. Четкое фиксирование точки зрения позволяет разгрузить модель, отказавшись от детализации и исследования отдельных элементов, не являющихся необходимыми, исходя из выбранной точки зрения на систему. Правильный выбор точки зрения существенно сокращает временные затраты на построение конечной модели.

После описания системы в целом проводится разбиение ее на крупные фрагменты. Этот процесс называется функциональной декомпозицией, а диаграммы, которые описывают каждый фрагмент и взаимодействие фрагментов, называются диаграммами декомпозиции. При декомпозиции блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами и т.д. Во всех случаях каждая подфункция может содержать только те элементы, которые входят в исходную функцию. Модель не может опустить какие-либо элементы, так как родительский блок и его интерфейсы обеспечивают контекст. К нему нельзя ничего добавить и из него не может быть ничего удалено.

После каждого сеанса декомпозиции проводятся сеансы экспертизы — эксперты предметной области указывают на соответствие реальных бизнес-процессов созданным диаграммам. Найденные несоответствия исправляются, и только после прохождения экспертизы без замечаний можно приступать к следующему сеансу декомпозиции. Синтаксис описания системы в целом и каждого ее фрагмента одинаков во всей модели.

На рисунке 4.2 в качестве примера приведены четыре диаграммы и их взаимосвязи в рамках IDEF0 модели. Каждая диаграмма иллюстрирует "внутреннее строение" блока на родительской диаграмме.

Рис. 4.2. Декомпозиция диаграмм

Заметим, что на IDEF0-диаграммах не указаны явно ни последовательность, ни время.

*Диаграмма дерева узлов* показывает иерархическую зависимость *работ*, но не взаимосвязи между *работами*. Для того, чтобы указать положение любой диаграммы или блока в иерархии, используются номера диаграмм. Нумерация проводится слева направо и сверху вниз. На рисунке 4.3 в качестве примера показано типичное дерево диаграмм.

Рис. 4.3. Иерархия диаграмм

*Диаграмм деревьев узлов* может быть в модели сколь угодно много, поскольку дерево может быть построено на произвольную глубину и не обязательно с корня. ***Диаграмма дерева узлов*** позволяет рассмотреть всю модель целиком, но не показывает взаимосвязи между *работами*. Процесс создания модели *работ* является итерационным, следовательно, *работы* могут менять свое расположение в дереве узлов многократно. Чтобы не запутаться и проверить способ декомпозиции, следует после каждого изменения создавать *диаграмму дерева узлов*.

*Диаграммы для экспозиции (FEO - For Exposition Only)* строятся для иллюстрации отдельных фрагментов модели, для иллюстрации альтернативной точки зрения, либо для специальных целей. Эти д*иаграммы* часто используются в модели для отображения отдельных деталей, которые не поддерживаются явно синтаксисом IDEF0. Диаграммы FEO позволяют нарушить любое синтаксическое правило, так как, по сути, являются просто картинками.

### 4.1.3 Стрелки и туннелирование стрелок на диаграммах IDEF0

*Стрелки* (Arrow) описывают взаимодействие *работ* и представляют собой некую информацию, выраженную существительными (например, "Звонки клиентов", "Правила и процедуры", "Бухгалтерская система").

В IDEF0 кроме 4-х основных видов стрелок (Вход, Выход, Управление, Механизм, см. п. 4.1.1) - выделяют дополнительный тип стрелок:**Вызов** (**Call**). Это специальная стрелка, указывающая на другую модель работы. Стрелка вызова рисуется как исходящая из нижней грани работы. Стрелка вызова используется для указания того, что некоторая работа выполняется за пределами моделируемой системы.

Вводится также понятие **Граничных** *стрелок*. Стрелки на контекстной диаграмме служат для описания взаимодействия системы с окружающим миром. Они могут начинаться у границы диаграммы и заканчиваться у работы, или наоборот. Такие стрелки и называются граничными. Для их обозначения используются **ICOM-коды (**ICOM - аббревиатура от Input, Control, Output и Mechanism). Код ICOM содержит префикс, соответствующий типу *стрелки* (I, С, О или М), и порядковый номер.

В моделях используется словарь **стрелок,** в котором определяется стрелка и вносится относящийся к ней комментарий. Словарь стрелок решает очень важную задачу - обеспечить однозначное и правильное понимание сути объектов аналитиками и специалистом предметной области. Поскольку формальные определения часто сложны для восприятия, аналитик вынужден употреблять профессиональный жаргон, а чтобы не возникло неоднозначных трактовок, в словаре стрелок каждому понятию можно дать расширенное и, если это необходимо, формальное определение.

Для описания особенностей стрелок используются следующие термины (табл.4.2)

Табл. 4.2

Особенные стрелки в методологии IDEF0

|  |  |
| --- | --- |
| Термин | Определение |
| Внутренние *стрелки* | Для связи *работ* между собой используются внутренние *стрелки*, то есть *стрелки*, которые не касаются границы диаграммы, начинаются у одной и кончаются у другой *работы* |
| Явные *стрелки* | Явная *стрелка* имеет источником одну-единственную *работу* и назначением тоже одну-единственную *работу* |
| Разветвляющиеся и сливающиеся *стрелки* | Используются для моделирования в IDEF0 следующих ситуаций: 1) Одни и те же данные или объекты, порожденные одной *работой*, могут использоваться сразу в нескольких других *работах*; 2) С*трелки*, порожденные в разных *работах*, могут представлять собой одинаковые или однородные данные или объекты, которые в дальнейшем используются или перерабатываются в одном месте. |

Смысл разветвляющихся и сливающихся стрелок передается именованием каждой ветви стрелок. Существуют определенные правила именования таких стрелок.

Иногда отдельные интерфейсные дуги высшего уровня не имеет смысла продолжать рассматривать на диаграммах нижнего уровня, или наоборот — отдельные дуги нижнего отражать на диаграммах более высоких уровней – это будет только перегружать диаграммы и делать их сложными для восприятия. Для решения подобных задач в стандарте IDEF0 предусмотрено понятие туннелирования. Обозначение "туннеля" (Arrow Tunnel) в виде двух круглых скобок вокруг начала интерфейсной дуги обозначает, что эта дуга не была унаследована от функционального родительского блока и появилась (из "туннеля") только на этой диаграмме. В свою очередь, такое же обозначение вокруг конца (стрелки) интерфейсной дуги в непосредственной близи от блока–приемника означает тот факт, что в дочерней по отношению к этому блоку диаграмме эта дуга отображаться и рассматриваться не будет. Чаще всего бывает, что отдельные объекты и соответствующие им интерфейсные дуги не рассматриваются на некоторых промежуточных уровнях иерархии, – в таком случае они сначала "погружаются в туннель", а затем при необходимости "возвращаются из туннеля.

### 4.1.4 Каркас диаграммы, слияние и расщепление моделей

Схемы строят на бланках, в верхней и нижней частях которых выделяемой области идентификации - граничные рамки, которые называются каркасом диаграммы.

Каркас содержит заголовок (верхняя часть рамки) и подвал (нижняя часть). Заголовок каркаса используется для отслеживания диаграммы в процессе моделирования. Нижняя часть используется для идентификации и позиционирования в иерархии диаграммы. Смысл элементов каркаса приведен в табл. 4.3 – 4.4.

Таблица 4.3

Поля заголовка каркаса (слева направо)

|  |  |
| --- | --- |
| **Поле** | **Смысл** |
| Used At | Используется для указания на родительскую работу в случае, если на текущую диаграмму ссылались посредством стрелки вызова |
| Autor, Date, Rev, Project | Имя создателя диаграммы, дата создания и имя проекта, в рамках которого была создана диаграмма. REV-дата последнего редактирования диаграммы |
| Notes 123456789 10 | Используется при проведении сеанса экспертизы. Эксперт должен (на бумажной копии диаграммы) указать число замечаний, вычеркивая цифру из списка каждый раз при внесении нового замечания |
| Status | Статус отображает стадию создания диаграммы, отображая все этапы публикации |
| Working | Новая диаграмма, кардинально обновленная диаграмма или новый автор диаграммы |
| Draft | Диаграмма прошла первичную экспертизу и готова к дальнейшему обсуждению |
| Recommended | Диаграмма и все ее сопровождающие документы прошли экспертизу. Новых изменений не ожидается |
| Publication | Диаграмма готова к окончательной печати и публикации |
| Reader | Имя читателя (эксперта) |
| Date | Дата прочтения (экспертизы) |
| Context | Схема расположения *работ* в диаграмме верхнего уровня. *Работа*, являющаяся родительской, показана темным прямоугольником, остальные – светлым. На *контекстной диаграмме* (А-0) показана надпись ТОР. В левом нижнем углу показывается номер по узлу родительской диаграммы: |

Таблица 4.4

Поля подвала каркаса (слева направо)

|  |  |
| --- | --- |
| **Поле** | **Смысл** |
| Node | Номер узла диаграммы (номер родительской *работы* ) |
| Title | Имя диаграммы. По умолчанию — имя родительской *работы* |
| Number | C-Number, уникальный номер версии диаграммы |
| Page | Номер страницы, может использоваться как номер страницы при формировании папки |

Возможность слияния и расщепления моделей обеспечивает коллективную работу над проектом. Так, руководитель проекта может создать декомпозицию верхнего уровня и дать задание аналитикам продолжить декомпозицию каждой ветви дерева в виде отдельных моделей. После окончания работы над отдельными ветвями все подмодели могут быть слиты в единую модель. С другой стороны, отдельная ветвь модели может быть отщеплена для использования в качестве независимой модели, для доработки или архивирования.

Обычно для организации процесса слияния и разветвления моделей используют стрелки вызова.

## *4.2 Методология DFD моделирования потоков данных (процессов)*

### 4.2.1 Характеристика методологии и основные символы DFD

Для целей моделирования систем, вообще, и структурного анализа, в частности, используются *три группы средств*, иллюстрирующих:

1.    Функции, которые система должна выполнять;

2.    Отношение между данными;

3.    Зависящее от времени поведение системы (аспекты реального времени).

Наиболее часто и эффективно применяемыми являются следующие *средства решения* указанных задач (см. табл. 3.31):

-        DFD (Data Flow Diagrams) – диаграммы потоков данных, совместно со словарями данных и спецификациями процессов или миниспецификациями;

-        ERD (Entity Relationship Diagrams) диаграммы «сущность - связь»;

-        STD (State Transition Diagrams) – диаграммы переходов состояний.

Все они содержат графические и текстовые средства моделирования: первые используются для удобства демонстрации основных компонентов модели, вторые – для обеспечения точного определения её компонентов и связей.

Целью функциональной методологии моделирования потоков данных является построение модели рассматриваемой системы в виде диаграммы потоков данных (DFD), обеспечивающей правильное описание выходов (отклика системы в виде данных) при заданном воздействии на вход системы (подаче сигналов через внешние интерфейсы).

***DFD*** показывает внешние по отношению к системе источники и стоки данных (адресаты), идентифицирует логические функции (процессы) и группы элементов данных, связывающие одну функцию с другой (потоки), а также идентифицирует хранилища данных (накопители), к которым осуществляется доступ. Структуры потоков данных и определения их компонентов хранятся и анализируются в словаре данных. Каждая логическая функция (процесс) может быть детализирована с помощью DFD нижнего уровня. Когда дальнейшая детализация перестаёт быть полезной, переходят к выражению логики функций с помощью спецификаций процессов (миниспецификации). Содержимое каждого хранилища также сохраняют в словаре данных, модель данных раскрывается с помощью ***ERD***. В случае наличия реального времени DFD дополняются средствами описания поведения, зависящего от времени, раскрывающимися с помощью ***STD*** (рис.4.4).

Таким образом, строится ***логическая функциональная спецификация*** – подробное описание того, что должна делать система, освобождённое, насколько это возможно, от рассмотрения путей реализации. Это даёт проектировщику чёткое представление о конечных результатах, которых следует достигать.

Рис. 4.4 Взаимоотношение наиболее часто используемых средств структурного анализа в рамках методологии DFD

DFD являются основным средством моделирования функциональных требований к проектируемой системе. С их помощью эти требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных.

Главная цель таких средств – продемонстрировать как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

Для изображения DFD традиционно используются две нотации: Йодана и Гейна-Сарсона. Соответствующие символы представлены на рис. 4.5. Назначение символов уточняется в табл. 4.5.

|  |  |  |
| --- | --- | --- |
| *Символ (объект)* | *Нотация Йодана* | *Нотация Гейна-Сарсона* |
| ПОТОК ДАННЫХ |  |  |
| ПРОЦЕСС |  |  |
| ХРАНИЛИЩЕ ДАННЫХ |  |  |
| ВНЕШНЯЯ СУЩНОСТЬ |  |  |

Рис. 4.5 Основные символы DFD

Табл. 4.5

Назначение символов DFD

|  |  |
| --- | --- |
| Символ | Назначение |
| ***ПОТОК ДАННЫХ*** | является механизмом, использующимся для моделирования передачи инфор-мации или даже физических компонентов из одной части системы в другую. Иногда информация может двигаться в одном направлении, обрабатываться и возвращаться назад в её источник. Такая ситуация может моделироваться либо двумя различными потоками, либо одним, но двунаправленным. |
| ***ПРОЦЕСС*** | предназначен для продуцирования выходных потоков из входных в соответствии с действием, задаваемым именем процесса. Это имя должно содержать глагол в неопределённой форме с последующим дополнением, например, ВЫЧИСЛИТЬ МАКСИМАЛЬНУЮ ВЫСОТУ. Каждый процесс должен иметь уникальный номер для ссылок на него внутри диаграммы. Этот номер может использоваться совместно с номером диаграммы для получения уникального индекса процесса во всей модели. |
| ***ХРАНИЛИЩЕ ДАННЫХ*** | позволяет на определённых участках определить данные, которые будут сохраняться в памяти между процессами. Фактически хранилище является «срезом» потоков данных во времени.  Информация, которую оно содержит, может использоваться в любое время после её определения, а также её можно выбирать в любом порядке. В случае, когда поток данных входит или выходит из хранилища, его структура соответствует структуре хранилища полностью или в определённой заранее степени. Если соответствие полное, то поток данных и хранилище имеют одинаковое имя, которое для потока можно не  писать на диаграмме. Имя хранилища должно идентифицировать его содержимое и быть существительным. |
| ***ВНЕШНЯЯ СУЩНОСТЬ*** | представляет собой материальный объект вне контекста системы, являющийся источником или приёмником системных данных. Её имя должно содержать существительное, например, СКЛАД ТОВАРОВ. Предполагается, что объекты, представленные такими узлами, не должны участвовать ни в какой обработке. |

Кроме основных элементов, в состав DFD входят словари данных и миниспецификации (спецификации процессов обработки).

**Словари данных** являются каталогами всех элементов данных, присутствующих в DFD, включая групповые и индивидуальные потоки данных, хранилища и процессы, а также все их атрибуты.

**Миниспецификации**— описывают DFD-процессы нижнего уровня. Фактически миниспецификации представляют собой алгоритмические описания задач, выполняемых процессами. Множество всех миниспецификаций является полной спецификацией системы.

Отдельные данные в системе часто являются независимыми.  Иногда необходимо иметь дело с несколькими независимыми данными одновременно. Например, в системе имеются потоки ЯБЛОКИ, АПЕЛЬСИНЫ, ГРУШИ. Эти потоки могут быть сгруппированы с помощью введения нового потока ФРУКТЫ. Для этого необходимо определить формально поток ФРУКТЫ, как состоящий из нескольких элементов-потомков. Такое определение задаётся с помощью ***формы Бэкуса-Наура*** (БНФ) в словаре данных (см. ниже п.4.4.5).  В свою очередь поток ФРУКТЫ может сам содержаться в потоке ЕДА совместно с потоками ОВОЩИ, МЯСО и т.п. Такие потоки, объединяющие несколько потоков, называются ***групповыми потоками***.

Обратная операция расщепления потоков на подпотоки осуществляется с использованием группового узла, позволяющего расщепить поток на любое число подпотоков. При расщеплении также необходимо формально определить подпотоки в словаре данных с помощью БНФ. Аналогичным образом осуществляется и декомпозиция потоков через границы диаграмм, что позволяет упростить детализирующую DFD.

На рис. 4.6 представлены, а в табл. 4.6 описаны символы, используемые для расширения диаграмм потоков данных.

|  |  |
| --- | --- |
| *Символ (объект)* | *Нотация Йодана* |
| ГРУППОВОЙ УЗЕЛ |  |
| УЗЕЛ ПРЕДОК |  |
| НЕИСПОЛЬЗУЕМЫЙ УЗЕЛ |  |
| УЗЕЛ ИЗМЕНЕНИЯ ИМЕНИ |  |

Рисунок 4.6 Символы, используемые для расширения диаграмм потоков данных

Табл. 4.6

Назначение дополнительных символов DFD

|  |  |
| --- | --- |
| Символ | Назначение |
| ***ГРУППОВОЙ УЗЕЛ*** | предназначен для расщепления и объединения потоков. В некоторых случаях могут отсутствовать (то есть, фактически вырождаться в точки слияния / расщепления потоков данных на диаграмме). |
| ***УЗЕЛ ПРЕДОК*** | позволяет увязывать входные и выходные потоки между детализируемым процессом и детализирующей DFD. |
| ***НЕИСПОЛЬЗУЕМЫЙ УЗЕЛ*** | применяется в ситуации, когда декомпозиция данных производится в групповом узле, но при этом требуются не все элементы входящего в узел потока. |
| ***УЗЕЛ ИЗМЕНЕНИЯ ИМЕНИ*** | позволяет неоднозначно именовать потоки, хотя при этом их содержимое эквивалентно. Например, если при проектировании разных частей системы один и тот же фрагмент данных получил различные имена, то эквивалентность соответствующих потоков данных обеспечивается узлом изменения имени. При этом один из потоков данных является входным для данного узла, а другой – выходным. |

Дополнительные символы обеспечивают более полную структуризацию данных, увеличивают наглядность и читабельность диаграмм. Дополнительным компонентом DFD является текст в свободном формате, размещаемый в любом месте диаграммы.

### 4.2.2 Расширения реального времени в DFD

В системах реального времени для дополнения модели функционирования данных (иерархии DFD) средствами описания управляющих аспектов используются те же символы, что рассмотрены выше (см. рис. 4.5, 4.6), только в их названиях добавляется слово «управляющий» и они рисуются штриховой линией. Уточним их назначение в табл. 4.7.

Табл. 4.7

Назначение символов DFD для учета аспектов реального времени

|  |  |
| --- | --- |
| Символ | Назначение |
| ***УПРАВЛЯЮЩИЙ ПРОЦЕСС*** | представляет собой интерфейс между DFD и спецификациями управления, собственно моделирующими и документирующими аспекты реального времени. Его имя указывает на тип управляющей деятельности, вырабатываемой спецификацией. Фактически управляющий процесс представляет собой преобразователь входных управляющих потоков в выходные управляющие потоки; при этом точное описание этого преобразования должно задаваться в спецификации управления. |
| ***УПРАВЛЯЮЩЕЕ ХРАНИЛИЩЕ ДАННЫХ*** | представляет собой срез управляющего потока во времени. Содержащаяся в нём управляющая информация может использоваться в любое время после занесения её в хранилище, при этом соответствующие данные могут быть использованы в произвольном порядке. Имя управляющего хранилища должно идентифицировать его содержимое и быть существительным. Отличие управляющего хранилища от традиционного в том, что оно может содержать только управляющие потоки, при этом все дркгие характеристики хранилищ идентичны. |
| ***УПРАВЛЯЮЩИЙ ПОТОК ДАННЫХ*** | представляет собой «трубопровод», через который проходит управляющая информация. Его имя не должно содержать глаголов, а только существительные и прилагательные. Обычно управляющий поток имеет дискретное значение. Это может быть, например, сигнал, представляющий состояние или вид операции. Логически управляющий процесс есть некий командный пункт, реагирующий на изменение внешних условий, передаваемые ему с помощью управляющих потоков, и продуцирующий в соответствии со своей внутренней логикой выполняемые процессами команды. |

При этом режим выполнения процесса зависит от типа управляющего потока. Имеются следующие *типы управляющих потоков*:***Т-поток*** (Trigger flow),***A-поток*** (Activator flow) и***E/D-поток*** (Enable / Disable flow) (табл. 4.8).

Табл. 4.8

Описание типов управляющих потоков

|  |  |
| --- | --- |
| Символ | Назначение |
| ***Т-поток***  (Trigger flow) | является потоком управления процессом, который может вызывать выполнение процесса. При этом процесс как бы включается одной короткой операцией. Это аналог выключателя света, единственным нажатием которого запускается процесс горения лампы. |
| ***A-поток***  (Activator flow) | является потоком управления процессом, который может изменять выполнение отдельного процесса. Используется для обеспечения непрерывности выполнения процесса до тех пор, пока поток включён (то есть течёт непрерывно), с выключением потока выполнение процесса завершается. Аналог этого – переключатель лампы, который может быть включен и выключен. |
| ***E/D-поток***  (Enable / Disable flow) | является потоком управления процессом, который может переключать выполнение отдельного процесса. Течение по Е-линии вызывает выполнение процесса, которое продолжается, пока не возбуждается течение по D-линии. Аналог этого – выключатель с двумя кнопками: для включения света и для выключения света. Отметим, что можно использовать три типа таких потоков: E-поток, D-поток, E/D-поток. |

Иногда возникает необходимость в представлении одного и того же фрагмента данных потоками различных типов. Например, поток данных СКОРОСТЬ МАШИНЫ в отдельных случаях может использоваться как управляющий для контроля критического значения. Для обеспечения этого используется ***УЗЕЛ ИЗМЕНЕНИЯ ТИПА*** (рис. 4.7). Для узла в примере поток данных является входным, а управляющий поток – выходным.

Рис. 4.7 УЗЕЛ ИЗМЕНЕНИЯ ТИПА

### 4.2.3 Рекомендации по практическому применению методологии DFD

При построении иерархии DFD целесообразно следовать определенным рекомендациям (табл. 4.9).

Табл. 4.9

Рекомендации по построению иерархии DFD

|  |  |
| --- | --- |
| № | Рекомендация |
| 1 | Не загромождать диаграмму несущественными на данном уровне деталями |
| 2 | Декомпозиция потоков данных должна осуществляться параллельно с декомпозицией процессов |
| 3 | Выбирать ясные, отражающие суть дела имена процессов и потоков с целью улучшения понимания диаграмм (по возможности, не использовать аббревиатуры) |
| 4 | Однозначно определять функционально-идентичные процессы на самом верхнем уровне, где такой процесс необходим, и ссылаться на него на нижнем уровне |
| 5 | Пользоваться простейшими диаграммными техниками: если что-то можно описать с помощью DFD, то это и необходимо делать, не используя более сложные объекты |
| 6 | Отделять управляющие структуры от обрабатывающих (то есть, процессов), использовать управляющие структуры |

В соответствии с этими рекомендациями процесс построения структурно-аналитической модели разбивается на ряд этапов (табл. 4.10).

Табл. 4.10

Этапы  процесса построения структурно-аналитической модели

по методологии DFD

|  |  |
| --- | --- |
| № этапа | Содержание этапа |
| 1 | Расчленение множества требований и организация их в основные функциональные группы |
| 2 | Идентификация внешних объектов, с которыми система должна быть связана |
| 3 | Идентификация основных видов информации, циркулирующей между системой и внешними объектами |
| 4 | Предварительная разработка контекстной диаграммы, на которой основные функциональные группы представляются процессами, внешние объекты – внешними сущностями, основные виды информации – потоками данных между процессами и внешними сущностями |
| 5 | Изучение предварительной контекстной диаграммы и внесение в неё изменений по результатам ответов на возникшие в процессе такого изучения вопросов по всем её частям |
| 6 | Построение контекстной диаграммы путём объединения всех процессов предварительной диаграммы в один процесс, а также группирования потоков |
| 7 | Формирование DFD первого уровня на базе процессов предварительной контекстной диаграммы |
| 8 | Проверка основных требований по DFD первого уровня |
| 9 | Декомпозиция каждого процесса текущей DFD с помощью детализирующей DFD или спецификации процессов |
| 10 | Проверка основных требований по DFD соответствующего уровня |
| 11 | Добавление определений новых потоков в словарь данных при каждом их появлении на диаграмме |
| 12 | Параллельное с процессом декомпозиции изучение требований (в т.ч. и вновь поступающих), разбиение их на элементарные и идентификация процессов или их спецификаций, соответствующих этим требованиям |
| 13 | После построения двух-трёх уровней, проведение ревизий с целью проверки корректности и улучшения понимания модели |
| 14 | Построение миниспецификации (а не простейшей диаграммы) в случае, если некоторую функцию сложно или невозможно выразить комбинацией процессов. |

Представленный выше набор этапов может быть детализирован в виде пошагового процесса построения DFD (табл. 4.11).

Табл. 4.11

Шаги  процесса построения DFD

|  |  |
| --- | --- |
| № шага | Содержание работ на шаге процесса |
| 1 | Процесс построения DFD начинается с создания так называемой основной диаграммы типа "звезда", на которой представлен моделируемый процесс и все внешние сущности, с которыми он взаимодействует. В случае сложного основного процесса он сразу представляется в виде декомпозиции на ряд взаимодействующих процессов. Критериями сложности в данном случае являются: наличие большого числа внешних сущностей, многофункциональность системы, ее распределенный характер. Внешние сущности выделяются по отношению к основному процессу. Для их определения необходимо выделить поставщиков и потребителей основного процесса, т.е. все объекты, которые взаимодействуют с основным процессом. На этом этапе описание взаимодействия заключается в выборе глагола, дающего представление о том, как внешняя сущность использует основной процесс или используется им. Например, основной процесс – "учет обращений граждан", внешняя сущность – "граждане", описание взаимодействия – "подает заявления и получает ответы". Этот шаг является принципиально важным, поскольку именно он определяет границы моделируемой системы и ее возможные подсистемы |
| 2 | Для всех внешних сущностей строится таблица событий, описывающая их взаимодействие с основным процессом. Таблица событий включает в себя наименование внешней сущности, событие, его тип (типичный для системы или исключительный, реализующийся при определенных условиях) и реакцию системы |
| 3 | Производится декомпозиция основного процесса на набор взаимосвязанных процессов, обменивающихся потоками данных. Сами потоки не конкретизируются, определяется лишь характер взаимодействия. Декомпозиция завершается, когда процесс становится простым, т.е.:  1.    процесс имеет два-три входных и выходных потока;  2.    процесс может быть описан в виде преобразования входных данных в выходные;  3.    процесс может быть описан в виде последовательного алгоритма. |
| 4 | Для простых процессов строится миниспецификация – формальное описание алгоритма преобразования входных данных в выходные.  Миниспецификация удовлетворяет следующим требованиям: для каждого процесса строится одна спецификация; спецификация однозначно определяет входные и выходные потоки для данного процесса; спецификация не определяет способ преобразования входных потоков в выходные; спецификация ссылается на имеющиеся элементы, не вводя новые; спецификация по возможности использует стандартные подходы и *операции*. |
| 5 | После декомпозиции основного процесса для каждого *подпроцесса* строится аналогичная таблица внутренних событий. |
| 6 | Выделяются **потоки данных**, которыми обмениваются процессы и внешние сущности. Простейший способ выделения потоков данных заключается в анализе таблиц событий. События преобразуются в потоки данных от инициатора события к запрашиваемому процессу, а реакции – в обратный поток событий. После построения входных и выходных потоков аналогичным образом строятся внутренние потоки. Для их выделения для каждого из внутренних процессов выделяются поставщики и потребители информации. Если поставщик или потребитель информации представляет процесс сохранения или запроса информации, то вводится хранилище данных, для которого данный процесс является интерфейсом. |
| 7 | После построения потоков данных диаграмма должна быть проверена на полноту и непротиворечивость. Полнота диаграммы обеспечивается, если в системе нет "повисших" процессов, не используемых в процессе преобразования входных потоков в выходные. Непротиворечивость системы обеспечивается выполнением наборов формальных правил о возможных типах процессов: на диаграмме не может быть потока, связывающего две внешние сущности – это взаимодействие удаляется из рассмотрения; ни одна сущность не может непосредственно получать или отдавать информацию в хранилище данных – хранилище данных является пассивным элементом, управляемым с помощью интерфейсного процесса; два хранилища данных не могут непосредственно обмениваться информацией – эти хранилища должны быть объединены. |

Преимущества и недостатки методологии DFD представлены в табл. 4.12.

Табл. 4.12

Преимущества и недостатки методологии DFD

|  |  |
| --- | --- |
| Преимущества | -       возможность однозначно определить внешние сущности, анализируя потоки информации внутри и вне системы;  -       возможность проектирования сверху вниз, что облегчает построение модели "как должно быть";  -       наличие спецификаций процессов нижнего уровня (миниспецификаций), что позволяет преодолеть логическую незавершенность функциональной модели и построить полную функциональную спецификацию разрабатываемой системы. |
| Недостатки | -         необходимость искусственного ввода управляющих процессов, поскольку управляющие воздействия (потоки) и управляющие процессы с точки зрения DFD ничем не отличаются от обычных;  -         отсутствие понятия времени, т.е. отсутствие анализа временных промежутков при преобразовании данных (все ограничения по времени должны быть введены в спецификациях процессов). |

Пример контекстной диаграммы, полученной в процессе проектирования системы, организующей работу банкомата по обслуживанию клиента по его кредитной карте, представлен на рис. 4.8. DFD первого уровня, детализирующая контекстный процесс, представлена на рис. 4.9.

Рис. 4.8 Контекстная диаграмма системы «Банкомат» в нотации Йодана

***Процесс 1.1*** (ПОЛУЧИТЬ ПАРОЛЬ) – осуществляет приём и проверку пароля клиента.

***Процесс 1.2*** (ПОЛУЧИТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ) – осуществляет приём и проверку запроса клиента на проведение необходимой ему банковской операции. Имеет на входе и выходе следующие потоки.

***Процесс 1.3*** (ОБРАБОТАТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ) – обеспечивает выполнение действий согласно запросу на обслуживание.

***Процесс 1.4*** (ОБРАБОТАТЬ КРЕДИТНУЮ КАРТУ) осуществляет считывание информации с кредитной карты.

На схеме хранилище ДАННЫЕ КРЕДИТНОЙ КАРТЫ для удобочитаемости показано дважды.

Процессы 1.1, 1.2 и 1.4 являются элементарными и не требуют детализации с помощью DFD второго уровня (они могут быть раскрыты с помощью спецификаций процессов). Процесс 1.3 может быть детализирован с помощью DFD второго уровня.

Модель банковской системы также содержит управляющий процесс и управляющие потоки, позволяющие описать её функционирование в реальном времени.

***Управляющий процесс 1.5*** (УПРАВЛЕНИЕ ОБСЛУЖИВАНИЕМ), получив информацию о том, что кредитная карта введена (поток ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА), вызывает выполнение процесса 1.1 (поток А). Получив информацию о введенном пароле (поток КОРРЕКТНЫЙ ПАРОЛЬ), процесс 1.5 информирует процесс 1.4 о необходимости удаления кредитной карты (поток УДАЛЁННАЯ КРЕДИТНАЯ КАРТА) и с помощью потока Т (ОБЕСПЕЧИТЬ ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ) вызывает выполнение процесса 1.2, а затем 1.3 (поток ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ).

Рис. 4.9  DFD первого уровня с комментариями

## *4.3 Методологии описания процессов  IDEF3, спецификаций управления STD и имитационного моделирования*

### 4.3.1 Сущность методологии IDEF3

DFD позволяют более эффективно и наглядно описать *процесс* документооборота. Однако для описания логики взаимодействия информационных потоков более подходит методология IDEF3, называемая также workflow diagramming, — методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы Workflow могут быть использованы в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например последовательность обработки заказа или события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

IDEF3 — это метод, имеющий основной целью дать возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе. Техника описания набора данных IDEF3 является частью структурного анализа. IDEF3 не ограничивает аналитика чрезмерно жесткими рамками синтаксиса, что может привести к созданию неполных или противоречивых моделей.

IDEF3 может быть также использован как метод создания процессов. IDEF3 дополняет IDEF0 и содержит все необходимое для построения моделей, которые в дальнейшем могут быть использованы для имитационного анализа.

Каждая работа в IDEF3 описывает какой-либо сценарий бизнес-процесса и может являться составляющей другой работы. Поскольку сценарий описывает цель и рамки модели, важно, чтобы работы именовались отглагольным существительным, обозначающим процесс действия, или фразой, содержащей такое существительное.

Точка зрения на модель должна быть документирована. Обычно это точка зрения человека, ответственного за работу в целом. Также необходимо документировать цель модели — те вопросы, на которые призвана ответить модель.

**Диаграмма** является основной единицей описания в IDEF3. Важно правильно построить диаграммы, поскольку они предназначены для чтения другими людьми (а не только автором).

Рассмотрим основные элементы диаграмм IDEF3.

**Единицы работы** — **Unit of Work (UOW)** — или просто **работы** (activity), являются центральными компонентами модели. В IDEF3 работы изображаются прямоугольниками с прямыми углами и имеют имя, выраженное отглагольным **существительным**, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы (например, "Изготовление изделия"). Часто имя существительное в имени работы меняется в процессе моделирования, поскольку модель может уточняться и редактироваться. Идентификатор работы присваивается при создании и не меняется никогда. Даже если работа будет удалена, ее идентификатор не будет вновь использоваться для других работ. Обычно номер работы состоит из номера родительской работы и порядкового номера на текущей диаграмме.

**Связи** показывают взаимоотношения работ. Все *связи* в IDEF3 однонаправлены и могут быть направлены куда угодно, но обычно диаграммы IDEF3 стараются построить так, чтобы связи были направлены слева направо. В IDEF3 различают **три типа стрелок, изображающих**связи (табл. 4.13).

Табл. 4.13

Типы стрелок, изображающих связи

|  |  |  |
| --- | --- | --- |
| Вид стрелки | Название стрелки | Описание связи |
|  | **Старшая (Prece-dence**) | сплошная линия, связывающая единицы работ (UOW). Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется. |
|  | **Отноше-ния (Relational Link)** | пунктирная линия, использующаяся для изображения *связей* между единицами работ (UOW), а также между единицами работ и объектами ссылок. Отношение показывает, что стрелка является альтернативой старшей стрелке или потоку объектов в смысле задания последовательности выполнения работ — работа-источник не обязательно должна закончиться, прежде чем работа-цель начнется. Более того, работа-цель может закончиться прежде, чем закончится работа-источник |
|  | **Потоки объектов (Object Flow)** | стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой |

Имя стрелки должно ясно идентифицировать отображаемый объект. Поток объектов имеет ту же семантику, что и старшая стрелка.

Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы, используются **перекрестки** (Junction). Различают перекрестки для слияния (Fan-in Junction) и разветвления стрелок (Fan-out Junction). Перекресток не может использоваться одновременно для слияния и для разветвления.

Существует 5 типов перекрестков. Смысл каждого типа приведен в табл. 4.14.

Таблица 4.14

Типы перекрестков

|  |  |  |  |
| --- | --- | --- | --- |
| **Обозначение** | **Наименование** | **Смысл в случае слияния стрелок (Fan-in Junction)** | **Смысл в случае разветвления стрелок (Fan-out Junction)** |
|  | Асинхронное И Asynchronous AND | Все предшествующие *процессы* должны быть завершены | Все следующие *процессы* должны быть запущены |
|  | Синхронное И Synchronous AND | Все предшествующие *процессы* должны быть завершены одновременно | Все следующие *процессы* запускаются одновременно |
|  | Асинхронное ИЛИ Asynchronous OR | Один или несколько предшествующих *процессов* должны быть завершены | Один или несколько следующих *процессов* должны быть запущены |
|  | Синхронное ИЛИ Synchronous OR | Один или несколько предшествующих *процессов* должны быть завершены одновременно | Один или несколько следующих *процессов* запускаются одновременно |
|  | Исключающее ИЛИ XOR (Exclusive OR) | Только один предшествую-щий *процесс* должен быть завершен | Только один следующий *процесс* запускается |

Все *перекрестки* на диаграмме нумеруются, каждый номер имеет префикс J. В отличие от IDEF0 и DFD в IDEF3 стрелки могут сливаться и разветвляться только через перекрестки.

**Объект ссылки** в IDEF3 выражает некую идею, концепцию или данные, которые нельзя связать со стрелкой, *перекрестком* или работой. Объект ссылки изображается в виде прямоугольника, похожего на прямоугольник работы

В качестве имени объекта ссылки можно использовать имя какой-либо стрелки с других диаграмм или имя сущности из модели данных. Объекты ссылки должны быть связаны с единицами работ или *перекрестками* пунктирными линиями. Официальная спецификация IDEF3 различает **три стиля объектов ссылок** — **безусловные** (unconditional), **синхронные** (synchronous) и **асинхронные** (asynchronous).

При внесении объектов ссылок помимо имени следует указывать тип объекта ссылки. Типы объектов ссылок приведены в табл. 4.15.

Таблица 4.15

Типы объектов ссылок

|  |  |
| --- | --- |
| **Тип объекта ссылки** | **Цель описания** |
| OBJECT | Описывает участие важного объекта в работе |
| GOTO | Инструмент циклического перехода (в повторяющейся последовательности работ), возможно на текущей диаграмме, но не обязательно. Если все работы цикла присутствуют на текущей диаграмме, цикл может также изображаться стрелкой, возвращающейся на стартовую работу. GOTO может ссылаться на перекресток |
| UOB (Unit of behaviour) | Применяется, когда необходимо подчеркнуть множественное использование какой-либо работы, но без цикла. Например, работа "Контроль качества" может быть использована в *процессе* "Изготовление изделия" несколько раз, после каждой единичной операции. Обычно этот тип ссылки не используется для моделирования автоматически запускающихся работ |
| NOTE | Используется для документирования важной информации, относящейся к каким-либо графическим объектам на диаграмме. NOTE является альтернативой внесению текстового объекта в диаграмму |
| ELAB (Elaboration) | Используется для усовершенствования графиков или их более детального описания. Обычно употребляется для детального описания разветвления и слияния стрелок на *перекрестках* |

В IDEF3 **декомпозиция** используется для детализации работ. Методология IDEF3 позволяет декомпозировать работу многократно, т.е. работа может иметь множество версий дочерних работ. Это позволяет в одной модели описать альтернативные потоки. Возможность множественной декомпозиции предъявляет дополнительные требования к нумерации работ. Так, номер работы состоит из номера родительской работы, версии декомпозиции и собственного номера работы на текущей диаграмме.

Декомпозиция диаграмм IDEF3предполагает взаимодействие автора (аналитика) и одного или нескольких экспертов предметной области.

Перед проведением сеанса экспертизы у экспертов предметной области должны быть документированные сценарии и рамки модели, для того чтобы понять цели декомпозиции. Обычно эксперт предметной области передает аналитику текстовое описание сценария. В дополнение к этому может существовать документация, описывающая интересующие процессы. Из этой информации аналитик должен составить предварительный список работ (отглагольные существительные, обозначающие процесс) и объектов (существительные, обозначающие результат выполнения работы), которые необходимы для перечисленных работ. В некоторых случаях целесообразно создать графическую модель для представления ее эксперту предметной области. Эксперты предметной области периодически проводят сеансы экспертизы моделей, построенных аналитиками.

На рис. 4.10 представлено описание *процесса* "Сборка настольных компьютеров" в методологии IDEF3.

Рис. 4.10  Пример описания процесса в методологии IDEF3

Поскольку разные фрагменты модели IDEF3 могут быть созданы разными группами аналитиков в разное время, IDEF3 поддерживает простую схему нумерации работ в рамках всей модели. Разные аналитики оперируют разными диапазонами номеров, работая при этом независимо. Пример выделения диапазона приведен в табл. 4.16.

Таблица 4.16

Диапазоны номеров работ

|  |  |
| --- | --- |
| **Аналитик** | **Диапазон номеров IDEF3** |
| Иванов | 1-999 |
| Петров | 1000-1999 |
| Сидоров | 2000-2999 |

В результате дополнения диаграмм IDEF0 диаграммами DFD и IDEF3 может быть создана смешанная модель, которая наилучшим образом описывает все стороны деятельности предприятия. Иерархию работ в смешанной модели можно увидеть в окне CASE-средства Model Explorer (рис. 4.11). Модели в нотации IDEF0 изображаются зеленым цветом, в IDEF3 — желтым,   в DFD — голубым.

Рис. 4.11  Представление смешанной модели в окне Model Explorer

### 4.3.2 Методы задания спецификаций процессов

***Спецификации процессов*** используются для описания функционирования процесса в случае отсутствия необходимости детализировать его с помощью DFD, то есть, если он достаточно мал и его описание может занимать до одной станицы текста. Фактически, спецификация процесса представляет собой описание алгоритма задачи, выполняемой процессом.

Спецификация процесса содержит номер и/или имя процесса, списки входных и выходных данных и тело (описание) процесса, являющееся спецификацией алгоритма или операции, трансформирующей входные потоки данных в выходные. Известно множество разнообразных методов, позволяющих задать тело процесса. Соответствующий язык может варьироваться от структурированного естественного языка или псевдокода до визуальных языков проектирования (типа FLOW-форм и диаграмм Насси-Шнейдермана) и формальных компьютерных языков.

Ниже представим некоторые наиболее часто используемые методы задания спецификации процесса.

#### 4.3.2.1 Структурированный естественный язык - псевдокод

***Псевдокод*** – это частично формализованная запись для наглядного текстового представления последовательности целенаправленных действий, а также схем, алгоритмов и программ. Псевдокод применяется для читабельности и строгого описания спецификации процесса. Он успешно заменяет обычные графические схемы алгоритмов, которые при числе блоков более двадцати совершенно не наглядны.

Псевдокод занимает промежуточное положение между алгоритмическими языками программирования (АЯП) высокого уровня и естественным языком. Он является разумной комбинацией строгости первого и читабельности второго.

При записи действий в форме псевдокода используется три основных канонических структуры алгоритмического языка программирования (табл. 4.17), но на них не накладываются строгие синтаксические ограничения алгоритмического языка программирования. В теории структурного программирования доказано, что с помощью канонических структур можно описать любой сколь угодно сложный алгоритм.

Таблица 4.17

Канонические структуры псевдокода

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Смысл** | **Формальная запись** |
| Простая последовательность действий (последовательная конструкция). | Задается повествовательными предложениями, обычно в неопределённой форме, повелительном наклонении (например, *Ввести очередную запись, Упорядочить массив* и т.п.). | ***ВЫПОЛНИТЬ****функция1*  ***ВЫПОЛНИТЬ****функция2*  ***…***  ***ВЫПОЛНИТЬ****функцияN* |
| Проверка условий или отношений (конструкция выбора). | задается словами: «***ЕСЛИ (условие)– ТО – ИНАЧЕ – ВСЕЕСЛИ***» (для английской записи: «***IF – THEN – ELSE – ENDIF***»). Здесь в месте проверяемого условия записывается предикат, в зависимости от истинности или ложности которого выполняются действия, указанные в ветвях «***ТО***» или «***ИНАЧЕ***». | ***ЕСЛИ****<условие>*  ***ТО***  ***ВЫПОЛНИТЬ****функция1*  ***ВЫПОЛНИТЬ****функция2*  ***…***  ***ВЫПОЛНИТЬ****функцияi*  ***ИНАЧЕ***  ***ВЫПОЛНИТЬ****функцияi+1*  ***ВЫПОЛНИТЬ****функцияi+2*  ***…***  ***ВЫПОЛНИТЬ****функцияN*  ***ВСЕЕСЛИ*** |
| Цикл (конструкция итерации). | Задается словами: «***ЦИКЛ – ДЛЯ – КОНЕЦЦИКЛ***» («***DO – FOR – ENDDO***») и «***ЦИКЛ – ПОКА – КОНЕЦЦИКЛ***» («***DO – WHILE – ENDDO***»). В заголовке конструкции записывается предикат, в зависимости от истинности которого либо выполняется тело цикла, либо цикл завершается. | ***ЦИКЛ   ПОКА****<условие>*  ***ВЫПОЛНИТЬ****функция1*  ***ВЫПОЛНИТЬ****функция2*  ***…***  ***ВЫПОЛНИТЬ****функцияN*  ***КОНЕЦЦИКЛ*** |

При записи псевдокода применяются символы, описанные в табл.4.18, и соглашения, приведенные в табл.4.19.

Таблица 4.18

Основные символы, применяемые при записи псевдокода

|  |  |
| --- | --- |
| № | Описание символа |
| 1 | Глаголы, ориентированные на действие и применяемые к объектам |
| 2 | Термины, определяемые на любой другой стадии проекта (например, задачи, процедуры, символы данных и т.п.) |
| 3 | Предлоги и союзы, используемые в логических отношениях |
| 4 | Общеупотребительные математические, физические и технические термины |
| 5 | Арифметические уравнения |
| 6 | Таблицы, диаграммы, графы и т.п. |
| 7 | Комментарии |

Таблица 4.19

Соглашения, применяемые при использовании псевдокода

|  |  |
| --- | --- |
| № | Описание соглашения |
| 1 | Логика процесса выражается в виде последовательности конструкций: последовательность, выбор, цикл |
| 2 | Ключевые слова должны быть выделены (заглавными буквами, подчеркиванием и т.п.) |
| 3 | Слова или фразы, определённые в словаре данных, должны быть написаны заглавными буквами |
| 4 | Глаголы должны быть активными, недвусмысленными и ориентированными на целевое действие (например, *Заполнить*, *Вычислить*, *Извлечь*, но не *Модернизировать*, *Обработать* и т.п.) |
| 5 | Логика процесса должна быть выражена чётко и недвусмысленно |

Направления использования псевдокода и достоинства метода приведены в табл. 4.20.

Таблица 4.20

Направления использования псевдокода и достоинства метода

|  |  |
| --- | --- |
| Направ-ления исполь-зования | -      Фиксация с нужным уровнем детализации алгоритма в процесс его разработки;  -      Формулировка заданий на кодирование программы на АЯП;  -      Описание логики программы;  -      Документирование алгоритмов с целью облегчения понимания текстов соответствующих программ на алгоритмическом языке программирования. |
| Достоин-ства метода | 1. Освобождает разработчика от необходимости следить за точным соблюдением формальных языковых правил;  2. Позволяет концентрировать внимание на содержании решаемых логических проблем;  3. Облегчает фиксацию и анализ способов решения проблем;  4. Анализ задания на языке псевдокода позволяет обосновано выбрать подходящий алгоритмический язык программирования и контролировать ход работ по программированию. |

Примеры псевдокодов: ПДП (язык проектирования и документирования программ), PSL, PDL.

#### 4.3.2.2 Таблицы и деревья решений

Псевдокод не удобен для описания некоторых типов преобразований. Например, если действие зависит от нескольких переменных, которые в совокупности могут продуцировать большое число комбинаций, то его описание будет слишком запутанным и с большим числом уровней вложенности. Для описания подобных действий традиционно используются таблицы и деревья решений.

***Таблица решений*** является средством компактного описания логических правил, используемых для выбора действий, которые нужно выполнить в соответствии с условием и порядком решения рассматриваемой задачи.

В таблице решений выделяют части, связанные с условиями, действиями и правилами.

Представим в виде таблицы решений логику рассуждений некоего субъекта S в связи с принятием решения о том, как провести ближайший выходной день. Соответствующее текстовое описание объекта анализа и представление логики рассуждений S в виде таблицы решений приведено на рис. 4.12.

Проведение S ближайшего выходного дня зависит от двух условий: состояния погоды и программы ТВ. В число возможных вариантов входит: рыбалка, просмотр ТВ, посещение кино, семейный разговор. S придерживается следующих правил:

На рыбалку S идет независимо от программы по ТВ, если будет хорошая погода;

Он останется дома, если погода будет плохой, а по ТВ будут интересные передачи;

Он пойдет в кино, если погода будет плохой и по ТВ будут неинтересные передачи;

Если S останется дома, он в первую очередь займётся обсуждением проблем в кругу семьи, а затем будет смотреть ТВ.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Правила | | | |
|  |  |  | R1 | R2 | R3 | R4 |
| Условия | C1 | Погода хорошая | Да | Да | Нет | Нет |
| C2 | ТВ-программы интересные | Да | Нет | Да | Нет |
| D1 | Рыбалка | х | х |  |  |
| Действия | D2 | Просмотр ТВ-программ |  |  | х2 |  |
| D3 | Посещение кинотеатра |  |  |  | х |
| D4 | Обсуждение проблем в кругу семьи |  |  | х1 |  |

Рис. 4.12 Описание объекта и соответствующая таблица решений с ограниченными входами

Существует *два способа представления* таблиц решений:

горизонтальное представление, когда условия и действия записываются по горизонтали, а правила читаются по вертикали;

вертикальное представление, когда условия и действия записываются по вертикали, а правила читаются по горизонтали.

Различают таблицы решений с ограниченными и расширенными входами.

Таблица решений ***с ограниченными входами*** - это таблица, в которой строки условий в позициях для записи правил заполнены словами «*Да*», «*Нет*» и «=». А строки действий в соответствующих позициях заполняются знаками «x» (возможно с указанием номера, определяющего последовательность действий).

*Знак «=»* означает, что:

соответствующее этой строке условие не может влиять на рассматриваемое правило;

соответствующее этой строке условие несущественно.

Таким образом, знак «=» означает «*Да или Нет*».

С учётом изложенного таблица для данного примера может быть редуцирована (рис.4.13).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  | Правила | | |
|  |  |  | R1,2 | R3 | R4 |
| Условия | C1 | Погода хорошая | Да | Нет | Нет |
| C2 | ТВ-программы интересные | = | Да | Нет |
| D1 | Рыбалка | х |  |  |
| Действия | D2 | Просмотр ТВ-программ |  | х2 |  |
| D3 | Посещение кинотеатра |  |  | х |
| D4 | Обсуждение проблем в кругу семьи |  | х1 |  |

Рис. 4.13 Редуцированная таблица решений с ограниченными входами

Таблица решений ***с расширенными входами*** в строке, соответствующей условиям содержит количественные или качественные данные, характеризующие условие. А строки, соответствующие действиям, содержат такие же данные, характеризующие действия (рис. 4.14).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Состояние погоды | Хорошая | Хорошая | Плохая | Плохая |
| Содержание ТВ-программ | Интересные | Неинтересные | Интересные | Неинтересные |
| Времяпровождение вне дома | Рыбалка | Рыбалка | Нет | Поход в кино |
| Времяпровождение в кругу семьи | Нет | Нет | Обсуждение семейных проблем;  Просмотр ТВ | Нет |

Рис. 4.14 Таблица решений с расширенными входами

Таблицы решений с расширенными входами часто являются более понятными, т.к. они менее громоздки. Таблицы решений с ограниченными входами лучше подготовлены к программированию. Возможно взаимное преобразование таблиц с разными типами входов.

Если задача оказывается слишком громоздкой, чтобы быть представленной с помощью одной таблицы решений, часто оказывается возможным разложить её на более простые задачи, для которых могут быть построены частные таблицы решений. В результате получается несколько связанных между собой таблиц решений. Небольшая таблица решений строится, обрабатывается и корректируется гораздо легче, чем одна большая таблица. Связи между частными сцепленными таблицами реализуются с помощью действий следующих двух типов (подобно обращению к подпрограмме без возврата, и с возвратом в вызывающую процедуру): «Перейти к таблице i»; «Выполнить таблицу i».

Построение таблиц решений рекомендуется осуществлять по шагам, описанным в табл. 4.21. Достоинства таблиц решений приведены в табл.4.22.

Таблица 4.21

Шаги построения таблиц решений

|  |  |
| --- | --- |
| № | Описание шага |
| 1 | Идентифицировать все условия или переменные в спецификации; идентифицировать все значения, которые каждая переменная может иметь |
| 2 | Вычислить число комбинаций условий (если все условия являются бинарными, то существует 2N комбинаций N переменных) |
| 3 | Идентифицировать каждое из возможных действий, которое может вызываться в спецификации |
| 4 | Построить пустую таблицу, включающую все возможные условия и действия, а также номера комбинаций условий |
| 5 | Выписать и занести в таблицу все возможные комбинации условий |
| 6 | Редуцировать комбинации условий |
| 7 | Проверить каждую комбинацию условий и идентифицировать соответствующие выполняемые действия |
| 8 | Выделить комбинации условий, для которых спецификация не указывает список выполняемых действий |
| 9 | Обсудить построенную таблицу |

Таблица 4.22

Достоинства таблиц решений

|  |  |
| --- | --- |
| № | Описание достоинства |
| 1 | Точное определение логики принятия изучаемых решений и ее понятное и исчерпывающее описание |
| 2 | Простота выявления ошибок в логике и пропущенных вариантов |
| 3 | Простота изменений в логике выбора решений и наглядное представление последствий этих изменений для всей логики в целом |
| 4 | Возможность реализации разбиения проблемы на подпроблемы |
| 5 | Легкость взаимопонимания между специалистами, т.к. задачи, описанные с помощью таблиц решений, как правило, не требуют пояснений (они формируются в доступном для восприятия виде и понятны всем) |

Вариантом таблицы решений является ***дерево решений***, позволяющее взглянуть на процесс условного выбора с позиции схемы. Дерево решений для вышерассмотренного примера приведено на рис. 4.15.

Рис. 4.15 Дерево решений

Обычно дерево решений используется при малом числе действий и когда не все комбинации условий возможны, а таблицы решений – при большом числе действий и когда возможно большинство комбинаций условий.

#### 4.3.2.3 Визуальные языки проектирования спецификаций

Визуальные языки проектирования спецификаций являются относительно новой оригинальной методикой разработки спецификаций процесса. Они базируются на основных идеях структурного программирования и позволяют определять потоки управления с помощью специальных иерархически-организованных схем.

Одним из наиболее известных подходов к визуальному проектированию спецификаций является подход, основанный на использовании FLOW-форм.

Символы FLOW-форм приведены на рис. 4.16. Каждый символ FLOW-формы имеет вид прямоугольника и может быть вписан в любой внутренний прямоугольник любого другого символа. Символы помечаются с помощью предложений на естественном языке или с использованием математической нотации. Каждый символ является блоком обработки. Каждый прямоугольник внутри любого символа также является блоком обработки.

Рис. 4.16 Символы FLOW-форм: а) Последовательная обработка;

б) Условный выбор; в) CASE-выбор; г) Цикл типа 1; д) Цикл типа 2;

е) Цикл типа 3

Приведём пример использования данного подхода при проектировании спецификации процесса, обеспечивающего упорядочивание определённым образом элементов массива и являющегося фрагментом алгоритма сортировки «Методом поплавка» (рис.4.17).

|  |  |  |
| --- | --- | --- |
| **FOR j = 1 TO n** | | |
| **DO** | … сравнить и поменять местами | |
| **IF a[j] > a[j+1]** | |
| **THEN** | … поменять местами |
| **temp = a[j]** |
| **a[j] = a[j+1]** |
| **a[j+1] = temp** |

Рис. 4.17  Использование FLOW-формы при проектировании спецификации фрагмента алгоритма сортировки элементов массива «Методом поплавка»

Дальнейшее развитие FLOW-формы получили в ***диаграммах Насси-Шнейдермана***, на которых символы последовательной обработки и цикла изображаются так же, как соответствующие символы FLOW-форм. В символах условного и CASE-выбора собственно условие располагается в верхнем треугольнике, выбираемые варианты – на нижних сторонах треугольника, а блоки обработки – под выбираемыми вариантами.

Такая диаграмма для вышерассмотренного примера представлена на рис. 4.18.

|  |  |  |
| --- | --- | --- |
| **FOR j = 1 TO n** | | |
| **DO** | … сравнить и поменять местами | |
| **IF a[j]  >**        Нет | **a[j+1]**                Да |
|  | … поменять местами |
| **temp = a[j]** |
| **a[j] = a[j+1]** |
| **a[j+1] = temp** |

Рис. 4.18  Использование диаграмм Насси-Шнейдермана при проектировании спецификации фрагмента алгоритма сортировки элементов массива «Методом поплавка»

### 4.3.3 Спецификации управления

#### 4.3.3.1 Понятие диаграммы переходов состояний

***Спецификации управления*** предназначены для моделирования и документирования аспектов системы, зависящих от времени или реакции на событие. Они позволяют осуществлять декомпозицию управляющих процессов и описывают отношения между входными и выходными управляющими потоками на управляющем процессе-предке. Для этой цели обычно используются диаграммы переходов состояний, то есть STD.

С помощью STD можно моделировать последующее функционирование системы на основе её предыдущего и текущего функционирования. Моделирование системы в любой заданный момент времени предполагает, что она находится в одном из конечного множества состояний. С течением времени она может изменить своё состояние, при этом переходы между состояниями должны быть  точно определены.

STD состоитиз следующих объектов: СОСТОЯНИЕ, НАЧАЛЬНОЕ СОСТОЯНИЕ, ПЕРЕХОД, УСЛОВИЕ, ДЕЙСТВИЕ, описанных в табл. 4.23.

Таблица 4.23

Описание объектов STD

|  |  |
| --- | --- |
| Объект | Описание объекта |
| ***СОСТОЯНИЕ*** | Может рассматриваться как условие устойчивости для системы. Находясь в определённом состоянии, мы имеем достаточно информации о прошлой истории системы, чтобы определить очередное состояние в зависимости от текущего входного события. Имя состояния должно отражать реальную ситуацию, в которой находится система (например, НАГРЕВАНИЕ, ОХЛАЖДЕНИЕ, ОЖИДАНИЕ и т.п.) |
| ***НАЧАЛНОЕ СОСТОЯНИЕ*** | Узел STD, являющийся стартовой точкой функционирования. Для начального системного перехода STD имеет ровно одно начальное состояние, соответствующее состоянию системы после её инсталляции, но перед началом реальной обработки, а также любое (конечное) множество завершающих состояний. Начальное состояние на диаграмме изображается в верхней части и должно иметь входной переход, изображаемый потоком из подразумеваемого стартового узла. |
| ***ПЕРЕХОД*** | Определяет перемещение моделируемой системы из одного состояния в другое в зависимости от УСЛОВИЯ. При этом имя перехода идентифицирует событие, являющееся причиной перехода и управляющее им. Это событие обычно состоит из управляющего потока (сигнала), возникающего как во внешнем мире, так и внутри моделируемой системы при выполнении некоторого условия (например, КНОПКА НАЖАТА, СЧЁТЧИК = 777). Таким образом, кроме УСЛОВИЯ с переходом может связываться ДЕЙСТВИЕ или ряд действий, выполняющихся в случае, когда переход имеет место. |
| ***УСЛОВИЕ*** | Представляет собой событие (события), вызывающее переход и идентифицируемое именем перехода. Если в условии участвует входной управляющий поток управляющего процесса, то имя этого потока должно быть заключено в кавычки (например, “ПАРОЛЬ”=111, где ПАРОЛЬ – входной поток). Кроме того, для спецификации A, T и E/D-потоков или переключаемого процесса имя также должно заключаться в кавычки (например, А: “ПОЛУЧИТЬ ПАРОЛЬ”, что означает: активизировать процесс ПОЛУЧИТЬ ПАРОЛЬ). Фактически, условие есть некоторое внешнее или внутреннее событие, которое система способна обнаружить и на которое она должна отреагировать, определённым образом изменяя своё состояние. Условия возбуждают выполнение перехода и называются иначе стимулирующими событиями. Действия или отклики на события привязываются к переходам и записываются под соответствующим условием. |
| ***ДЕЙСТВИЕ*** | Операция, которая может иметь место при выполнении перехода. Если действие необходимо для выбора выходного управляющего потока, то имя этого потока должно заключаться в кавычки (например, “ВВЕДЁННАЯ КРЕДИТНАЯ КАРТА” = TRUE, где ВВЕДЁННАЯ КАРТА – выходной поток). При изменении состояния система обычно выполняет одно или более действий (выводит сообщение, выполняет вычисления и т.п.). Таким образом, действие представляет собой отклик, посылаемый во внешнее окружение, или вычисление, результаты которого выполняются в системе для того, чтобы обеспечить реакцию на некоторые из планируемых в будущем события |

Принцип построения STD приведен на рис. 4.19, а пример построения STD для задачи с банкоматом - на рис. 4.20.

Рис. 4.19 Принцип построения STD

Диаграмма содержит два состояния: ОЖИДАНИЕ и ОБРАБОТКА.

Переход из состояния ОЖИДАНИЕ в состояние ОБРАБОТКА осуществляется при условии ввода кредитной карты. При этом выполняется действие по запуску соответствующего процесса ПОЛУЧИТЬ ПАРОЛЬ. Отметим, что для запуска используется А-поток, обеспечивающий непрерывность процесса, то есть, возможность повторного ввода пароля.

Переход из состояния ОБРАБОТКА в состояние ОЖИДАНИЕ может осуществляться двумя различными способами:

1.     При условии трёхкратного ввода неверного пароля (согласно условиям задачи) кредитная карта удаляется из системы, при этом система переходит в режим ОЖИДАНИЕ очередного клиента.

2.     При условии КОРРЕКТНЫЙ ПАРОЛЬ выполняются действия по обеспечению требуемого сервиса (последовательное включение соответствующих процессов) и удалению кредитной карты, после чего система переходит в режим ОЖИДАНИЕ очередного клиента.

Рис. 4.20 Диаграмма переходов состояний для банковской задачи

#### 4.3.3.2 Правила и способы построения STD

При построении STD рекомендуется следовать 4-м правилам (табл. 4.24).

Таблица 4.24

Правила построения STD

|  |  |
| --- | --- |
| № | Описание правила |
| 1 | Строить STD на как можно более высоком уровне детализации DFD |
| 2 | Строить как можно более простые STD |
| 3 | По возможности детализировать STD |
| 4 | Использовать те же принципы именования состояний, событий и действий, что и при именовании процессов и потоков |

Применяются два способа построения STD:

-      идентификация всех возможных состояний и дальнейшее исследование всех небессмысленных связей (переходов) между ними;

-      сначала строится начальное состояние, а затем следующее за ним и т.д.

Результат (по обоим способам) – предварительная STD, для которой затем осуществляется ***контроль состоятельности***, заключающийся в ответе на вопросы, приведенные в табл.4.25.

Таблица 4.25

Вопросы для контроля состоятельности STD

|  |  |
| --- | --- |
| № | Вопрос |
| 1 | Все ли состояния определены и имеют уникальное имя? |
| 2 | Все ли состояния достижимы? |
| 3 | Все ли состояния имеют выход? |
| 4 | Реагирует ли система для каждого состояния соответствующим образом на все возможные условия (особенно на ненормальные)? |
| 5 | Все ли входные (выходные) потоки управляющего процесса отражены в условиях (действиях) на STD? |

#### 4.3.3.3 Таблицы и матрицы переходов состояний

В ситуации, когда число состояний и/или переходов велико для проектирования спецификаций управления могут использоваться таблицы и матрицы переходов состояний. Обе эти нотации позволяют зафиксировать ту же информацию, что и STD, но в другой форме.

На рис.4.21 приведен пример такой таблицы, соответствующей диаграмме переходов состояний для банковской задачи (рис. 4.20).

|  |  |  |  |
| --- | --- | --- | --- |
| *Текущее состояние* | *Условие* | *Действие* | *Следующее состояние* |
| НАЧАЛЬНОЕ СОСТОЯНИЕ | АКТИВИРУЕТСЯ КАЖДЫЙ РАЗ | - | ОЖИДАНИЕ |
| ОЖИДАНИЕ | ВВЕДЁННАЯ КРЕДИТНАЯ КАРТА | ПОЛУЧИТЬ  ПАРОЛЬ | ОБРАБОТКА |
| ОБРАБОТКА | НЕКОРРЕКТНЫЙ ПАРОЛЬ | УДАЛИТЬ КРЕДИТНУЮ КАРТУ | ОЖИДАНИЕ |
| ОБРАБОТКА | КОРРЕКТНЫЙ ПАРОЛЬ | ОБЕСПЕЧИТЬ ТРЕБУЕМЫЙ СЕРВИС И УДАЛИТЬ КАРТУ | ОЖИДАНИЕ |

Рис. 4.21 Таблица переходов состояний для банковской задачи

***Матрица переходов состояний*** содержат по вертикали перечень состояний системы, по горизонтали – список условий. Каждый её элемент содержит список действий и имя состояния, в которое осуществляется переход. Возможен другой вариант данной нотации: по вертикали указываются состояния, из которых осуществляется переход, а по горизонтали – состояния в которые осуществляется переход. При этом каждый элемент матрицы содержит соответствующие условия и действия, обеспечивающие переход из «вертикального» состояния в «горизонтальное».

### 4.3.4 Роль и место имитационного моделирования в проектировании АС

**Оценочные аспекты** моделирования предметной области связаны с разрабатываемыми показателями эффективности автоматизируемых процессов.

Метод функционального моделирования позволяет оптимизировать существующие на предприятии бизнес-процессы, однако для оптимизации конкретных технологических операций функциональной модели может быть недостаточно. В этом случае целесообразно использовать имитационное моделирование.

Имитационное моделирование – это метод, позволяющий строить модели, учитывающие время выполнения операций, и обеспечивающий наиболее полные средства анализа динамики бизнес-процессов. Имитационные модели описывают не только потоки сущностей, информации и управления, но и различные метрики. Полученную модель можно "проиграть" во времени и получить статистику происходящих процессов так, как это было бы в реальности. В имитационной модели изменения процессов и данных ассоциируются с событиями. "Проигрывание" модели заключается в последовательном переходе от одного события к другому.

Связь между имитационными моделями и моделями процессов заключается в возможности преобразования модели процессов в имитационную модель. Имитационная модель дает больше информации для анализа системы, в свою *очередь* результаты такого анализа могут быть причиной модификации модели *процессов*.

Одним из наиболее эффективных инструментов имитационного моделирования является система ARENA, разработанная фирмой System Modeling Corporation. Система позволяет строить имитационные модели, проигрывать их и анализировать результаты.

Основные элементы, которые включает имитационная модель, описаны в табл. 4.26.

Таблица 4.26

Основные элементы, которые включает имитационная модель

|  |  |
| --- | --- |
| Элемент | Описание |
| Источники | Это элементы, от которых в модель поступает информация или объекты. По смыслу они близки к понятиям "внешняя ссылка" на DFD-диаграммах или "объект ссылки" на диаграммах IDEF3. Скорость поступления данных или объектов от *источника* обычно задается статистической функцией. |
| Стоки | Это устройства для приема информации или объектов. |
| Очереди | Понятие *очереди* близко к понятию *хранилища данных* на DFD-диаграммах — это место, где объекты ожидают обработки. Время обработки объектов в разных работах может быть разным. В результате перед некоторыми работами могут накапливаться объекты, ожидающие своей *очереди*. Часто целью имитационного моделирования является минимизация количества объектов в *очередях*. |
| Процессы | Это аналог работ в модели *процессов*. В имитационной модели может быть задана производительность *процессов* |

Функциональные и имитационные модели тесно взаимосвязаны и эффективно дополняют друг друга. Целесообразно сначала строить функциональную модель, а на ее основе — имитационную

## *4.4 Методологии моделирования информационного обеспечения IDEF1, IDEF1X, ERD*

### 4.4.1 Основные положения, уровни и методологии моделирования данных

Разработка баз данных (БД) выполняется с помощью моделирования данных. Вопросы моделирования данных достаточно подробно рассмотрены в курсе «Базы и банки данных». Здесь мы лишь кратко вспомним основные положения, связанные с проектированием информационного обеспечения АС.

При проектировании БД проектировщик оперирует с описаниями данных и отношений между ними. Описания бывают двух типов: логические (инфологическая или концептуальная модель БД) и физические (даталогическая. модель БД). Логическое описание данных не зависит от СУБД и указывает на то, в каком виде данные представляются прикладному программисту или пользователю данных. Физическое описание данных определяет способы физической записи данных во внешней памяти с учетом специфики выбранной СУБД.

Различают три уровня логической модели, отличающихся по глубине представления информации о данных (табл. 4.27):

диаграмма сущность-связь (Entity Relationship Diagram, ERD);

модель данных, основанная на ключах (Key Based model, KB);

полная атрибутивная модель (Fully Attributed model, FA).

Таблица 4.27

Три уровня логической модели данных

|  |  |
| --- | --- |
| Уровень | Описание |
| Диаграмма сущность-связь (Entity Relationship Diagram, ERD) | Представляет собой модель данных верхнего уровня. Она включает сущности и взаимосвязи, отражающие основные бизнес-правила предметной области. Такая диаграмма не слишком детализирована, в нее включаются основные сущности и связи между ними, которые удовлетворяют основным требованиям, предъявляемым к АС. Диаграмма сущность-связь может включать связи "многие-ко-многим" и не включать описание ключей. Как правило, ERD используется для презентаций и обсуждения структуры данных с экспертами предметной области. |
| Модель данных, основанная на ключах (Key Based model, KB) | Более подробное представление данных. Она включает описание всех сущностей и первичных ключей и предназначена для представления структуры данных и ключей, которые соответствуют предметной области. |
| Полная атрибутивная модель (Fully Attributed model, FA) | Наиболее детальное представление структуры данных: представляет данные в третьей нормальной форме и включает все сущности, атрибуты и связи |

Физическая модель содержит всю информацию, необходимую для реализации конкретной БД. Различают два уровня физической модели:

трансформационную модель;

модель СУБД.

Трансформационная модель содержит информацию для реализации отдельного проекта, который может быть частью общей АС и описывать подмножество предметной области. Данная модель позволяет проектировщикам и администраторам БД лучше представить, какие объекты БД хранятся в словаре данных, и проверить, насколько физическая модель удовлетворяет требованиям к АС.

Модель СУБД автоматически генерируется из трансформационной модели и является точным отображением системного каталога СУБД.

По умолчанию CASE-средство проектирования генерирует имена таблиц, их полей и индексов по шаблону на основе имен соответствующих сущностей, атрибутов и ключей логической модели, которые в дальнейшем могут быть откорректированы вручную.

Наиболее распространенным средством моделирования данных являются диаграммы "сущность-связь" (ERD). С помощью ERD осуществляется детализация накопителей данных DFD – диаграммы, а также документируются информационные аспекты бизнес-системы.

Наиболее распространенными методологиями для построения ER-диаграмм являются IDEF1 (IDEF1Х) и методология Баркера.

Методология IDEF1 основана на подходе (нотации) Чена и используются в ряде распространенных CASE-средств (в частности, ERwin, Design/IDEF). Краткие определения методологий IDEF1 и IDEF1X даны в табл. 4.28

Таблица 4.28

Краткие определения методологий IDEF1 и IDEF1X

|  |  |
| --- | --- |
| Название | Определение |
| *IDEF1* - Information Modeling | Методология моделирования информационных потоков внутри системы, позволяющая отображать и анализировать их структуру и взаимосвязи. Позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме |
| *IDEF1X* ( *IDEF1* Extended) - Data Modeling | Методология проектирования реляционных баз данных. Заключается в построении моделей данных типа "сущность-связь" (ERD) в нотации этого стандарта. Разработана с учетом таких требований, как простота для изучения и возможность автоматизации |

Методология Баркера основана на нотации, предложенной автором, и используется в CASE -средстве Oracle Designer.

### 4.4.2 Сущности, отношения и связи в нотации Чена

Нотация Чена (в IDEF1 и IDEF1Х) представляет богатый набор средств моделирования данных, включая собственно ERD, а также диаграммы атрибутов и диаграммы декомпозиции или категоризации сущностей. Эти диаграммные техники используются прежде всего для проектирования реляционных БД (но могут с успехом применяться для сетевых и иерархических БД).

Основные компоненты ERD — это сущности, атрибуты, отношения и связи (табл. 4.29).

Таблица 4.29

Определение компонентов ERD

|  |  |
| --- | --- |
| Название | Определение |
| ***СУЩНОСТЬ*** | Представляет собой множество экземпляров реальных или абстрактных объектов (людей, событий, состояний, идей, предметов и т.д.), обладающих общими атрибутами или характеристиками. Любой объект системы может быть представлен только одной сущностью, которая должна быть уникально идентифицирована. Имя сущности должно отражать тип или класс объектов, а не его конкретный экземпляр (например, АЭРОПОРТ, а не МИНСК-1). Сущности соответствует таблица БД |
| ***АТРИБУТ*** | Выражает определенное свойство объекта. Каждая сущность обладает одним или несколькими атрибутами, которые однозначно идентифицируют каждый экземпляр сущности. При этом любой атрибут может быть определён как ключевой. Экземпляру сущности соответствует строка в таблице БД, атрибуту — колонка таблицы |
| ***ОТНОШЕНИЕ*** | Выражает взаимозависимость двух и более сущностей. Именование отношения осуществляется с помощью грамматического оборота глагола (например, ИМЕЕТ, ОПРЕДЕЛЯЕТ и т.д.) |
| ***СВЯЗЬ*** | В самом общем виде - графическое представление отношения между двумя и более сущностями |

Сущности представляют собой базовые типы информации, хранимой в БД, а отношения показывают как эти типы данных взаимосвязаны друг с другом. Введение подобных отношений преследует *две основные цели*:

Обеспечение хранения информации в единственном месте (даже если она используется в различных комбинациях);

Использование этой информации различными приложениями.

Разновидности сущностей и отношений определенны в табл. 4.30, а соответствующие им символы ERD представлены на рис.4.22.

Таблица 4.30

Разновидности сущностей и отношений

|  |  |
| --- | --- |
| Компонент | Определение |
| ***СУЩНОСТЬ*** | ***Независимая сущность*** представляет независимые данные, которые всегда присутствуют в системе. При этом отношения с другими сущностями могут как существовать, так и отсутствовать.  ***Зависимая сущность*** представляет данные, зависящие от других сущностей в системе, поэтому они должны всегда иметь отношения с другими сущностями.  ***Ассоциированная сущность*** представляет данные, которые ассоциируются с отношениями между двумя и более сущностями. |
| ***ОТНОШЕНИЕ*** | ***Неограниченное (обязательное) отношение*** представляет собой безусловное отношение, то есть отношение, которое всегда существует до тех пор, пока существуют относящиеся к делу сущности.  ***Ограниченное (необязательное) отношение*** представляет собой условное отношение между сущностями.  ***Существенно ограниченное отношение*** используется, когда соответствующие сущности взаимно условно зависимы в системе. |

Рис. 4.22 Символы ERD: а) Независимая сущность; б) Зависимая сущность; в) Частично зависимая (ассоциированная) сущность; г) Неограниченное отношение; д) Ограниченное отношение; е) Существенно ограниченное отношение

Для идентификации требований, в соответствии с которыми сущности вовлекаются в отношения, используются ***связи***. Каждая связь соединяет сущность и отношение. Она может быть направлена только от отношения к сущности и может иметь ***ЗНАЧЕНИЕ,***характеризующее её тип (рис. 4.23).

|  |  |  |
| --- | --- | --- |
|  | « 0 или 1» | - буква Z на схеме |
|  | « 0 или более » | - буква  N на схеме |
|  | « 1 » | - цифра 1 на схеме |
|  | « 1 или более » | - буква P на схеме |
|  | « p : q » | - диапазон чисел на схеме |

Рис. 4.23 Отображение возможных значений связей

Практика показала, что для большинства приложений достаточно использовать следующие *типы отношений*: « 1 : 1 »; « 1 : n »; « m : n ».

Последнее отношение в дальнейшем должно быть преобразовано в комбинацию отношений первых двух типов (возможно, с добавлением вспомогательных сущностей и введением новых отношений).

Наряду с термином***ЗНАЧЕНИЕ*** используется термин ***МОЩНОСТЬ*** связей (Cardinality), который также служит для обозначения отношения числа экземпляров родительской *сущности* к числу экземпляров дочерней.

На рис. 4.24 представлена диаграмма «сущность - связь», демонстрирующая отношение между объектами банковской системы (см. п.4.2.3).

Рис. 4.24  Диаграмма «сущность - связь», демонстрирующая отношения между объектами банковской системы

### 4.4.3 Диаграммы атрибутов и категоризации сущностей в нотации Чена

Детализация сущностей осуществляется с помощью ***диаграмм атрибутов***, которые раскрывают ассоциированные с сущностью атрибуты. Диаграмма атрибутов состоит из детализируемой сущности, соответствующих атрибутов и доменов, описывающих области значений атрибутов. На диаграмме каждый атрибут представляется в виде связи между сущностью и соответствующим доменом, являющимся графическим представлением множества возможных значений атрибута. Все атрибутные связи имеет значения на своем окончании. Для идентификации ключевого атрибута используется подчеркивание имени атрибута.

На рис. 4.25 представлена диаграмма атрибутов, детализирующая сущность КРЕДИТНАЯ КАРТА.

Рис. 4.25 Диаграмма атрибутов, детализирующая сущность КРЕДИТНАЯ КАРТА

Сущность может быть представлена в виде двух или более ***сущностей-категорий***, каждая из которых имеет общие атрибуты и/или отношения, которые определяются однажды на верхнем уровне и наследуются на нижнем. Таким образом, сущность может быть разделена на компоненты. Сущности-категории могут иметь и свои собственные атрибуты и/или отношения, а также могут быть декомпозированы своими сущностями-категориями на следующем уровне. Расщепляемая на категории сущность называется ***общей сущностью***. Отметим, что на промежуточных уровнях декомпозиции одна и та же сущность может быть как общей, так и сущностью-категорией.

Для демонстрации декомпозиции сущности на категории используются ***диаграммы категоризации***, которые содержат общие сущности, две или более сущности-категории и специальный узел – дискриминатор, который описывает способы декомпозиции сущности (рис. 4.26).

Рис. 4.26 Диаграмма категоризации

Существует четыре возможных типа дискриминатора,которые описаны на рис. 4.27.

1.     ***Полное обязательное вхождение*** (E/M) – сущность должна быть одной и только одной из следуемых категорий. Для данного примера это означает, что ПРЕПАДАВАТЕЛЕМ является ФИЗИК (Ф) или ХИМИК (Х), или МАТЕМАТИК (М);

2.     ***Полное необязательное вхождение*** (Е/О) – сущность может быть одной и только одной из следуемых категорий. В примере это означает, что ПРЕПОДАВАТЕЛЕМ является ФИЗИК (Ф) или ХИМИК (Х), или МАТЕМАТИК (М) или преподаватель другой дисциплины (например, ИСТОРИК);

3.     ***Неполное обязательное вхождение***(I/M) -  сущность должна быть по крайней мере одной из следуемых категорий. В примере это означает, что ПРЕПАДАВАТЕЛЕМ является ФИЗИК (Ф) или ХИМИК (Х), или МАТЕМАТИК (М). При этом имеется возможность задавать ситуацию, когда  ПРЕПАДАВАТЕЛЕМ является одновременно и ФИЗИК (Ф) и ХИМИК (Х);

4.     ***Неполное необязательное вхождение*** (I/О) - сущность может быть по крайней мере одной из следуемых категорий. В примере это означает, что ПРЕПАДАВАТЕЛЕМ является ФИЗИК (Ф), или ХИМИК (Х), или МАТЕМАТИК (М), или преподаватель другой дисциплины (например, ИСТОРИК). При этом имеется возможность задавать ситуацию, когда  ПРЕПАДАВАТЕЛЕМ является одновременно и ФИЗИК (Ф) и ХИМИК (Х).

Используются следующие обозначения: E – EXCLUSIVE, I – INCLUSIVE, M – MANDATORY, O – OPTIONAL

Рис. 4.27 Типы дискриминатора: а) E/M; б) E/O; в) I/M; г) I/O

### 4.4.4 Нотация Баркера

Дальнейшее развитие ER-подход получил в работах Баркера, предложившего оригинальную нотацию, которая позволила на верхнем уровне интегрировать предложенные Ченом средства описания моделей.

В нотации Баркера используется только один тип диаграмм – ERD.

Сущность на ERD представляется прямоугольником любого размера, содержащим внутри имя сущности, а также список имён атрибутов (возможно, не полный) и указатели ключевых атрибутов (знак # перед именем атрибута).

Все связи являются бинарными и представляются линиями, состоящими из двух частей, соединяющих сущности. Для каждой части должно быть определено имя, степень множественности (один или много объектов участвуют в связи) и степень обязательности (обязательна или не обязательна связь между сущностями). Для множественной связи линия присоединяется к прямоугольнику сущности в трёх точках, а для одиночной связи – в одной точке. При обязательной связи рисуется непрерывная линия до середины связи, при необязательной – пунктирная. Отметим, что понятия «категория» и «общая сущность» заменяются Баркером на эквивалентные понятия «подтип» и «супертип» соответственно.

На рис. 4.28 приведен фрагмент ERD банковской задачи в нотации Баркера.

Читается связь отдельно для каждой части. На рисунке показано, как сущность КЛИЕНТ связывается с сущностью КРЕДИТНАЯ КАРТА, и наоборот. При этом необходимо учитывать степень обязательности выбранной части связи, для этого используются слова «ДОЛЖЕН (БЫТЬ)» или «МОЖЕТ БЫТЬ». Например, связь на рис.3.37 читается следующим образом: «Каждый КЛИЕНТ может ВЛАДЕТЬ одной или более КРЕДИТНОЙ КАРТОЙ. Каждая КРЕДИТНАЯ КАРТА должна ПРИНАДЛЕЖАТЬ ровно одному КЛИЕНТУ».

Рис. 4.28 Фрагмент ERD для банковской задачи в нотации Баркера

### 4.4.5 Словарь данных

#### 4.4.5.1 Назначение, определение и содержимое словаря данных

Диаграммы потоков данных DFD обеспечивают удобное описание функционирования компонентов системы, но не снабжают аналитика средствами описания деталей этих компонентов, а именно: какая информация преобразуется процессами и как она преобразуется. Для решения первой из перечисленных задач предназначены текстовые средства моделирования, служащие для описания структуры преобразуемой информации и получившие название словарей данных.

***Словарь данных*** – определённым образом организованный список всех элементов-данных системы с их точными определениями, что даёт возможность различным категориям пользователей (от системных аналитиков до программистов) иметь общее понимание всех входящих и выходящих потоков и компонентов хранилищ.

Для каждого потока данных в словаре необходимо хранить имя потока, его тип и атрибуты. Информация по каждому потоку состоит из ряда словарных статей, каждая из которых начинается с ключевого слова (заголовка соответствующей статьи), которому предшествует символ @.

В табл. 4.31 представлен состав некоторых понятий, связанных со словарями данных.

Таблица 4.31

Определения и состав понятий, связанных со словарями данных

|  |  |
| --- | --- |
| Компонент | Определение и/или состав |
| Виды описаний элементов данных в словаре | -   Описанием значений потоков и хранилищ, изображенных на DFD;  -   Описанием композиции агрегатов данных, движущихся вдоль потоков, то есть комплексных данных, которые могут расчленяться на элементарные символы (например, АДРЕС ПОКУПАТЕЛЯ содержит ПОЧТОВЫЙ ИНДЕКС, ГОРОД, УЛИЦА и т.д.);  -   Описанием композиции групповых данных в хранилище;  -   Специфицированием значений и областей действия элементарных фрагментов информации в потоках данных и хранилищах;  -   Описанием деталей отношений между хранилищами. |
| Типы потоков | -   Простые (элементарные) потоки или групповые (комплексные) потоки;  -   Внутренние (существуют только внутри системы) или внешние (связывающие систему с другими системами) потоки;  -   Потоки данных или потоки управления;  -   Непрерывные (принимающие любые значения в пределах определённого диапазона) или дискретные (принимающие определённые значения) потоки. |
| Атрибуты потоков | 1.     Имена-синонимы потока данных в соответствии с узлами изменения имени;  2.     БНФ-определения в случае группового потока (см. п. 4.4.5.2);  3.     Единицы измерения потока;  4.     Диапазон значений для непрерывного потока, типичные его значения и информация по обработке экстремальных значений;  5.     Список значений и их смысл для дискретного потока;  6.     Список номеров диаграмм различных типов, в которых поток встречается;  7.     Список потоков, в которых данный поток входит (как элемент БНФ-определения);  8.     Комментарии, включающие дополнительную информацию (например, цели введения данного потока) |

#### 4.4.5.2 БНФ-нотация

**БНФ-нотация** позволяет формально описать расщепление / объединение потоков (основана на форме Бэкуса-Наура, откуда следует аббревиатура БНФ). Поток может расщепляться на собственные отдельные ветви, на компоненты потока-предка или на то и другое одновременно. При расщеплении / объединении потока важно, чтобы каждый компонент потока-предка являлся именованным. Если поток расщепляется на подпотоки, необходимо чтобы все подпотоки являлись компонентами потока-предка. И наоборот, при объединении потоков каждый компонент потока-предка должен по крайней мере однажды встречаться среди подпотоков. Отметим, что при объединении подпотоков нет необходимости осуществлять исключение общих компонентов, а при расщеплении подпотоки могут иметь такие общие (одинаковые) компоненты.

Важно понимать, что точные определения потоков содержатся в словаре данных, а не на диаграммах. Например, на диаграмме может иметься групповой узел с входным потоком Х и выходным потоком Y и Z. Однако, это вовсе не означает, что соответствующее определение в словаре данных обязательно должно быть

***X = Y + Z***

Это определение может быть следующим:

***X = A + B + C***

***Y = A + B***

***Z = B + C***

Такие определения хранятся в словаре данных в так называемой ***БНФ-статье***, которая используется для описания компонентов данных в потоках данных и в хранилищах. Её синтаксис имеет вид, представленный на рис. 4.29. Там же приведены примеры описания потоков данных с помощью БНФ.

***@БНФ =***< *ПРОСТОЙ ОПЕРАТОР* >!< *БНФ-ВЫРАЖЕНИЕ* >

где ПРОСТОЙ ОПЕРАТОР – текстовое описание, заключенное в /…/

                  БНФ-ВЫРАЖЕНИЕ – выражение в форме Бэкуса-Наура, допускающее следующие *операции отношений*:

**=**  –  «Композиция из»;

**+**  –  «И»;

[!] – «ИЛИ»;

( ) – Компонент в скобках не обязателен;

{ } – Итерация компонента в скобках;

“ “  – Литерал.

Итерационные скобки могут иметь нижний и верхний предел. Например, *3{ВИНТ}7* - от трёх до семи итераций; *1{ВИНТ}* – одна и более итераций; *{ШАЙБА}3* – не более трёх итераций.

БНФ-выражение может содержать произвольные комбинации операций, например:

***@БНФ =****[ВИНТ ! БОЛТ + 2{ГАЙКА}2 + (ПРОКЛАДКА) ! КЛЕЙ]*

Приведём примеры описания потоков данных с помощью БНФ:

***@ИМЯ =****ВОСЬМИРИЧНАЯ ЦИФРА*

***@ТИП =****ДИСКРЕТНЫЙ ПОТОК*

***@БНФ****= [ “1” ! “2” ! “3” ! “4” ! “5” ! “6” ! “7” ! “0” ]*

***@ИМЯ =****ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА*

***@ТИП =****УПРАВЛЯЮЩИЙ ПОТОК*

***@БНФ****= / Указывает, что кредитная карта введена /*

***@ИМЯ =****ДАННЫЕ КРЕДИТНОЙ КАРТЫ*

***@ТИП =****ДИСКРЕТНЫЙ ПОТОК*

***@БНФ****= ПАРОЛЬ + ДЕТАЛИ КЛИЕНТА + ЛИМИТ ДЕНЕГ*

Рис. 4.29 Синтаксис БНФ-статьи и примеры описания потоков данных с помощью БНФ

Попутно отметим, что ***БНФ-нотация*** используется также при формировании миниспецификаций. Независимо от используемой нотации (псевдокод, FLOW-форма и т.п.), спецификация процесса начинается с ключевого слова, определяющего его суть (например, *@СПЕЦПРОЦ*). Требуемые входные и выходные данные должны быть специфицированы так, как показано на рис. 4.30:

***@ВХОД =****< Имя символа данных >*

***@ВЫХОД =****< Имя символа данных >*

***@ВХОДВЫХОД****= < Имя символа данных >*

где *< Имя символа данных >* - соответствующее имя из словаря данных.

В ситуациях, когда символ данных одновременно является входным и выходным, можно поступать двояко: либо символ описывается два раза с помощью ***@ВХОД*** и ***@ВЫХОД***, либо один раз с помощью ***@ВХОДВЫХОД***.

Указанные ключевые слова должны записываться перед определением спецификации процесса. Например:

***@ВХОД =****СЛОВА ПАМЯТИ*

***@ВЫХОД =****ХРАНИМЫЕ ЗНАЧЕНИЯ*

***@СПЕЦПРОЦ***

***Для всех СЛОВ ПАМЯТИ выполнить:***

***Распечатать ХРАНИМЫЕ ЗАЧЕНИЯ***

***@***

Рис. 4.30 Синтаксис и пример спецификации входных и выходных данных

Иногда в спецификации процесса задаются предусловия и постусловия выполнения данного процесса. В ***предусловии*** записываются объекты, значения которых должны быть истинны перед началом выполнения процесса (это обеспечивает определённые гарантии безопасности для пользователя). Аналогично для случая наличия ***постусловия*** гарантируется, что значения всех входящих в него объектов будут истинны при завершении процесса.

[4 ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608390)

[4.1 Методология IDEF0 (SADT) функционального моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608391)

[4.1.1 Состав функциональной модели и принципы ее построения](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608392)

[4.1.2 Разновидности диаграмм IDEF0](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608393)

[4.1.3 Стрелки и туннелирование стрелок на диаграммах IDEF0](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608394)

[4.1.4 Каркас диаграммы, слияние и расщепление моделей](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608395)

[4.2 Методология DFD моделирования потоков данных (процессов)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608396)

[4.2.1 Характеристика методологии и основные символы DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608397)

[4.2.2 Расширения реального времени в DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608398)

[4.2.3 Рекомендации по практическому применению методологии DFD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608399)

[4.3 Методологии описания процессов  IDEF3, спецификаций управления STD и имитационного моделирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608400)

[4.3.1 Сущность методологии IDEF3](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608401)

[4.3.2 Методы задания спецификаций процессов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608402)

[4.3.2.1 Структурированный естественный язык - псевдокод](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608403)

[4.3.2.2 Таблицы и деревья решений](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608404)

[4.3.2.3 Визуальные языки проектирования спецификаций](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608405)

[4.3.3 Спецификации управления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608406)

[4.3.3.1 Понятие диаграммы переходов состояний](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608407)

[4.3.3.2 Правила и способы построения STD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608408)

[4.3.3.3 Таблицы и матрицы переходов состояний](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608409)

[4.3.4 Роль и место имитационного моделирования в проектировании АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608410)

[4.4 Методологии моделирования информационного обеспечения IDEF1, IDEF1X, ERD](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608411)

[4.4.1 Основные положения, уровни и методологии моделирования данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608412)

[4.4.2 Сущности, отношения и связи в нотации Чена](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608413)

[4.4.3 Диаграммы атрибутов и категоризации сущностей в нотации Чена](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608414)

[4.4.4 Нотация Баркера](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608415)

[4.4.5 Словарь данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608416)

[4.4.5.1 Назначение, определение и содержимое словаря данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608417)

[4.4.5.2 БНФ-нотация](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk4\4%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc453608418)

[5 ОБЪЕКТНО-ОРИЕНТИРОВАНННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872406)

[5.1 UML как средство анализа и проектирования систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872407)

[5.1.1 Сущность объектно-ориентированного анализа и проектирования на основе UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872408)

[5.1.2 Характеристика канонических диаграмм UML и особенности их графического изображения](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872409)

[5.2 Синтаксис и семантика основных объектов UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872410)

[5.2.1 Диаграммы классов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872411)

[5.2.2 Диаграммы вариантов использования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872412)

[5.2.3 Диаграммы взаимодействия](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872413)

[5.2.4 Диаграммы состояний и деятельности](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872414)

[5.2.5 Диаграммы компонентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872415)

[5.3 Этапы проектирования АС с применением UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872416)

[5.3.1 Основные типы UML-диаграмм, используемые в проектировании АС, и их взаимосвязи](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872417)

[5.3.2 Разработка модели бизнес-прецедентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872418)

[5.3.3 Разработка модели бизнес-объектов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872419)

[5.3.4 Разработка концептуальной модели данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872420)

[5.3.5  Разработка требований к системе](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872421)

[5.3.6 Анализ требований и предварительное проектирование системы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872422)

[5.3.7 Разработка моделей базы данных и приложений](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872423)

[5.3.8 Проектирование физической реализации системы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872424)

**5 ОБЪЕКТНО-ОРИЕНТИРОВАНННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ**

***5.1 UML как средство анализа и проектирования систем***

**5.1.1 Сущность объектно-ориентированного анализа и проектирования на основе UML**

В п.3.4 показана актуальность и целесообразность использования объектно-ориентированного подхода при создании АС. Такой подход предполагает объектно-ориентированную методологию представления *предметной области* (в виде объектов, являющихся экземплярами соответствующих классов)и использование для разработки автоматизированных систем технологии ***объектно-ориентированного анализа и проектирования*** (ООАП, Object-Oriented Analysis/Design).

Недостаток классических нотаций структурно-функционального анализа (SADT, DFD и др.) связан с отсутствием явных средств для объектно-ориентированного представления *моделей* сложных систем, а также сложных алгоритмов обработки данных. Поскольку на рассмотренных типах диаграмм не указываются характеристики времени выполнения отдельных процессов и передачи данных между процессами, то модели систем, реализующих синхронную обработку данных, не могут быть адекватно представлены в этих нотациях. Все это ограничило возможности широкого применения соответствующих нотаций и послужило основой для разработки унифицированного языка моделирования UML.

UML (Unified Modeling Language) - унифицированный язык объектно-ориентированного моделирования явился средством достижения компромисса между двумя подходами к проектированию сложных АС:

1) жестко структурированным подходом, полно охватывающим все этапы ЖЦ системы, но не позволяющим учесть тонкие детали при разработке систем;

2) гибким подходом, обладающим возможностями тонкой настройки на специфику систем и процессов, но имеющим проблему организации взаимодействия между различными командами, реализующими проект.

Язык UML представляет собой общецелевой язык визуального моделирования, который разработан для спецификации, визуализации, проектирования и документирования бизнес-процессов, модулей программного обеспечения и других компонентов систем.

*UML* включает внутренний набор средств моделирования (ядро модулей), которые сейчас приняты во многих методах и средствах моделирования. При этом пользователям языка предоставлены возможности:

строить модели на основе средств ядра (без использования механизмов расширения для большинства типовых приложений);

добавлять при необходимости новые элементы и условные обозначения, если они не входят в ядро, или специализировать компоненты, систему условных обозначений (нотацию) и ограничения для конкретных предметных областей.

Разработка и использование моделей языка UML осуществляется в рамках общей концепции ООАП, которая, в свою очередь, является обобщением методологии объектно-ориентированного программирования (ООП, Object-Oriented Programming).

UML позволяет строить достаточно полную модель сложной системы, представляющую собой определенное число взаимосвязанных представлений (views), каждое из которых адекватно отражает конкретный аспект поведения или структуры системы. При этом наиболее общими представлениями сложной системы принято считать статическое и динамическое, которые в свою очередь могут подразделяться на другие более частные. В UML реализуется принцип иерархического построения моделей сложных систем, который предписывает рассматривать процесс построения моделей на разных уровнях (layer) абстрагирования или детализации в рамках фиксированных представлений.

В целом же процесс ООАП на базе UML можно рассматривать как последовательный переход от разработки наиболее общих моделей и представлений концептуального уровня к более частным и детальным представлениям логического и физического уровня (рис. 5.1). При этом на каждом этапе ООАП данные модели последовательно дополняются все большим количеством деталей, что позволяет им более адекватно отражать различные аспекты конкретной реализации сложной системы.

Рис. 5.1 Общая схема взаимосвязей моделей и представлений сложной системы в процессе объектно-ориентированного анализа и проектирования

ООАП предполагает, что в системе выделяются "активные сущности" – объекты (или компоненты), которые взаимодействуют друг с другом, обмениваясь сообщениями и выполняя соответствующие функции (методы) объекта. Для визуального моделирования такой системы следует использовать UML. Однако, если в основу проектирования положена функциональная декомпозиция АС, то UML не нужен и следует использовать рассмотренные ранее структурные нотации.

При выборе подхода к разработке современных АС, сложность, масштабы и функциональность которых постоянно возрастают, следует учитывать, что визуальные объектные модели все более широко используются в существующих технологиях проектирования систем. Они хорошо приспособлены для решения таких задач как: физическое перераспределение вычислений и данных, обеспечение параллелизма вычислений, репликация БД, обеспечение безопасности доступа к АС, оптимизация балансировки нагрузки АС, устойчивость к сбоям и т.п.

Визуализированные средствами UML модели АС позволяют наладить плодотворное взаимодействие между заказчиками, пользователями и командой разработчиков. Они обеспечивают ясность представления выбранных проектных решений и позволяют понять разрабатываемую систему во всей ее полноте.

**5.1.2 Характеристика канонических диаграмм UML и особенности их графического изображения**

Диаграмма (diagram) — графическое представление совокупности элементов модели в форме связного графа, вершинам и ребрам (дугам) которого приписывается определенная семантика.

Нотация **канонических диаграмм** - основное средство разработки моделей на языке UML (табл. 5.1). Перечень диаграмм и их названия являются каноническими в том смысле, что представляют собой неотъемлемую часть графической нотации языка UML. Более того, процесс ООАП неразрывно связан с процессом построения этих диаграмм.

Таблица 5.1

Виды канонических диаграмм в языке UML

|  |  |  |
| --- | --- | --- |
| № | Название английское | Название русское |
| 1 | use case diagram | вариантов использования (прецедентов) |
| 2 | class diagram | классов |
| 3 | collaboration diagram | кооперации |
| 4 | sequence diagram | последовательности |
| 5 | statechart diagram | состояний |
| 6 | activity diagram | деятельности |
| 7 | component diagram | компонентов |
| 8 | deployment diagram | развертывания |

Совокупность канонических диаграмм является самодостаточной в том смысле, что в них содержится вся информация, которая необходима для реализации проекта сложной системы. Следовательно, интегрированная модель любой сколь угодно сложной АС в нотации UML может быть представлена в виде совокупности указанных диаграмм ([рис. 5.2](http://www.intuit.ru/department/pl/umlbasics/2/3.html#image.2.7#image.2.7)).

Рис. 5.2  Интегрированная модель сложной системы в нотации UML

Каждая из этих диаграмм детализирует и конкретизирует различные представления о модели сложной системы в терминах языка UML. При этом диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. Диаграмма классов, по своей сути, логическая модель, отражающая статические аспекты структурного построения сложной системы.

Диаграммы кооперации и последовательностей представляют собой разновидности логической модели, которые отражают динамические аспекты функционирования сложной системы. Диаграммы состояний и деятельности предназначены для моделирования поведения системы. И, наконец, диаграммы компонентов и развертывания служат для представления физических компонентов сложной системы и поэтому относятся к ее физической модели.

Для построения диаграмм языка UML используются три типа визуальных графических обозначений, которые важны с точки зрения заключенной в них информации: геометрические фигуры на плоскости, графические взаимосвязи, специальные графические символы, а также строки текста (табл. 5.2). Текстовая информация в виде строк текста расширяет семантику базовых элементов.

Таблица 5.2

Типы визуальных графических обозначений в языке UML:

|  |  |  |
| --- | --- | --- |
| № | Название обозначения | Описание |
| 1 | Геометрические фигуры на плоскости | Играют роль вершин графов соответствующих *диаграмм*. При этом сами геометрические фигуры выступают в роли графических примитивов языка UML, а форма этих фигур (прямоугольник, эллипс) должна строго соответствовать изображению отдельных элементов языка UML (класс, вариант использования, состояние, деятельность). Графические примитивы языка UML имеют фиксированную семантику, переопределять которую пользователям не допускается. Графические примитивы должны иметь собственные имена, а, возможно, и другой текст, который содержится внутри границ соответствующих геометрических фигур или, как исключение, вблизи этих фигур. Информация, содержащаяся внутри фигур, имеет значение для конкретной *модели* проектируемой системы, поскольку регламентирует реализацию соответствующих элементов в программном коде. |
| 2 | Графические взаимосвязи | Представляются различными линиями на плоскости. Взаимосвязи в языке UML обобщают понятие дуг и ребер из теории графов, но имеют менее формальный характер и более развитую семантику. При этом концевые точки отрезков линий должны обязательно соприкасаться с геометрическими фигурами, служащими для обозначения вершин *диаграмм*. Путям в языке UML придается особое значение, поскольку это простые топологические сущности. |
| 3 | Специальные графические символы | Изображаются вблизи от тех или иных визуальных элементов *диаграмм* и имеют характер дополнительной спецификации («украшений»). Представляют собой графические фигуры фиксированного размера и формы. Они не могут содержать внутри себя дополнительные символы. Значки размещаются как внутри других графических конструкций, так и вне их. Примерами значков могут служить окончания связей элементов *диаграмм* или графические обозначения кванторов видимости атрибутов и операций классов. |
| 4 | Строки текста | Служат для представления различных видов информации в грамматической форме. Предполагается, что каждое использование строки текста должно соответствовать синтаксису в нотации языка UML. В отдельных случаях может быть реализован грамматический разбор этой строки, который необходим для получения дополнительной информации о *модели*. Например, строки текста в различных секциях обозначения класса могут соответствовать атрибутам этого класса или его операциям. На использование строк накладывается условие: требуется, чтобы семантика всех допустимых символов была заранее определена в языке UML или служила предметом его расширения в конкретной *модели*. |

При графическом изображении диаграмм следует придерживаться ряда основных рекомендаций (табл. 5.3).

Таблица 5.3

Рекомендации по графическому изображению диаграмм UML

|  |  |
| --- | --- |
| Рекомендация | Пояснение |
| Каждая диаграмма должна служить законченным пред-ставлением соответствую-щего фрагмента моделируе-мой предметной области. | В процессе разработки диаграммы необходимо учесть все сущности, важные с точки зрения контекста данной модели и диаграммы. Отсутствие каких-то элементов на диаграмме служит признаком неполноты модели и может потребовать ее последующей доработки. |
| Все сущности на диаграмме модели должны быть одного уровня представления. | Имеется в виду согласованность не только имен одинаковых элементов, но и возможность вложения отдельных диаграмм друг в друга для достижения полноты представлений. В случае сложных моделей систем желательно придерживаться стратегии последовательного уточнения или детализации отдельных диаграмм |
| Вся информация о сущностях должна быть явно представлена на диаграммах. | В языке UML при отсутствии некоторых символов на диаграмме могут быть использованы их значения по умолчанию (например, в случае неявного указания видимости атрибутов и операций классов), тем не менее, необходимо стремиться к явному указанию свойств всех элементов диаграмм. |
| Диаграммы не должны содержать противоречивой информации. | Противоречивость модели может служить причиной серьезных проблем при ее реализации и последующем использовании на практике. Например, наличие замкнутых путей при изображении отношений агрегирования или композиции приводит к ошибкам в программном коде, который будет реализовывать соответствующие классы. Наличие элементов с одинаковыми именами и различными атрибутами свойств в одном пространстве имен также приводит к неоднозначной интерпретации и проблемам. |
| Каждая диаграмма должна быть самодостаточной для правильной интерпретации всех ее элементов и понимания семантики всех используемых графических символов. | Любые пояснительные тексты, которые не являются собственными элементами диаграммы (например, комментариями), не должны приниматься во внимание разработчиками. В то же время общие фрагменты диаграмм могут уточняться или детализироваться на других диаграммах этого же типа, образуя вложенные или подчиненные диаграммы. Таким образом, модель системы на языке UML представляет собой пакет иерархически вложенных диаграмм, детализация которых должна быть достаточной для последующей генерации программного кода, реализующего проект соответствующей системы. |
| Количество типов диаграмм для конкретной модели приложения строго не фиксировано. | Для простых приложений нет необходимости строить все без исключения типы диаграмм. Некоторые из них могут просто отсутствовать в проекте системы, и это не будет считаться ошибкой разработчика. Например, модель системы может не содержать диаграмму развертывания для приложения, выполняемого локально на компьютере пользователя. Важно понимать, что перечень диаграмм зависит от специфики конкретного проекта системы. |
| Любая модель системы должна содержать только те элементы, которые определены в нотации языка UML. | Имеется в виду требование начинать разработку проекта, используя только те конструкции, которые уже определены в метамодели UML. Как показывает практика, этих конструкций вполне достаточно для представления большинства типовых проектов систем. И только при отсутствии необходимых базовых элементов языка UML следует использовать механизмы их расширения для адекватного представления конкретной модели. Не допускается переопределение семантики тех элементов, которые отнесены к базовой нотации метамодели языка UML. |
| Разработчики АС должны глубокого знать нотацию и семантику всех элементов языка UML | Наличие в инструментальных CASE-средствах встроенной поддержки визуализации различных диаграмм языка UML позволяет в некоторой степени исключить ошибочное использование графических символов, а также контролировать уникальность имен элементов диаграмм и т.п. Однако, поскольку вся ответственность за окончательный контроль непротиворечивости модели лежит на разработчике, необходимо помнить, что недостаточно формальный характер языка UML и возможность его расширения может служить источником потенциальных ошибок, которые в полном объеме вряд ли будут выявлены инструментальными средствами. |

***5.2 Синтаксис и семантика основных объектов UML***

**5.2.1 Диаграммы классов**

Классы представляют собой описание совокупностей однородных объектов с присущими им свойствами — атрибутами, операциями, отношениями и семантикой (табл. 5.4). В рамках модели каждому классу присваивается уникальное имя, отличающее его от других классов.

Таблица 5.4

Свойства классов UML

|  |  |
| --- | --- |
| Название свойства | Описание свойства |
| Атрибут | Свойство класса, которое может принимать множество значений. Множество допустимых значений атрибута образует домен. Атрибут имеет имя и отражает некоторое свойство моделируемой сущности, общее для всех объектов данного класса. Класс может иметь произвольное количество атрибутов. |
| Операция | Реализация функции или процедуры, которую можно запросить у любого объекта класса. Операция показывает, что можно сделать с объектом. Исполнение операции часто связано с обработкой и изменением значений атрибутов объекта и/или изменением состояния объекта. Одна и та же операция, если она полиморфна, может быть применена к разным классам. Методом называется реализация операции в конкретном классе. |
| Отношение | Свойство, характеризующее вид и степень взаимосвязи классов.  Основные варианты отношений:  ·   зависимости, которые описывают существующие между *классами* отношения использования;  ·   обобщения, связывающие обобщенные *классы* со специализированными;  ·   ассоциации, отражающие структурные отношения между объектами *классов*. |
| Семантика | Смысловое наполнение понятия конкретного класса.  Принципиальное назначение классов характеризуют стереотипы. Это, фактически, классификация объектов на высоком уровне, позволяющая определить некоторые основные свойства объекта (пример стереотипа — класс "действующее лицо"). Механизм стереотипов является также средством расширения словаря UML за счет создания на основе существующих блоков языка новых блоков, специфичных для решения конкретной проблемы. |

На рис. 5.3 приведено графическое изображение класса "Заказ" в нотации UML и типичныйсинтаксис UML для свойств классов(в отдельных программных средствах порядок записи параметров может быть иным). Упорядоченный набор значений характеристик свойств класса называется сигнатурой.

<признак видимости> <имя атрибута> : <тип данных> [= <значение по умолчанию>]

<признак видимости> <имя операции> <([список аргументов])>

Рис. 5.3  Изображение класса в UML и типовой синтаксис для свойств

Видимость свойства указывает на возможность его использования другими классами. Один класс может "видеть" другой, если тот находится в области действия первого и между ними существует явное или неявное отношение. В языке UML определены три основных уровня видимости (общий, защищенный и закрытый) и один дополнительный (пакетный), описанные в табл. 5.5.

Область действия свойства указывает, будет ли оно проявлять себя по-разному в каждом экземпляре класса, или одно и то же значение свойства будет совместно использоваться всеми экземплярами (табл. 5.5).

Возможное количество экземпляров класса называется его кратностью. В UML можно определять четыре разновидности классов с точки зрения ихкратности (табл. 5.5).

Таблица 5.5

Характеристики свойств классов в UML

|  |  |
| --- | --- |
| Характеристика | Описание |
| Уровень видимости | ·       public (общий) — любой внешний *класс*, который "видит" данный, может пользоваться его общими свойствами. Обозначаются символом " + " перед именем атрибута или операции;  ·       protected (защищенный) — только любой потомок данного *класса* может пользоваться его защищенными свойствами. Символ " # ";  ·      private (закрытый) — только данный *класс* может пользоваться этими свойствами. Символ " - ";  ·      package (пакетный) - а*трибут* с этой областью видимости недоступен или невиден для всех *классов* за пределами пакета, в котором определен *класс*-владелец данного *атрибута.* Символ "~". |
| Область действия | ·        instance (экземпляр) — у каждого экземпляра *класса* есть собственное значение данного свойства;  ·        classifier (классификатор) — все экземпляры совместно используют общее значение данного свойства (выделяется на диаграммах подчеркиванием). |
| Кратность класса | ·         не содержащие ни одного экземпляра — тогда *класс* становится служебным (Abstract);  ·         содержащие ровно один экземпляр (Singleton); символ "1" – обязательное единственное значение, символ " 0..1" - необязательное единственное значение;  ·         содержащие заданное число n экземпляров; символы " n" или " 0..n" в зависимости от степени обязательности наличия ровно n экземпляров;  ·         содержащие произвольное число экземпляров; символ "\*" |

Диаграммы классовпозволяют описать систему в статическом состоянии — определить типы объектов системы (собственно классы) и различного рода статические связи между ними. Между классами возможны различные отношения, перечисленные в табл. 5.4 (зависимости, обобщения и ассоциации) и представленные графически на рис. 5.4. Укрупненное описание отношений приведено в табл. 5.6.

Рис. 5.4  Отображение связей между классами

Таблица 5.6

Укрупненное описание отношений в UML

|  |  |
| --- | --- |
| Отношение | Описание |
| Зависимость | Отношение использования, согласно которому изменение в спецификации одного элемента (например, класса "товар") может повлиять на использующий его элемент (класс "строка заказа"). Часто зависимости показывают, что один класс использует другой в качестве аргумента. На диаграмме представляется обычной стрелкой. |
| Обобщение | Отношение между общей сущностью (родителем — класс "клиент") и ее конкретным воплощением (потомком — *классы* "корпоративный клиент" или "частный клиент"). На диаграмме представляется закрашенной стрелкой. Объекты класса -потомка могут использоваться всюду, где встречаются объекты класса -родителя, но не наоборот. При этом он наследует свойства родителя (его атрибуты и операции). Операция потомка с той же сигнатурой, что и у родителя, замещает операцию родителя; это свойство называют полиморфизмом. Класс, у которого нет родителей, но есть потомки, называется корневым. Класс, у которого нет потомков, называется листовым. |
| Ассоциация | Отношение, показывающее, что объекты одного типа неким образом связаны с объектами другого типа (" клиент " может сделать " заказ "). На диаграмме представляется линией. Если между двумя классами определена ассоциация, то можно перемещаться от объектов одного класса к объектам другого. При необходимости направление навигации может задаваться стрелкой. Допускается задание ассоциаций на одном классе. В этом случае оба конца ассоциации относятся к одному и тому же классу. Это означает, что с объектом некоторого *класса* можно связать другие объекты из того же класса. |

Ассоциации может быть присвоено имя, описывающее семантику отношений. Каждая ассоциация имеет две роли, которые могут быть отражены на диаграмме (рис. 5.5). Роль ассоциации обладает свойством множественности, которое показывает, сколько соответствующих объектов может участвовать в данной связи.

Иллюстрируется модель формирования заказа. Каждый заказ может быть создан единственным клиентом (множественность роли 1...1). Каждый клиент может создать один и более заказов (множественность роли 1..n). Направление навигации показывает, что каждый заказ должен быть " привязан " к определенному клиенту.

Рис. 5.5  Свойства ассоциации

Такого рода ассоциация является простой и отражает отношение между равноправными сущностями, когда оба класса находятся на одном концептуальном уровне и ни один не является более важным, чем другой. Если приходится моделировать отношение типа "часть-целое", то используется специальный тип ассоциации — агрегирование. В такой ассоциации один из классов имеет более высокий ранг (целое — класс"заказ", рис. 5.4) и состоит из нескольких меньших по рангу классов (частей — класс "строка заказа"). На диаграмме представляется стрелкой с ромбовидным концом.

**5.2.2 Диаграммы вариантов использования**

Диаграммы вариантов использования, называемые еще диаграммами прецедентов, описывают функциональность АС, которая будет видна пользователям системы. Прецедент — это типичное взаимодействие пользователя с системой, которое при этом:

-      описывает видимую пользователем функцию,

-      может представлять различные уровни детализации,

-      обеспечивает достижение конкретной цели, важной для пользователя.

Прецедент обозначается на диаграмме овалом, связанным с пользователями, которых принято называть действующими лицами, исполнителями или актерами (actors) и обозначать символом «человечек». Актер использует систему (или используются системой) в данном прецеденте и выполняет в нем некоторую роль. На диаграмме изображается только один актер, однако реальных пользователей, выступающих в данной роли по отношению к АС, может быть много. Список всех прецедентов фактически определяет функциональные требования к АС, которые лежат в основе разработки технического задания на ее создание.

На диаграммах прецедентов, кроме связей между актерами и прецедентами, возможно использование еще двух видов связей между прецедентами: "использование" и "расширение" (рис. 5.6).

Показано: 1) при исполнении прецедента "Сформировать заказ" возможно использование информации из предыдущего заказа, что позволит не вводить все необходимые данные;

2) при исполнении прецедентов "Оценить риск сделки" и "Согласовать цену" необходимо выполнить одно и то же действие — рассчитать стоимость заказа.

Рис. 5.6  Связи на диаграммах прецедентов

Связь типа "расширение" применяется, когда один прецедент подобен другому, но несет несколько большую функциональную нагрузку. Ее следует применять при описании изменений в нормальном поведении системы. Связь типа "использование" позволяет выделить некий фрагмент поведения системы и включать его в различные прецеденты без повторного описания.

**5.2.3 Диаграммы взаимодействия**

В UMLописание поведения системы задается диаграммами взаимодействия, определяющими потоки сообщений между объектами.  Как правило, диаграмма взаимодействия охватывает поведение объектов в рамках одного варианта использования. Существуют два вида ***диаграмм взаимодействия*:** диаграммы последовательностей и кооперативные диаграммы.

Диаграмма последовательностей используется для точного определения логики (сценария) выполнения прецедента. Эти диаграммы отображают типы объектов, взаимодействующих при исполнении прецедентов, сообщения, которые они посылают друг другу, и любые возвращаемые значения, ассоциированные с этими сообщениями.

Прямоугольники на диаграмме представляют различные объекты и роли, которые они имеют в системе, а горизонтальные линии отображают отношения (или ассоциации) между объектами. Вертикальные линии показывают "время жизни" объекта. Сообщения обозначаются ярлыками возле стрелок, они могут иметь нумерацию и показывать возвращаемые значения. Кроме этого линии со стрелками и надписями названий методов означают вызов метода у объекта (рис. 5.7).

·        вводятся строки заказа;

* по каждой строке проверяется наличие товара;
* если запас достаточен — инициируется поставка;
* если запас недостаточен — инициируется дозаказ (повторный заказ).

Рис. 5.7  Диаграмма последовательности обработки заказа

Сообщения появляются в той последовательности, как они показаны на диаграмме — сверху вниз. Если предусматривается отправка сообщения объектом самому себе (самоделегирование), то стрелка начинается и заканчивается на одной линии жизни".

На диаграммы может быть добавлена управляющая информация: описание условий, при которых посылается сообщение; признак многократной отправки сообщения (маркер итерации); признак возврата сообщения.

На кооперативных диаграммах объекты показываются в виде прямоугольников, а стрелками обозначаются сообщения, которыми они обмениваются в рамках одного прецедента (варианта использования). Временная последовательность сообщений отражается их нумерацией (рис.5.8).

Рис. 5.8   Кооперативная диаграмма прохождения заказа

**5.2.4 Диаграммы состояний и деятельности**

Диаграммы состояний используются для описания поведения сложных систем и определяют все возможные состояния, в которых может находиться объект, а также процесс смены состояний объекта в результате некоторых событий. Эти диаграммы обычно используются для описания поведения одного объекта в нескольких прецедентах (рис.5.9).

Переходы имеют метки, которые синтаксически состоят из трех необязательных частей:

<Событие> <[Условие]> < / Действие>.

На диаграммах также отображаются функции, которые выполняются объектом в определенном состоянии. Синтаксис метки деятельности:

Выполнить / < деятельность >.

Рис. 5.9  Диаграмма состояний объекта «заказ»

Прямоугольниками представляются состояния, через которые проходит объект во время своего поведения. Состояниям соответствуют определенные значения атрибутов объектов. Стрелки представляют переходы от одного состояния к другому, которые вызываются выполнением некоторых функций объекта. Имеется также два вида псевдо-состояний: начальное состояние, в котором находится только что созданный объект, и конечное состояние, которое объект не покидает, как только туда перешел.

Диаграмма деятельности — это частный случай диаграммы состояний*.* На диаграмме деятельности представлены переходы потока управления от одной деятельности к другой внутри системы. Этот вид диаграмм обычно используется для описания поведения, включающего в себя множество параллельных процессов.

Основные элементы диаграмм деятельности и пример такой диаграммы показаны на рис 5.10.

Основные элементы диаграмм деятельности:

·        овалы, изображающие действия объекта;

* линейки синхронизации, указывающие на необходимость завершить или начать несколько действий (модель логического условия "И");
* ромбы, отражающие принятие решений по выбору одного из маршрутов выполнения процесса (модель логического условия "ИЛИ");
* стрелки — отражают последовательность действий, могут иметь метки условий.

Рис. 5.10   Диаграмма деятельности — обработка заказа

**5.2.5 Диаграммы компонентов**

***Диаграммы компонентов*** позволяют изобразить модель системы на физическом уровне. Элементами диаграммы являются компоненты —замещаемые модули реальной системы. Каждый компонент является полностью независимым элементом системы. Разновидностью компонентов являются узлы. Узлы системы делятся на два типа: устройства (в них данные не обрабатываются) и процессоры (осуществляют обработку данных).

Компонентом может быть любой достаточно крупный модульный объект, такой как таблица или экстент базы данных, подсистема, бинарный исполняемый файл, готовая к использованию система или приложение. Таким образом, диаграмму компонентов можно рассматривать как диаграмму классов в более крупном (менее детальном) масштабе. Компонент, как правило, представляет собой физическую упаковку логических элементов, таких как классы, интерфейсы и кооперации.

Основное назначение диаграмм компонентов — разделение системы на элементы, которые имеют стабильный интерфейс и образуют единое целое. Это позволяет создать ядро системы, которое не будет меняться в ответ на изменения, происходящие на уровне подсистем.

На рис. 5.11 приведен пример представления диаграммы компонентов..

Рис. 5.11  Диаграмма компонентов фрагмента АС

Каждый компонент диаграммы при необходимости документируется с помощью более детальной диаграммы компонентов, диаграммы сценариев или диаграммы классов.

***5.3 Этапы проектирования АС с применением UML***

**5.3.1 Основные типы UML-диаграмм, используемые в проектировании АС, и их взаимосвязи**

UML обеспечивает поддержку всех этапов жизненного цикла АС. При этом используются как канонические диаграммы UML, так и ряд производных от них диаграмм, специализированных именно для целей проектирования АС (в табл. 5.7 перечислены наиболее часто используемые диаграммы).

Таблица 5.7

UML-диаграммы, используемые в проектировании АС

|  |  |  |
| --- | --- | --- |
| № | Название диаграммы | Характеристика |
| 1 | Диаграммы прецедентов (диаграммы вариантов использования) | Обобщенная модель функционирования системы в окружающей среде. |
| 2 | Диаграммы видов деятельности (диаграммы деятельностей) | Модель бизнес-процесса или поведения системы в рамках прецедента. |
| 3 | Диаграммы взаимодействия (interaction diagrams) | Модель процесса обмена сообщениями между объектами, представляется в виде диаграмм последовательностей (sequence diagrams) или кооперативных диаграмм (collaboration diagrams). |
| 4 | Диаграммы состояний | Модель динамического поведения системы и ее компонентов при переходе из одного состояния в другое. |
| 5 | Диаграммы классов | Логическая модель базовой структуры системы, отражает статическую структуру системы и связи между ее элементами. |
| 6 | Диаграммы базы данных | Модель структуры базы данных, отображает таблицы, столбцы, ограничения и т.п. |
| 7 | Диаграммы компонентов | Модель иерархии подсистем, отражает физическое размещение баз данных, приложений и интерфейсов АС. |
| 8 | Диаграммы развертывания (диаграммы размещения) | Модель физической архитектуры системы, отображает аппаратную конфигурацию АС. |

На этапе создания концептуальной модели для описания бизнес-деятельности используются модели бизнес-прецедентов и диаграммы видов деятельности, для описания бизнес-объектов – модели бизнес-объектов и диаграммы последовательностей.

На этапе создания логической модели АС описание требований к системе задается в виде модели и описания системных прецедентов, а предварительное проектирование осуществляется с использованием диаграмм классов, диаграмм последовательностей и диаграмм состояний.

На этапе создания физической модели детальное проектирование выполняется с использованием диаграмм классов, диаграмм компонентов, диаграмм развертывания.

На рис. 5.12 показаны отношения между различными видами диаграмм UML. Приведенная схема является наглядной иллюстрацией итеративного характера разработки моделей с использованием UML.

Указатели стрелок следует интерпретировать как отношение "является источником входных данных для..." (например, диаграмма прецедентов является источником данных для диаграмм видов деятельности, классов и компонентов).

Рис. 5.12   Взаимосвязи между диаграммами UML

**5.3.2 Разработка модели бизнес-прецедентов**

**Модель бизнес-прецедентов** описывает бизнес-процессы с точки зрения внешнего пользователя, т.е. отражает взгляд на деятельность организации извне. Проектирование системы начинается с изучения и моделирования бизнес-деятельности организации. На этом этапе вводится и отображается в модели ряд понятий, свойственных объектно-ориентированному подходу: исполнитель, прецедент, класс, отношение и т.д. (см. пп.5.1-5.2). Для включения в диаграмму выбранные прецеденты должны удовлетворять следующим критериям:

прецедент должен описывать, **ЧТО** нужно делать, а не **КАК**;

прецедент должен описывать действия с точки зрения **ИСПОЛНИТЕЛЯ**;

прецедент должен возвращать исполнителю некоторое **СООБЩЕНИЕ**;

последовательность действий внутри прецедента должна представлять собой одну **НЕДЕЛИМУЮ** цепочку.

Для иллюстрации этапов разработки проекта используем адаптированные материалы проекта АС медицинского центра. Назначение АС – автоматизация ведения и использования клинических записей о пациентах. В настоящее время эта работа производится вручную персоналом центра. На рис. 5.13 представлена общая модель деятельности центра в виде диаграммы прецедентов. Прецедент "Обслуживание пациента" реализуется через множество других, более ограниченных прецедентов (рис. 5.14), отражающих детализацию представления функционирования центра.

Рис. 5.13  Общая диаграмма деятельности медицинского центра по обслуживанию пациента

Рис. 5.14  Модель бизнес-прецедентов, составляющих обслуживание пациента

Исходя из цели создания системы, для дальнейшего исследования и моделирования отбираются только те бизнес-прецеденты, которые связаны с использованием клинических записей.

Выполнение прецедента описывается с помощью диаграмм видов деятельности, которые отображают исполнителей и последовательность выполнения соответствующих бизнес-процессов (рис. 5.15).

Рис. 5.15  Диаграмма видов деятельности для прецедента "Оказание медицинской помощи"

Общее поле диаграммы деятельности делится на несколько "плавательных дорожек", каждая из которых содержит описание действий одного из исполнителей. Основными элементами диаграмм видов деятельности являются обозначения состояния ("начало", "конец "), действия (овал) и момента синхронизации действий (линейка синхронизации, на которой сходятся или разветвляются несколько стрелок).

Диаграмма подходит для описания действий как внешнего, так и внутреннего специалиста центра.

Этап завершается после разработки диаграмм видов деятельности для всех выделенных в модели бизнес-прецедентов. Разработанные на данном этапе модели могут еще неоднократно корректироваться.

**5.3.3 Разработка модели бизнес-объектов**

Модели бизнес-объектов показывают выполнение бизнес-процессов организации ее внутренними исполнителями. Основными компонентами этих моделей являются внешние и внутренние исполнители, а также бизнес-сущности, отображающие все, что используют внутренние исполнители для реализации бизнес-процессов. Пример модели бизнес-объектов для прецедента "Ответ на запрос" приведен на рис. 5.16.

Рис. 5.16  Модель бизнес-объектов прецедента "Ответ на запрос"

Здесь появилось новое действующее лицо – отправитель запроса. Понятие "Отправитель запроса" служит для обобщенного представления разных действующих лиц при описании прецедента "Ответ на запрос" (рис. 5.17). "Отправитель запроса" становится суперклассом по отношению к обобщаемым понятиям (подклассам).

Рис. 5.17  Обобщение классов

Для детального описания выполнения бизнес-процессов обычно используются диаграммы последовательностей (рис. 5.18). Результатом этого этапа являются согласованные с заказчиком и достаточно подробные описания действий специалистов организации, внедряющей АС, необходимые для обеспечения исполнения ее функций.

Рис. 5.18  Диаграмма последовательностей для прецедента "Ответ на запрос"

**5.3.4 Разработка концептуальной модели данных**

На основе информации, выявленной на этапах бизнес-моделирования, выполняется разработка ***концептуальной модели данных***, которые будут использоваться в разрабатываемой системе. На рис. 5.19 представлена в виде диаграммы классов модель данных для объекта "Клинические записи". Модель показывает, что клинические записи включают (агрегируют) ряд блоков.

Этот этап завершает процедуры бизнес-моделирования и позволяет представить команде проектировщиков в едином формате ту информацию, которая будет необходима для создания системы. Разработанные диаграммы являются отправной точкой в процессах проектирования баз данных и приложений системы. Эти диаграммы, конечно же, могут еще меняться, однако они обеспечивают согласованность действий бизнес-аналитиков и разработчиков в процессе дальнейшей работы над системой.

Рис. 5.19  Концептуальная модель данных

**5.3.5  Разработка требований к системе**

Основой разработки требований является ***модель системных прецедентов***, отражающая выполнение конкретных обязанностей внутренними и внешними исполнителями с использованием АС.

Источником данных для создания модели системных прецедентов являются разработанные на предыдущем этапе бизнес-модели. Однако при создании модели полезно предварительно составить детальные описания прецедентов, содержащие определения используемых данных и точную последовательность их выполнения. Описание осуществляется в соответствии с принятым в организации шаблоном, который обычно включает разделы, указанные в табл. 5.8.

Таблица 5.8

Разделы шаблона описания прецедентов

|  |  |
| --- | --- |
| № | Содержимое раздела |
| 1 | Заголовок (название прецедента, ответственный за исполнение, дата создания шаблона / внесения изменений) |
| 2 | Краткое описание прецедента |
| 3 | Ограничения |
| 4 | Предусловия (необходимое состояние системы или условия, при которых должен выполняться прецедент) |
| 5 | Постусловия (возможные состояния системы после выполнения прецедента) |
| 6 | Предположения |
| 7 | Основная последовательность действий |
| 8 | Альтернативные последовательности действий и условия, их инициирующие |
| 9 | Точки расширения и включения прецедентов |

В процессе создания модели системных прецедентов осуществляется преобразование и перенос компонентов бизнес-моделей на новые диаграммы. Типовые преобразования приведены в табл. 5.9.

Таблица 5.9

Типовые преобразования бизнес-моделей в процессе создания модели системных прецедентов

|  |  |
| --- | --- |
| **Элементы бизнес-модели** | **Элементы *модели системных прецедентов*** |
| Бизнес-прецеденты | Подсистемы |
| Внешние исполнители | Исполнители |
| Внутренние исполнители | Исполнители или прецеденты |
| Процессы, выполняемые внутренними исполнителями | Прецеденты |

На рис. 5.20 представлена модель системных прецедентов для бизнес-прецедента "Оказание медицинской помощи".

Рис. 5.20  Модель системных прецедентов

 Еще раз напомним, что исходя из цели создания системы, в модели системных прецедентов отражены только те действия исполнителей, которые связаны с предоставлением доступа и обновлением клинических записей.

Описываемые моделью функции характерны только для одного вида деятельности – оказания медицинской помощи, и в основном не используются в других видах деятельности Центра. Это позволяет объединить выделенные функции в некую единую подсистему проектируемой АС.

Внутренний исполнитель "Персонал центра" (см. рис. 5.15, рис. 5.18) и выполняемый им ручной процесс преобразован в системный прецедент "Предоставление доступа к клиническим записям".

Внешние исполнители (например, "Производитель медицинского оборудования") непосредственно взаимодействуют с проектируемой системой, т.е. превращаются в исполнителей.

Прецедент "Проверка прав доступа" впервые появился на диаграммах и реализуется средствами разрабатываемой АС. Поэтому для него приходится разрабатывать диаграмму последовательностей, описывающую его исполнение (рис. 5.21). В результате в проектируемой АС появляются два новых объекта – программный модуль "Менеджер защиты" и информационный блок "Набор прав".

Рис. 5.21  Диаграмма последовательностей для прецедента "Проверка прав"

Таким образом, результатом разработки модели системных прецедентовявляется не только исчерпывающий перечень функций, которые должны быть реализованы в проектируемой системе, но и подробное описание необходимой реализации этих функций.

**5.3.6 Анализ требований и предварительное проектирование системы**

Основные задачи этапа:

определить проект системы, который будет отвечать всем бизнес-требованиям;

разработать общий предварительный проект для всех команд разработчиков (проектировщиков баз данных, разработчиков приложений, системных архитекторов и пр.)

Основным инструментом на данном этапе являются диаграммы классов системы, которые строятся на основе разработанной ***модели системных прецедентов***. Одновременно на этом этапе уточняются диаграммы последовательностей выполнения отдельных прецедентов, что приводит к изменениям в составе объектов и диаграммах классов.

Диаграммы классов системы заполняются объектами из модели системных прецедентов*.* Они содержат описание этих объектов в виде классов и описание взаимодействия между классами. Диаграмма классов, описывающая процедуры защиты доступа к данным, приведена на рис. 5.22.

Рис. 5.22  Диаграмма классов "Защита доступа"

Таким образом, в результате этого этапа проектирования появляется достаточно подробное описание состава и функций проектируемой системы, а также информации, которую необходимо использовать в базе данных и в приложениях.

**5.3.7 Разработка моделей базы данных и приложений**

На этом этапе осуществляется отображение элементов полученных ранее моделей классов в элементы моделей базы данных и приложений согласно порядку, показанному в табл. 5.10.

Таблица 5.10

Порядок отображения элементов моделей классов в элементы моделей базы данных и приложений

|  |  |
| --- | --- |
| Исходный элемент | Элемент отображения |
| классы | таблицы |
| атрибуты | столбцы |
| типы | типы данных используемой СУБД |
| ассоциации | связи между таблицами (ассоциации "многие-ко-многим" преобразуются в ассоциации "один-ко-многим" посредством создания дополнительных таблиц связей) |
| приложения | отдельные классы с окончательно определенными и связанными с данными в базе методами и атрибутами |

Поскольку модели базы данных и приложений строятся на основе единой логической модели, автоматически обеспечивается связность этих проектов (рис. 5.23).

Рис. 5.23  Связь между проектами базы данных и приложений

В модель базы данных отображаются только перманентные классы, из которых удаляются атрибуты, не отображаемые в столбцах. Для каждого простого класса в модели базы данных формируется отдельная таблица, включающая столбцы, соответствующие атрибутам класса.

Отображение классов подтипов в таблицы осуществляется одним из стандартных способов: одна таблица на класс; одна таблица на суперкласс; одна таблица на иерархию.

В последнем случае создается единая таблица, содержащая атрибуты как суперкласса, так и всех подклассов (рис. 5.24). При этом для выделения исходных таблиц подклассов в результирующую таблицу добавляется один или более дополнительных столбцов (на рисунке показан курсивом).

Рис. 5.24  Преобразование иерархии в таблицу

Разработка проекта базы данных осуществляется с использованием специального UML-профиля (Profile for Database Design), который включает основные компоненты диаграмм, описанные в табл. 5.11.

Таблица 5.11

Основные компоненты диаграмм в соответствии с UML-профилем

|  |  |
| --- | --- |
| Компонент | Описание |
| таблица | набор записей базы данных по определенному объекту |
| столбец | элемент таблицы, содержащий значения одного из атрибутов таблицы |
| первичный ключ (РК) | атрибут, однозначно идентифицирующий строку таблицы |
| внешний ключ (FK) | один или группа атрибутов одной таблицы, которые могут использоваться как первичный ключ другой таблицы |
| обязательная связь | связь между двумя таблицами, при которой дочерняя таблица существует только вместе с родительской |
| необязательная связь | связь между таблицами, при которой каждая из таблиц может существовать независимо от другой |
| представление | виртуальная таблица, которая обладает всеми свойствами обычной таблицы, но не хранится постоянно в базе данных |
| хранимая процедура | функция обработки данных, выполняемая на сервере |
| домен | множество допустимых значений для столбца таблицы |

На рис. 5.25 представлен фрагмент модели базы данных — две таблицы, соответствующие классам "пациент" (рис. 5.14, рис. 5.17) и "минимальный набор данных" (рис. 5.19). Связь между ними обязательная, поскольку "минимальный набор данных" не может существовать без "пациента".

Рис. 5.25  Фрагмент модели базы данных

На диаграммах указываются дополнительные характеристики таблиц и столбцов, например, показанные в табл. 5.12.

Таблица 5.12

Основные компоненты диаграмм в соответствии с UML-профилем

|  |  |
| --- | --- |
| Компонент | Описание |
| Ограничения | Определяют допустимые значения данных в столбце или операции над данными (ключ (PK,FK) – ограничение, определяющее тип ключа и его столбец; проверка (Check) – ограничение, определяющее правило контроля данных; уникальность (Unique) – ограничение, определяющее, что в столбце содержатся неповторяющиеся данные) |
| Триггер | Программа, выполняющая при определенных условиях предписанные действия с базой данных |
| Тип данных | Структурно-содержательная характеристика информации, размещаемой  в столбце таблицы |

Результатом этапа является детальное описание проекта базы данных и приложений системы.

**5.3.8 Проектирование физической реализации системы**

На этом этапе проектирования модели баз данных и приложений дополняются обозначениями их размещения на технических средствах разрабатываемой системы. На рис. 5.26 приведено изображение разделения таблицы " пациент " на три экстента ( <<Tablespace>> ) в соответствии с первой буквой фамилии пациента.

Рис. 5.26  Экстенты таблицы "Пациент"

Основные понятия UML, которые используются на данном этапе, приведены в табл. 5.13.

Таблица 5.13

Основные понятия UML для этапа физической реализации системы

|  |  |
| --- | --- |
| Понятие | Описание |
| Компонент | самостоятельный физический модуль системы |
| Зависимость | связь между двумя элементами, при которой изменения в одном элементе вызывают изменения другого элемента |
| Устройство | узел, не обрабатывающий данные |
| Процессор | узел, выполняющий обработку данных |
| Соединение | связь между устройствами и процессорами |

Диаграммы развертывания позволяют отобразить на единой схеме различные компоненты системы (программные и информационные) и их распределение по комплексу технических средств (рис. 5.27).

Рис. 5.27  Фрагмент диаграммы развертывания АС

[5 ОБЪЕКТНО-ОРИЕНТИРОВАНННЫЕ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872359)

[5.1 UML как средство анализа и проектирования систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872360)

[5.1.1 Сущность объектно-ориентированного анализа и проектирования на основе UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872361)

[5.1.2 Характеристика канонических диаграмм UML и особенности их графического изображения](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872362)

[5.2 Синтаксис и семантика основных объектов UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872363)

[5.2.1 Диаграммы классов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872364)

[5.2.2 Диаграммы вариантов использования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872365)

[5.2.3 Диаграммы взаимодействия](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872366)

[5.2.4 Диаграммы состояний и деятельности](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872367)

[5.2.5 Диаграммы компонентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872368)

[5.3 Этапы проектирования АС с применением UML](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872369)

[5.3.1 Основные типы UML-диаграмм, используемые в проектировании АС, и их взаимосвязи](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872370)

[5.3.2 Разработка модели бизнес-прецедентов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872371)

[5.3.3 Разработка модели бизнес-объектов](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872372)

[5.3.4 Разработка концептуальной модели данных](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872373)

[5.3.5  Разработка требований к системе](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872374)

[5.3.6 Анализ требований и предварительное проектирование системы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872375)

[5.3.7 Разработка моделей базы данных и приложений](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872376)

[5.3.8 Проектирование физической реализации системы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk5\5%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456872377)

[6 CASE-СРЕДСТВА ПРОЕКТИРОВАНИЯ АС (CASE)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541701)

[*6.1 Общие положения по CASE*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541702)

[6.1.1 Эволюция CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541703)

[6.1.2 Состав, структура и функциональные особенности CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541704)

[6.1.3 Поддержка графических моделей в CASE-средствах](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541705)

[6.1.4 Контроль ошибок в CASE-средствах](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541706)

[6.1.5 Организация и поддержка репозитория](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541707)

[6.1.6 Поддержка процесса проектирования и разработки в CASE](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541708)

[*6.2 Характеристика отдельных CASE-средств и их комплексов*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541709)

[6.2.1 Обзор CASE-средств Silverrun+JAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541710)

[6.2.1.1 Silverrun](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541711)

[6.2.1.2 JAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541712)

[6.2.2 Обзор CASE-средств Designer/2000 + Developer/2000](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541713)

[6.2.3 Обзор CASE-средств ERwin + BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541714)

[6.2.3.1 ERwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541715)

[6.2.3.2 BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541716)

[6.2.3.3 Функционально-стоимостный анализ в BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541717)

[6.2.3.4 Свойства, определяемые пользователем (UDP)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541718)

[6.2.4 Обзор CASE-средства Rational Rose](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541719)

[6.2.5 Обзор CASE-средства Sparx Enterprise Architect (SEA)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541720)

[6.2.6 Обзор вспомогательных средств поддержки жизненного цикла автоматизированных систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541721)

[6.2.6.1 Средства конфигурационного управления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541722)

[6.2.6.2 Средства документирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541723)

[6.2.6.3 Средства тестирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541724)

[6.2.6.4 Проблема рационального выбора CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541725)

**6 CASE-СРЕДСТВА ПРОЕКТИРОВАНИЯ АС (CASE)**

**6.1 Общие положения по CASE**

**6.1.1 Эволюция CASE-средств**

С самого начала CASE-технологии развивались с целью преодоления ограничений ручных применений методологий анализа и проектирования АС (сложности понимания, большой трудоемкости и стоимости, трудности внесения изменений в проект и т.д.) за счёт их автоматизации и интеграции средств. Современные CASE-технологии являются естественным продолжением эволюции всей отрасли разработки ПО. Традиционно выделяют шесть периодов такой эволюции, качественно отличающихся применяемой техникой и методами разработки ПО (табл. 6.1).

Таблица 6.1

Периоды эволюции отрасли разработки ПО

|  |  |
| --- | --- |
| № периода | Характеристика (применяемая техника и методы разработки ПО) |
| 1 | Ассемблеры, дампы памяти, анализаторы |
| 2 | Компиляторы, интерпретаторы, трассировщики |
| 3 | Символические отладчики, пакеты программ |
| 4 | Системы анализа и управления исходными текстами |
| 5 | CASE-средства анализа требований, проектирования спецификаций и структуры редактирования интерфейсов (I генерация CASE – CASE-I) |
| 6 | CASE-средства генерации исходных текстов и реализации интегрированного окружения поддержки полного ЖЦ разработки ПО (II генерация – CASE-II) |

Краткая характеристика генераций собственно CASE-средств дана в табл. 6.2.

Таблица 6.2

Краткая характеристика генераций CASE-средств

|  |  |
| --- | --- |
| Генерация | Характеристика |
| CASE-I | Технология, адресованная непосредственно системным аналитикам и проектировщикам. Включает средства для поддержки графических моделей, проектирования спецификаций, экранных редакторов и словарей данных. Не поддерживает полный ЖЦ АС и концентрирует внимание на функциональных спецификациях и начальных шагах проекта – системном статусе, определении требований, системном проектировании, логическом проектировании БД. |
| CASE-II | Отличается более развитыми возможностями, улучшенными характеристиками и исчерпывающим подходом к ЖЦ. На первый план выдвигаются средства поддержки автоматической кодогенерации, а также обеспечивается полная функциональная поддержка для порождения графических системных требований и спецификаций проекта; контроля, аннализа и связывания системной информации и информации по управле-нию проектированием; построения прототипов и моделей системы; тестиро-вания, верификации и анализа сгенерированных программ; генерации доку-ментов по проекту; контроля на соответствие стандартам по всем этапам ЖЦ. Включает свыше ста функциональных компонентов с возможностью выбора необходимых средств и их интеграции в нужном составе. |

**6.1.2 Состав, структура и функциональные особенности CASE-средств**

Обычно к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла АС и обладающее следующими основными характерными особенностями:

мощные графические средства для описания и документирования АС, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности;

интеграция отдельных компонентов CASE-средств, обеспечивающая управляемость процессом разработки АС;

использование специальным образом организованного хранилища проектных метаданных (репозитория).

Помимо этого в основе концептуального построения CASE-средств лежат следующие положения, представленные в табл. 6.3.

Таблица 6.3

Положения, лежащие в основе концептуального построения CASE-средств

|  |  |
| --- | --- |
| № | Характеристика положения |
| 1 | Учет человеческого фактора, определяющий разработку АС как лёгкий, удобный и экономичный процесс |
| 2 | Широкое использование базовых программных средств, получивших массовое распространение в различных приложениях (БД и СУБД, компиляторы, отладчики, документаторы, базы знаний, оболочки экспертных систем и т.п.) |
| 3 | Автоматизированная или автоматическая кодогенерация, выполняющая несколько видов генерации кодов: преобразования для получения документации, формирования БД, ввода или модификации данных, получения выполняемых машинных кодов из спецификации ПО, автоматической сборки модулей из словарей и моделей данных и повторно используемых программ, автоматической конверсии ранее используемых файлов в форматы новых требований |
| 4 | Ограничение сложности, позволяющее получать компоненты, поддающиеся управлению, обозримые и доступные для понимания, а также обладающие простой и ясной структурой |
| 5 | Доступность для разных категорий пользователей |
| 6 | Рентабельность (окупаемость затрат в заданные сроки) |
|  | Сопровождаемость, обеспечивающая способность адаптации при изменении требований и целей проекта |

Интегрированные CASE-средства содержат четыре основных компонента: «Репозиторий», «Средства ввода», «Средства анализа, проектирования и разработки» и «Средства вывода» (табл. 6.4).

Таблица 6.4

Основные компоненты интегрированных CASE-средств

|  |  |
| --- | --- |
| Компонент | Характеристика |
| Репозиторий | Средство централизованного хранения всей информации о проектируемом ПО в течение всего жизненного цикла – основа CASE-пакета. Соответствующая БД должна иметь возможность поддерживать большую систему описаний и характеристики объектов разного рода, предусматривать надежные меры по защите от ошибок и потерь информации. Репозиторий обеспечивает:  - Распространение действия нового или скорректированного описания на информационное пространство всего проекта;  - Синхронизацию поступления информации от различных пользователей;  - Хранение версий проекта и его отдельных компонентов;  - Сборку любой запрошенной версии;  - Контроль информации на корректность, полноту и состоятельность. |
| Средства ввода | Служат для ввода данных в репозиторий и организации взаимодействия с CASE-пакетом. Должны поддерживать различные методологии и использоваться на всем жизненном цикле разными категориями разработчиков: аналитиками, проектировщиками, инженерами. |
| Средства анализа, проектирования и разработки | Предназначены для обеспечения планирования и анализа различных описаний, а также их преобразования в процессе разработки. |
| Средства вывода | Предназначены для документирования, управления проектом и кодовой генерации. |

Эти компоненты в совокупности должны:

-         Поддерживать графические модели;

-         Контролировать ошибки;

-         Организовывать и поддерживать репозиторий;

-         Поддерживать процесс проектирования и разработки.

**6.1.3 Поддержка графических моделей в CASE-средствах**

Графическая ориентация CASE-средств заключается в том, что для представления проектируемой АС применяются структурные диаграммы различных типов (например, DFD, ERD, STD и др.), дополнительное достоинство которых – применение в качестве наследуемой «двумерной» документации по проекту. Создание и модификация подобных диаграмм осуществляется с помощью специальных графических редакторов (диаграммеров), являющихся сервисными средствами на этапах анализа требований и проектирования спецификаций.

Некоторые возможности современных диаграммеровприведены в табл. 6.5.

Таблица 6.5

Некоторые возможности современных диаграммеров

|  |  |
| --- | --- |
| № | Описание возможности |
| 1 | Создание иерархических взаимосвязанных диаграмм, в которых комбинируются графические и текстовые объекты |
| 2 | Создание и редактирование объектов в любом месте диаграммы |
| 3 | Создание, перемещение и выравнивание групп объектов, изменение их размеров, масштабирование |
| 4 | Сохранение связей между объектами при их перемещении и изменении размеров |
| 5 | Автоматический контроль ошибок |

Реализация таких возможностей позволяет пользователю целиком сосредоточиться на проектировании, не отвлекаясь на решение второстепенных вопросов (размещение элементов диаграмм, их компоновка и др.).

**6.1.4 Контроль ошибок в CASE-средствах**

CASE-средства обеспечивают автоматическую верификацию и контроль проекта на полноту и состоятельность на ранних этапах жизненного цикла АС, что влияет на успех разработки в целом. Это подтверждается результатами анализа, выполненного многими фирмами и показавшего, что:

§ При традиционной организации работ ошибки проектирования и кодирования составляют соответственно 64% и 32% от общего числа ошибок;

§ Ошибки проектирования в 100 раз труднее обнаружить на этапе сопровождения АС, чем на этапах анализа требований и проектирования спецификаций.

В CASE-средствах диаграммеры и верификаторы способны осуществлять следующие типы контроля: синтаксический контроль, контроль полноты и состоятельности диаграмм, контроль декомпозиции функций, сквозной контроль диаграмм (табл. 6.6).

Таблица 6.6

Типы контроля, осуществляемого  диаграммерами и верификаторами

|  |  |
| --- | --- |
| Тип контроля | Описание и примеры |
| Синтаксический контроль | Контроль синтаксиса диаграмм и синтаксиса типов их элементов – обычно осуществляется при вводе и редактировании элементов диаграмм. Примеры контролируемых ситуаций:  -   По синтаксису: любой функциональный элемент диаграммы должен иметь минимум один входной и выходной поток; два элемента данных не могут быть непосредственно связаны;  -   По типам: функциональный элемент должен всегда использоваться для представления процедурного компонента; поток данных всегда должен быть представлен компонентом данных. |
| Контроль полноты и состоятельности диаграмм | Все элементы диаграмм должны быть идентифицированы и отражены в репозитории. Например, для DFD контролируются неименованные или несвязанные потоки данных, процессы и хранилища; источники и стоки данных (внешние сущности) вне контекстной диаграммы; хранилища данных на контекстной диаграмме и т.п. При анализе словаря данных должны выявляться циклические определения, эквивалентные определения, неопределённые объекты. |
| Контроль декомпозиции функций | Контроль декомпозиции функций включает оценку качества на основе различных характеристик ПО и частный семантический контроль. |
| Сквозной контроль диаграмм | Сквозной контроль диаграмм одного или различных типов на предмет их состоятельности по уровням – вертикальное и горизонтальное балансирование диаграмм. При вертикальном балансировании (диаграммы одного типа) выявляются несбалансированные потоки данных между детализируемой диаграммой и детализирующей ее диаграммой. Горизонтальное балансирование определяет некорректность между DFD, ERD и STD, словарями данных и миниспецификациями процессов.  Например, при балансировании DFD-ERD контролируется соответствие каждого хранилища данных на DFD сущности или отношению на  ERD.  Контроль*DFD-STD* осуществляется по следующим правилам:  -   Каждый управляющий процесс на DFD детализируется спецификацией STD, и наоборот, каждой STD должен соответствовать управляющий процесс;  -   Каждое условие (действие) в STD должно соответствовать входному (выходному) управляющему потоку на DFD, и наоборот, каждому управляющему потоку в зависимости от направленности должно соответствовать условие (действие) на STD. |

**6.1.5 Организация и поддержка репозитория**

Основные функции средств организации и поддержки репозитория – хранение, доступ, обновление, анализ и визуализация всей информации по проекту АС. Содержимое репозитория включает не только информационные объекты различных типов, но и отношения между их компонентами, а также правила использования или обработки этих компонентов. В репозитории могут храниться свыше ста типов объектов, например, структурные диаграммы, определения экранов и меню, проекты отчётов, описания данных, описания логики обработки и т.п. Каждый информационный объект в репозитории описывается перечислением его свойств (табл. 6.7).

На основе репозитория осуществляется интеграция CASE-средств и разделение системной информации между разработчиками. Перечень возможных уровней интеграции CASE-средств дан в табл. 6.7.

Таблица 6.7

Описание некоторых характеристик репозитория

|  |  |
| --- | --- |
| Характеристика | Описание |
| Описание информационного объекта | Каждый информационный объект в репозитории описывается перечислением его свойств: идентификатор, имена-синонимы, тип, текстовое описание, компоненты, файл-хранилище и область значений. Кроме этого хранятся: все отношения с другими объектами (например, все объекты, в которых данный объект используется, все перекрёстные ссылки и т.п.), правила формирования и редактирования объекта, а также контрольная информация о времени порождения объекта, времени его последнего обновления, кем и в каком проекте он был порожден, номере версии, возможности обновления и т.п. |
| Возможные уровни интеграции | Возможности репозитория обеспечивают несколько уровней интеграции: общий пользовательский интерфейс по всем средствам, передачу данных между средствами, интеграцию этапов разработки через единую систему представления фаз жизненного цикла, передачу данных и средств между аппаратурными платформами. |

Репозиторий является базой для стандартизации документации по проекту и контроля состоятельности проектных спецификаций. Все отчёты строятся автоматически по репозиторию. Характеристика основных типов отчётов приведена в табл. 6.8.

Таблица 6.8

Характеристика основных типов отчётов на базе репозитория

|  |  |
| --- | --- |
| Тип отчетов | Конкретные отчеты |
| Отчёты по содержимому | -        Сводки потоков данных и их компонентов;  -        Сводки всех пар интерфейсов в описывающих межмодульные отношения структурных диаграммах;  -        Списки входных и выходных потоков для каждого функционального блока диаграмм;  -        Списки измененных за определённый период объектов;  -        Истории всех изменений объектов;  -        Описания модулей;  -        Планы тестирования модулей и подпрограмм;  -        Списки всех данных и их атрибутов, а также отношений между их компонентами и правил их обработки. |
| Отчёты по перекрёстным ссылкам | -        Списки всех вызывающих и вызываемых модулей;  -        Списки всех объектов репозитория, к которым имеет доступ определённый разработчик;  -        Сводки диаграмм, использующие конкретные данные;  -        Маршруты движения данных от входа к выходу. |
| Отчёты по результатам анализа | -        Сводки балансирования диаграмм по уровням;  -        Списки неопределённых информационных объектов;  -        Списки неполных диаграмм;  -        Сводки результатов анализа структуры проекта;  -        Списки несогласованных в диаграммах и репозитории объектов;  -        Списки удалённых объектов. |
| Отчёты по декомпозиции объектов | Таблицы иерархий всех объектов модели. |

**6.1.6 Поддержка процесса проектирования и разработки в CASE**

При поддержке процесса проектирования и разработки основную роль играют: покрытие жизненного цикла; поддержка прототипирования; поддержка структурных методологий; автоматическая кодогенерация (табл. 6.9).

Таблица 6.9

Характеристика составляющих поддержки процесса

проектирования и разработки

|  |  |
| --- | --- |
| Составляющая | Характеристика |
| Покрытие жизненного цикла | При этом наибольшее внимание уделяется наиболее критическим этапам ЖЦ – анализу требований и проектированию спецификаций. Последние являются основой всего проекта, поэтому их полнота и корректность влияют на успех разработки в целом. |
| Поддержка прототипиро-вания | Поддержка прототипирования играет важную роль при автоматизации ранних этапов жизненного цикла. Соответствующие средства используются для определения системных требований и ответа на вопросы об определённом поведении системы. Такие средства, как генераторы меню, экранов и отчётов, позволяют быстро построить прототипы пользовательских интерфейсов и снабдить моделью функционирования системы с точки зрения конечного пользователя. Использование языков четвертого поколения (4GL) позволяет строить более сложные модели, при этом прототип позволяет промоделировать основные функции системы, но он не способен контролировать её ожидаемое поведение. Исполняемые языки спецификаций преобразуют процесс разработки к следующему виду: спецификации определяются и выполняются, затем производится их переопределение и/или корректировка, замыкаясь по циклу. |
| Поддержка структурных методологий | Поддержка структурных методологий осуществляется за счёт средств их автоматизации на следующих двух уровнях:  1.    Подготовка документации, графическая поддержка структурных диаграмм различных типов, продуцирование спецификаций для детализации функциональных блоков в диаграммах и структур данных на нижних уровнях (для таких спецификаций введён термин «миниспецификации»);  2.    Корректное использование шагов обработки в методологиях. |
| Кодегенерация | Кодегенерация осуществляется на основе репозитория. Позволяет автоматически построить до 80-90% объектов кодов или текстов программ на языках высокого уровня. При этом разными CASE-средствами поддерживаются практически все известные языки программирования.  Средства кодегенерации по отношению к полноте целевого продукта делится на:  1.       Средства генерации каркаса ПО – автоматически строится откомментированная логика (потоки управления) ПО, а также коды для БД, файлов, экранов, отчётов и т.п., а остальные фрагменты ПО кодируются вручную;  2.       Средства генерации полного продукта – из проектных спецификаций генерируется полная документированная программа, включая выполняемый код, набор исходных текстов программ, пользовательскую программную документацию, иногда наборы тестов и т.д. Все эти компоненты полной программы связываются в единый объект, хранящийся в репозитории для облегчения доступа и сопровождения. |

Идея автоматической кодогенерации на основе модели заключается в следующем.

Согласно структурно-функциональному подходу любая программа схематически может быть представлена в виде тройки: обрабатываемые данные, логический каркас обработки, линейные участки обработки. Схема БД может быть легко сгенерирована на основании ER-модели («сущность - связь»), а современные средства информационного моделирования обеспечивают такую генерацию. Логика обработки генерируется на основе диаграмм потоков данных; причём известны алгоритмы автоматического преобразования иерархий DFD в структурные карты; а с задачей получения из структурных карт соответствующих кодов легко справляется теория компиляции. Линейным участкам соответствуют миниспецификации моделей. Именно здесь лежит ключ к высокому проценту автоматически сгенерированного кода, существенно зависящему от метода задания миниспецификаций.

**6.2 Характеристика отдельных CASE-средств и их комплексов**

**6.2.1 Обзор CASE-средств Silverrun+JAM**

**6.2.1.1 Silverrun**

CASE-средство Silverrun американской фирмы Сomputer Systems Advisers, Inc. (CSA) используется для анализа и проектирования АС бизнес-класса и ориентировано в большей степени на спиральную модель ЖЦ. Оно применимо для поддержки любой методологии, основанной на раздельном построении функциональной и информационной моделей (DFD и ERD).

В системе имеются готовые настройки для ряда наиболее распространенных методологий, но основной методологией, поддерживаемой Silverrun, является DATARUN. Архитектура Silverrun позволяет наращивать среду разработки по мере необходимости. При этом для каждого понятия, введенного в проекте, имеется возможность добавления собственных описателей.

Silverrun имеет модульную структуру и состоит из четырех модулей, каждый из которых является самостоятельным продуктом и может приобретаться и использоваться без связи с остальными модулями (табл. 6.10).  Краткая характеристика возможностей  Silverrun дана в табл. 6.11.

Таблица 6.10

Описание модулей  Silverrun

|  |  |
| --- | --- |
| Модуль | Описание |
| BPM - Business Process Modeler | Модуль построения моделей бизнес-процессов в форме диаграмм потоков данных позволяет моделировать функционирование обследуемой организации или создаваемой АС. В модуле BPM обеспечена возможность работы с моделями большой сложности: автоматическая перенумерация, работа с деревом процессов (включая визуальное перетаскивание ветвей), отсоединение и присоединение частей модели для коллективной разработки. Диаграммы могут изображаться в нескольких предопределенных нотациях, включая Yourdon/DeMarco и Gane/Sarson. Имеется также возможность создавать собственные нотации, в том числе добавлять в число изображаемых на схеме дескрипторов определенные пользователем поля. |
| ERX - Entity-Relationship eXpert | Модуль концептуального моделирования данных обеспечивает построение моделей данных "сущность-связь", не привязанных к конкретной реализации. Этот модуль имеет встроенную экспертную систему, позволяющую создать корректную нормализованную модель данных посредством ответов на содержательные вопросы о взаимосвязи данных. Возможно автоматическое построение модели данных из описаний структур данных. Анализ функциональных зависимостей атрибутов дает возможность проверить соответствие модели требованиям третьей нормальной формы и обеспечить их выполнение. Проверенная модель передается в модуль RDM. |
| RDM - Relational Data Modeler | Модуль реляционного моделирования позволяет создавать детализированные модели "сущность-связь", предназначенные для реализации в реляционной базе данных. В этом модуле документируются все конструкции, связанные с построением базы данных: индексы, триггеры, хранимые процедуры и т.д. Гибкая изменяемая нотация и расширяемость репозитория позволяют работать по любой методологии. Возможность создавать подсхемы соответствует подходу ANSI SPARC к представлению схемы базы данных. На языке подсхем моделируются как узлы распределенной обработки, так и пользовательские представления. Этот модуль обеспечивает проектирование и полное документирование реляционных баз данных. |
| WRM - Workgroup Repository Manager | Менеджер репозитория рабочей группы применяется как словарь данных для хранения общей для всех моделей информации, а также обеспечивает интеграцию модулей Silverrun в единую среду проектирования. |

Таблица 6.11

Краткая характеристика возможностей  Silverrun

|  |  |
| --- | --- |
| Возможности | Характеристика возможностей |
| Взаимодействие с другими средствами | Для автоматической генерации схем баз данных у Silverrun существуют мосты к наиболее распространенным СУБД: Oracle, Informix, DB2, Ingres, Progress, SQL Server, SQLBase, Sybase. Для передачи данных в средства разработки приложений имеются мосты к языкам 4GL: JAM, PowerBuilder, SQL Windows, Uniface, NewEra, Delphi. Все мосты позволяют загрузить в Silverrun RDM информацию из каталогов соответствующих СУБД или языков 4GL. Это позволяет документировать, перепроектировать или переносить на новые платформы уже находящиеся в эксплуатации базы данных и прикладные системы.  Для обмена данными с другими средствами автоматизации проектирования, создания специализированных процедур анализа и проверки проектных спецификаций, составления специализированных отчетов в соответствии с различными стандартами в системе Silverrun имеется три способа выдачи проектной информации во внешние файлы:  ·    Система отчетов. Можно, определив содержимое отчета по репозиторию, выдать отчет в текстовый файл. Этот файл можно затем загрузить в текстовый редактор или включить в другой отчет;  ·    Система экспорта/импорта. Для более полного контроля над структурой файлов в системе экспорта/импорта имеется возможность определять не только содержимое экспортного файла, но и разделители записей, полей в записях, маркеры начала и конца текстовых полей. Файлы с указанной структурой можно не только формировать, но и загружать в репозиторий. Это дает возможность обмениваться данными с различными системами: другими CASE-средствами, СУБД, текстовыми редакторами и электронными таблицами;  ·    Хранение репозитория во внешних файлах через ODBC-драйверы. Для доступа к данным репозитория из наиболее распространенных систем управления базами данных обеспечена возможность хранить всю проектную информацию непосредственно в формате этих СУБД. |
| Групповая работа | Групповая работа поддерживается в системе Silverrun двумя способами:  ·     В стандартной однопользовательской версии имеется механизм контролируемого разделения и слияния моделей. Разделив модель на части, можно раздать их нескольким разработчикам. После детальной доработки модели объединяются в единые спецификации;  ·     Сетевая версия Silverrun позволяет осуществлять одновременную групповую работу с моделями, хранящимися в сетевом репозитории на базе СУБД Oracle, Sybase или Informix. При этом несколько разработчиков могут работать с одной и той же моделью, так как блокировка объектов происходит на уровне отдельных элементов модели. |
| Среда функционирования | Имеются реализации Silverrun для трех платформ - MS Windows, Macintosh и OS/2 Presentation Manager - с возможностью обмена проектными данными между ними.  Для функционирования в среде Windows необходимо иметь компьютер с процессором модели не ниже i486 и оперативную память объемом не менее 8 Мб (рекомендуется 16 Мб). На диске полная инсталляция Silverrun занимает 20 Мб. |

**6.2.1.2 JAM**

Средство разработки приложений JAM [28] (JYACC's Application Manager) - продукт фирмы JYACC (США).

Основной чертой JAM является его соответствие методологии RAD (см. выше п. 2.2.1), поскольку он позволяет достаточно быстро реализовать цикл разработки приложения, заключающийся в формировании очередной версии прототипа приложения с учетом требований, выявленных на предыдущем шаге, и предъявить его пользователю.

JAM имеет модульную структуру и состоит из следующих компонентов, описанных в табл. 6.12. Краткая характеристика возможностей  JAM дана в табл. 6.13.

Таблица 6.12

Описание модулей  JAM

|  |  |
| --- | --- |
| Модуль | Описание |
| Ядро системы | Ядро системы (собственно, сам JAM) является законченным продуктом и может самостоятельно использоваться для разработки приложений. Все остальные модули являются дополнительными и самостоятельно использоваться не могут.  Ядро системы включает в себя следующие основные компоненты:   * редактор экранов. В состав редактора экранов входят: среда разработки экранов, визуальный репозиторий объектов, собственная реляционная СУБД JAM - JDB, менеджер транзакций, отладчик, редактор стилей; * редактор меню; * набор вспомогательных утилит; * средства изготовления промышленной версии приложения.   JAM содержит встроенный язык программирования JPL (JAM Procedural Language), с помощью которого в случае необходимости можно написать модули, реализующие специфические действия. Данный язык является интерпретируемым, что упрощает отладку. Существует возможность обмена информацией между средой визуально построенного приложения и такими модулями. Кроме того, в JAM реализована возможность подключения внешних модулей, написанных на каком-либо языке, совместимом по вызовам функций с языком C. |
| JAM/DBi | Специализированные модули интерфейса к СУБД (JAM/DBi-Oracle, JAM/DBi-Informix, JAM/DBi-ODBC и т.д.) |
| JAM/RW | Модуль генератора отчетов |
| JAM/CASEi | Специализированные модули интерфейса к CASE-средствам (JAM/CASE-TeamWork, JAM/CASE-Innovator и т.д.) |
| JAM/TPi | Специализированные модули интерфейса к менеджерам транзакций (например, JAM/TPi-Server TUXEDO и т.д.) |
| Jterm | Специализированный эмулятор X-терминала |

Таблица 6.13

Краткая характеристика возможностей  JAM

|  |  |
| --- | --- |
| Возможности | Характеристика возможностей |
| Взаимодейст-вие с другими средствами | Непосредственное взаимодействие с СУБД реализуют модули JAM/DBi (Data Base interface). Способы реализации взаимодействия в JAM разделяются на два класса: ручные и автоматические. При ручном способе разработчик приложения самостоятельно пишет запросы на SQL, в которых как источниками, так и адресатами приема результатов выполнения запроса могут быть как интерфейсные элементы визуально спроектированного внешнего уровня, так и внутренние, невидимые для конечного пользователя переменные. Автоматический режим, реализуемый менеджером транзакций JAM, осуществим для типовых и наиболее распространенных видов операций с БД, с учетом достаточно сложных взаимосвязей между таблицами БД и автоматическим управлением атрибутами экранных полей ввода/вывода в зависимости от вида транзакции (чтение, запись и т.д.), в которой участвует сгенерированный запрос.  JAM позволяет строить приложения для работы более чем с 20 СУБД: ORACLE, Informix, Sybase, Ingres, InterBase, NetWare SQL Server, Rdb, DB2, ODBC-совместимые СУБД и др.  Отличительной чертой JAM является высокий уровень переносимости приложений между различными платформами (MS DOS/MS Windows, SunOS, Solaris (i80x86, SPARC), HP-UX, AIX, VMS/Open VMS и др.).  При переносе приложения с платформы на платформу, как правило, требуется лишь определить соответствие между физическими устройствами ввода/вывода и их логическими представлениями для приложения.  Использование SQL в качестве средства взаимодействия с СУБД также создает предпосылки для обеспечения переносимости между СУБД.  При росте нагрузки на систему и сложности решаемых задач (распределенность и гетерогенность используемых ресурсов, количество одновременно подключенных пользователей, сложность логики приложения) применяется трехзвенная модель архитектуры "клиент-сервер" с использованием менеджеров транзакций.  *Интерфейс JAM/CASE* подобен интерфейсу к СУБД и позволяет осуществить обмен информацией между репозиторием объектов JAM и репозиторием CASE-средства. Отличие заключается в том, что в случае интерфейса к CASE этот обмен является двунаправленным.   * Мост (интерфейс) Silverrun-RDM <-> JAM реализует взаимодействие между CASE-средством Silverrun и JAM (перенос схемы базы данных и экранных форм приложения).   Режим реинжиниринга позволяет переносить модификации всех свойств экранов JAM, импортированных ранее из RDM, в схему Silverrun. На этом этапе для контроля целостности базы данных не допускаются изменения схемы в виде добавления или удаления таблиц и полей таблиц. |
| Групповая работа | Ядро JAM имеет встроенный интерфейс к средствам конфигурационного управления (PVCS на платформе Windows и SCCS на платформе UNIX). Под управлением этих систем передаются библиотеки экранов и/или репозитории. При отсутствии таких систем JAM самостоятельно реализует часть функций поддержки групповой разработки. |
| Среда функциониро-вания | JAM, как среда разработки, и приложения, построенные с его использованием, не являются ресурсоемкими системами. Например, на платформе MS-Windows достаточно иметь 8MB оперативной памяти и 50 MB дискового пространства для среды разработки. На UNIX-платформах требования к аппаратуре определяются самой операционной системой. |

**6.2.2 Обзор CASE-средств Designer/2000 + Developer/2000**

CASE-средство Designer/2000 фирмы ORACLE является интегрированным CASE-средством, обеспечивающим в совокупности со средствами разработки приложений Developer/2000 поддержку полного ЖЦ ПО для систем, использующих СУБД ORACLE.

Designer/2000 представляет собой семейство методологий и поддерживающих их программных продуктов. Базовая методология Designer/2000 - структурная методология проектирования систем CASE\*Method, полностью охватывающая все этапы жизненного цикла АС. Designer/2000 обеспечивает графический интерфейс при разработке различных моделей (диаграмм) предметной области.

В состав Designer/2000 входят компоненты, описанные в табл. 6.14.

Таблица 6.14

Компоненты Designer/2000

|  |  |
| --- | --- |
| **Компонент** | **Описание** |
| Repository Administrator | Средства управления репозиторием - создание и удаление приложений, управление доступом к данным со стороны различных пользователей, экспорт и импорт данных. |
| Repository Object Navigator | Средства доступа к репозиторию, обеспечивающие многооконный объектно-ориентированный интерфейс доступа ко всем элементам репозитория. |
| Process Modeller | Средство анализа и моделирования деловой деятельности, основывающееся на концепциях реинжиниринга бизнес-процессов (BPR) и глобальной системы управления качеством (TQM - Total Quality Management). |
| Systems Modeller | Набор средств построения функциональных и информационных моделей проектируемой ИС, включающий средства для построения диаграмм "сущность-связь" (ERD), диаграмм функциональных иерархий (Function Hierarchy Diagrammer), диаграмм потоков данных (DFD) и средство анализа и модификации связей объектов репозитория различных типов (Matrix Diagrammer). |
| Systems Designer | Набор средств проектирования АС, включающий средство построения структуры реляционной базы данных (Data Diagrammer), а также средства построения диаграмм, отображающих взаимодействие с данными, иерархию, структуру и логику приложений, реализуемую хранимыми процедурами на языке PL/SQL (Module Data Diagrammer, Module Structure Diagrammer и Module Logic Navigator). |
| Server Generator | Генератор описаний объектов БД ORACLE (таблиц, индексов, ключей, последовательностей и т.д.). Помимо продуктов ORACLE, генерация и реинжиниринг БД может выполняться для СУБД Informix, DB/2, Microsoft SQL Server, Sybase, а также для стандарта ANSI SQL DDL и баз данных, доступ к которым реализуется посредством ODBC. |
| Forms Generator | Генератор приложений для ORACLE Forms: генерируемые приложения включают в себя различные экранные формы, средства контроля данных, проверки ограничений целостности и автоматические подсказки. Дальнейшая работа с приложением выполняется в среде Developer/2000. |
| Repository Reports | Генератор стандартных отчетов, интегрированный с ORACLE Reports и позволяющий русифицировать отчеты, а также изменять структурное представление информации. |

Среда функционирования Designer/2000 и Developer/2000 – MS Windows. Физическая среда хранения репозитория - БД ORACLE. Генерация приложений, помимо продуктов ORACLE, выполняется для Visual Basic.

Designer/2000 можно интегрировать с другими средствами, используя открытый интерфейс приложений API. Кроме того, можно использовать средство ORACLE CASE Exchange для экспорта/импорта объектов репозитория с целью обмена информацией с другими CASE-средствами.

Developer/2000 обеспечивает разработку переносимых приложений, работающих в графической среде Windows, Macintosh или Motif. В среде Windows интеграция приложений Developer/2000 с другими средствами реализуется через механизм OLE и управляющие элементы VBX. Взаимодействие приложений с другими СУБД (DB/2, DB2/400, Rdb) реализуется с помощью средств ORACLE Client Adapter для ODBC, ORACLE Open Gateway и API.

**6.2.3 Обзор CASE-средств ERwin + BPwin**

**6.2.3.1 ERwin**

ERwin: изначально продукт фирмы Logic Works. В настоящее время производится фирмой Computer Associates Technologies и имеет название CA ERwin Data Modeler (v.8.0). Средство концептуального моделирования БД, использующее методологию IDEF1X (см. п. 4.4). Ключевые функции CA ERwin Data Modeler Standard Edition R8 представлены в табл. 6.15.

Таблица 6.15

Ключевые функции CA ERwin Data Modeler Standard Edition R8

|  |  |
| --- | --- |
| **Функция** | **Описание** |
| Диаграммы и визуализация | Реализует современную диаграммную технологию, предоставляющую гибкость графического редактора в сочетании с силой средства моделирования баз данных. Улучшается вид моделей, облегчается их создание и управление ими. |
| Редактор метаданных | позволяет редактировать различные наборы объектов в рамках единого интерфейса вида электронной таблицы, позволяющего вносить массовые изменения, экспортировать в Excel, выполнять запросы к метаданным. |
| Полное сравнение | Функция, автоматизирующая полную двунаправленную синхронизацию моделей, скриптов и БД, сравнивая каждый из элементов, отображая все различия, и позволяет выполнить двунаправленное выборочное обновление. Если изменения модели должны найти отражение в БД, ERwin может автоматически сгенерировать скрипт ALTER. |
| Создание структуры базы данных | Позволяет создать структуру БД непосредственно из визуальных моделей, увеличивая эффективность и снижая ошибки. Поддержка различных баз данных включает оптимизированные шаблоны триггеров целостности ссылок и богатый универсальный макроязык, позволяющий разработчикам настраивать триггеры, скрипты и хранимые процедуры. Настраиваемые шаблоны облегчают полную физическую реализацию модели и определений. |
| Реверсивное проектирование баз данных | Недокументированная информация, содержащаяся внутри скриптов SQL или БД, может быть визуализирована или повторно использована для создания новых моделей данных и/или объектов базы данных. |
| Стандарты проектирования для многократ-ного использо-вания | Расширенный набор объектов моделирования для многократного использования позволяет создавать, поддерживать, применять и использовать отображение трансформаций имен, отображение типов данных, шаблоны создания схем, определения доменов и множество других стандартов моделирования для увеличения возможности многократного использования внутри организации. |
| Отчеты и печать | Каждая копия CA ERwin Data Modeler включает копию SAP Business Objects Crystal Reports. Можно использовать встроенные отчеты или создавать свои собственные. Универсальный интерфейс ODBC доступен для заказчиков, которые используют другие инструменты отчетности. Отчеты могут быть сгенерированы в различных форматах, включая HTML, PDF, RTF и TXT. |
| Интеграция и обмен метаданных с другими инструментами | ERwin легко объединить с другими проектами и инструментами, благодаря средствам импорта и экспорта для разнообразных источников, включая BI tools, MDM hubs, другие средства моделирования данных, Extract, Transform, инструменты Load (ETL) и инструменты Unified Modeling Language (UML). |

Кроме этого для ряда средств разработки приложений (SQLWindows, Delphi, Visual Basic и др.) выполняется генерация форм и прототипов приложений. Сетевая версия Erwin обеспечивает согласованное проектирование БД и приложений в рамках рабочей группы.

Работа в ERwin по проектированию БД обычно изучается в рамках дисциплины «Базы и банки данных» (БиБД).

**6.2.3.2 BPwin**

Основные характеристики CASE-средства BPwin сведены в табл. 6.16.

Таблица 6.16

Основные характеристики BPwin

|  |  |
| --- | --- |
| Характеристика | **Описание** |
| Производитель и наименование | Изначально продукт фирмы Logic Works, сейчас – фирмы Computer Associates Technologies, называемый полностью CA ERwin Process Modeler (еще одно название AllFusion Process Modeler) |
| Поддерживаемые методологии | Поддерживает три методологии моделирования: функциональное моделирование (IDEF0); описание бизнес-процессов (IDEF3); диаграммы потоков данных (DFD). Каждая из них может быть выполнена отдельно с помощью BPwin, но их совокупность заключённая в модель даёт аналитику полную картину предметной области клиента. |
| Возможности системного анализа | Мощное средство системного анализа деловой и производственной активности, позволяющее адекватно отслеживать соответствие структуры бизнеса, документооборота, финансовых потоков жестким и динамичным требованиям экономики. BPwin - незаменимый инструмент менеджеров и бизнес-аналитиков, а в руках системных аналитиков и разработчиков - еще и мощное средство моделирования процессов при создании корпоративных информационных систем. BPwin предназначен для облегчения труда и увеличения производительности системного аналитика на первом этапе разработки системы. |
| Описательные возможности | -       Мощные редакторы для описания операций, связей и вычисления затрат на выполнение работ  -       Иерархическая структура диаграмм, облегчающая последовательное уточнение элементов модели  -       Контекстные диаграммы для описания границ системы, области действия, назначения объектов  -       Декомпозиционные диаграммы для описания особенностей взаимодействия различных процессов  Расширенные возможности по поддержанию ссылочной целостности |
| Интегрируемость | -       Экспорт моделей в средства имитационного моделирования  -       Интеграция и связь со средством проектирования баз данных ERwin (методология IDEF1X)  -       Поддержка свойств, определяемых пользователем. Описание моделей может быть расширено за счет свойств, определяемых пользователем, включая мультимедийные документы.  Интеграция с ModelMart. Сервер приложений для программных продуктов CA ModelMart поддерживает мощный набор инструментальных программных средств, обеспечивающих совместное (групповое) проектирование и разработку программных систем, включая механизмы объединения моделей и анализа изменений, контроль версий, возможность создания "компонентов" модели и т.д. Для организации хранилища моделей в ModelMart используются СУБД на платформах Oracle, Sybase, Informix или SQL Server. Кроме того, поддерживаются прямые связи ModelMart с ERwin и BPwin. |

BPwin имеет достаточно простой и интуитивно понятный интерфейс пользователя (рис. 6.1).

Рис. 6.1  Интегрированная среда разработки модели BPwin

При создании новой модели открывается диалоговое окно, показанное на рис. 6.2. В BPwin возможно построение смешанных моделей, т. е. модель может содержать одновременно диаграммы как IDEF0, так и IDEF3 и DFD.

Рис. 6.2  Диалоговое окно создания модели

Основы работы в BPwin рассматриваются в лабораторном практикуме. Далее рассмотрены лишь некоторые частные аспекты такой работы.

Как известно, IDEF0-модель предполагает наличие четко сформулированной цели, единственного субъекта моделирования и одной точки зрения. Для внесения соответствующих исходных данных следует выбрать пункт меню Model/Model Properties, вызывающий диалоговое окно Model Properties (рис. 6.3).

Рис. 6.3  Диалоговое окно задания свойств модели

Согласно технологии проектирования АС сначала создаётся модель существующей организации работ AS-IS (как есть). Найденные в модели AS-IS недостатки можно исправить при создании модели AS-ТО-ВЕ (как будет) — модели новой организации бизнес-процессов. Только на основе этой модели строится модель данных, прототип и окончательный вариант АС.

Результат описания модели можно получить в отчете Model Report. Диалоговое окно настройки отчета по модели вызывается из пункта меню Tools/Reports/Model Report (рис. 6.4). На рис. 6.5 представлен пример отчета. Всего имеется семь типов отчетов (табл. 6.17).

Рис. 6.4  Диалоговое окно для формирования отчета по модели

Рис. 6.5  Предварительный просмотр отчета

Таблица 6.17

Типы отчетов BPwin

|  |  |
| --- | --- |
| **Отчет** | **Описание** |
| Model Report | Включает информацию о контексте модели — имя модели, точку зрения, область, цель, имя автора, дату создания и др. |
| Diagram Report | Отчет по конкретной диаграмме. Включает список объектов (*работ*, стрелок, хранилищ данных, внешних ссылок и т. д.). |
| Diagram Object Report | Наиболее полный отчет по модели. Может включать полный список объектов модели (*работ*, стрелок с указанием их типа и др.) и свойства, определяемые пользователем |
| Activity Cost Report | Отчет о результатах стоимостного анализа. |
| Arrow Report | Отчет по *стрелкам*. Может содержать информацию из словаря стрелок, информацию о работе-источнике, работе-назначении *стрелки* и информацию о разветвлении и слиянии стрелок. |
| Data Usage Report | Отчет о результатах связывания моделей процессов и данных. |
| Model Consistency Report | Отчет, содержащий список синтаксических ошибок модели |

Как правило, в ходе проектирования АС строится несколько моделей AS-ТО-ВЕ, из которых по какому-либо критерию выбирается наилучшая с точки зрения эффективности бизнес-процессов. Проблема состоит в том, что таких критериев много и непросто определить важнейший. BPwin предоставляет аналитику два инструмента для оценки модели:

1)  функционально*-*стоимостный анализ (ФСА), основанный на работах (Activity Based Costing, ABC),

2)  свойства, определяемые пользователем (User Defined Properties, UDP).

**6.2.3.3 Функционально-стоимостный анализ в BPwin**

ФСА ABC – это основанная на модели работ технология выявления и исследования стоимости выполнения той или иной функции (действия). Исходными данными являются затраты на ресурсы (материалы, персонал и т.д.). ABC обеспечивает достаточно точный метод расчета стоимости производства продукции, основанный на стоимости выполнения всех технологических операций, выполняемых при ее выпуске.

ФСА ABC может проводиться только тогда, когда модель работы последовательная (следует синтаксическим правилам IDEF0), корректная (отражает бизнес), полная (охватывает всю рассматриваемую область) и стабильная (проходит цикл экспертизы без изменений), другими словами, когда создание модели работы закончено.

ABC включает следующие основные понятия: объект затрат, двигатель затрат,центр затрат, которые поясняются в табл. 6.18 и на рис.6.6.

Таблица 6.18

Основные понятия ФСА ABC

|  |  |
| --- | --- |
| Понятие | Описание |
| **Объект затрат** | Причина, по которой работа выполняется, обычно основной выход работы. Стоимость работ есть суммарная стоимость объектов затрат ("Сборка и тестирование компьютеров", рис. 6.6) |
| Двигатель затрат | Характеристики входов и управлений работы, которые влияют на то, как выполняется и как долго длится работа ("Заказы клиентов", "Правила сборки и тестирования", "Персонал производственного отдела", рис. 6.6) |
| Центры затрат | Могут трактоваться как статьи расхода. |

Рис. 6.6  Иллюстрация терминов ABC

При проведении ФСА в BPwin сначала задаются единицы измерения времени и денег в диалоговом окне Model Properties на вкладке ABC Units (рис. 6.7).

Рис. 6.7  Настройка единиц измерения валюты и времени

Затем описываются центры затрат (cost centers) в диалоговом окне Cost Center Editor (вход из пункта меню Model, рис. 6.8). Список центров затрат упорядочен. Порядок в списке можно менять при помощи стрелок, расположенных справа от списка.

Рис. 6.8  Диалоговое окно Cost Center Editor

Для задания стоимости работы (для каждой работы на диаграмме декомпозиции) следует щелкнуть правой кнопкой мыши по работе и из всплывающего меню выбрать Cost (рис. 6.9).

Рис. 6.9  Задание стоимости работ в диалоговом окне Activity Properties/Cost

Общие затраты по работе рассчитываются как сумма по всем центрам затрат*.* Если во всех работах модели включен режим Compute from Decompositions (рис. 6.9), подобные вычисления автоматически проводятся по всей иерархии работ снизу вверх (рис. 6.10).

Рис. 6.10  Вычисление затрат родительской работы

Этот достаточно упрощенный принцип подсчета справедлив, если работы выполняются последовательно. Если схема выполнения более сложная (например, работы производятся альтернативно), можно отказаться от автоматического подсчета и задать итоговые суммы для каждой работы вручную (режим Override Decompositions, рис. 6.9). В этом случае результаты расчетов с нижних уровней декомпозиции будут игнорироваться, и при расчетах на верхних уровнях будет учитываться сумма, заданная вручную. На любом уровне результаты расчетов сохраняются независимо от выбранного режима, поэтому при выключении опции Override Decompositions расчет снизу вверх производится обычным образом. Результаты ФСА могут существенно повлиять на очередность выполнения работ (табл. 6.19).

Таблица 6.19

Пример использования ФСА ABC

|  |  |
| --- | --- |
| Шаг | Описание |
| 1 | Предположим, что для оценки качества изделия необходимо провести три работы:   * внешний осмотр — стоимость 50 руб.; * пробное включение — стоимость 150 руб.; * испытание на стенде — стоимость 300 руб. |
| 2 | Предположим также, что с точки зрения технологии очередность проведения работ несущественна, а вероятность выявления брака одинакова (50%). Пусть необходимо проверить восемь изделий. |
| 3 | Если проводить работы в убывающем по стоимости порядке, то затраты на получение готового изделия составят:  300 руб. (испытание на стенде)\*8 +150 руб. (пробное включение) \*4 + 50 руб. (внешний осмотр) \*2 = 3100 руб. |
| 4 | Если проводить работы в возрастающем по стоимости порядке, то на получение готового изделия будет затрачено:  50 руб. (внешний осмотр) \*8 +150 руб. (пробное включение) \*4 + 300 руб. (испытание на стенде) \*2 = 1600 руб. |
| 5 | Следовательно, с целью минимизации затрат первой должна быть выполнена наиболее дешевая работа, затем — средняя по стоимости и в конце — наиболее дорогая. |

Результаты ФСА наглядно представляются на специальном отчете BPwin, настройка которого производится в диалоговом окне Activity Cost Report (меню Tools/Reports/Activity Cost Report) (рис. 6.11).

Рис. 6.11  Диалоговое окно настройки отчета по стоимости работ

**6.2.3.4 Свойства, определяемые пользователем (UDP)**

UDP позволяют провести дополнительный анализ, хотя и без суммирующих подсчетов, как в ФСА АВС. Для описания UDP служит диалоговое окно User-Defined Property Editor (меню Model/UDP Definition Editor) (рис. 6.12). Имеется возможность задания 18 различных типов UDP, в том числе управляющих команд и массивов, объединенных по категориям.

Рис. 6.12  Диалоговое окно описания UDP

Каждой работе можно поставить в соответствие набор UDP. Для этого следует щелкнуть правой кнопкой мыши по работе и выбрать пункт меню UDP. Во вкладке UDP Values диалогового окна IDEF0 Activity Properties можно задать значения UDP. Результат задания можно проанализировать в отчете Diagram Object Report (вход через пункты меню Tools/Report/Diagram Object Report) (рис. 6.13).

Рис. 6.13  Диалоговое окно настройки отчета Diagram Object Report

**6.2.4 Обзор CASE-средства Rational Rose**

Rational Rose - CASE-средство фирмы Rational Software Corporation (США) - предназначено для автоматизации этапов анализа и проектирования АС и их ПО, генерации кодов на различных языках и выпуска проектной документации. Rational Rose использует методологию объектно-ориентированного анализа и проектирования на базе UML.

Конкретный вариант Rational Rose определяется языком, на котором генерируются коды программ (C++, Smalltalk, Ada, PowerBuilder, SQLWindows и ObjectPro). Основной вариант - Rational Rose/C++ - позволяет разрабатывать проектную документацию в виде диаграмм и спецификаций, а также генерировать программные коды на С++.

Кроме того, Rational Rose содержит средства реинжиниринга программ, обеспечивающие повторное использование программных компонентов в новых проектах. Общая характеристика структуры и функций Rational Rose дана в табл. 6.20.

Таблица 6.20

Общая характеристика структуры и функций Rational Rose

|  |  |
| --- | --- |
| Объект | Описание |
| Струк-тура | В составе Rational Rose можно выделить 6 основных структурных компонентов: репозиторий, графический интерфейс пользователя, средства просмотра проекта (browser), средства контроля проекта, средства сбора статистики и генератор документов. К ним добавляются генератор кодов (индивидуальный для каждого языка) и анализатор для С++, обеспечивающий реинжиниринг - восстановление модели проекта по исходным текстам программ |
| Репози-тарий | Представляет собой объектно-ориентированную базу данных. Средства просмотра обеспечивают "навигацию" по проекту, в том числе, перемещение по иерархиям классов и подсистем, переключение от одного вида диаграмм к другому и т. д. Средства контроля и сбора статистики дают возможность находить и устранять ошибки по мере развития проекта, а не после завершения его описания. Генератор отчетов формирует тексты выходных документов на основе содержащейся в репозитории информации. |
| Форми-руемые докумен-ты | В результате разработки проекта с помощью CASE-средства Rational Rose формируются следующие документы:   * диаграммы классов; * диаграммы состояний; * диаграммы сценариев; * диаграммы модулей; * диаграммы процессов; * спецификации классов, объектов, атрибутов и операций * заготовки текстов программ; * модель разрабатываемой программной системы.   Последний из перечисленных документов является текстовым файлом, содержащим всю необходимую информацию о проекте (в том числе необходимую для получения всех диаграмм и спецификаций). |
| Автома-тическая генера-ция кодов прог-рамм на языке С++ | Соответствующие средства, используя информацию, содержащуюся в логической и физической моделях проекта, формируют файлы заголовков и файлы описаний классов и объектов. Создаваемый таким образом скелет программы может быть уточнен путем прямого программирования на языке С++. Тексты программ являются заготовками для последующей работы программистов. Они формируются в рабочем каталоге в виде файлов типов .h (заголовки, содержащие описания классов) и .cpp (заготовки программ для методов). Система включает в программные файлы собственные комментарии, которые начинаются с последовательности символов //##. Состав информации, включаемой в программные файлы, определяется либо по умолчанию, либо по усмотрению пользователя. В дальнейшем эти исходные тексты развиваются программистами в полноценные программы. |
| Анализа-тор кодов С++ | Реализован в виде отдельного программного модуля. Его назначение состоит в том, чтобы создавать модули проектов в форме Rational Rose на основе информации, содержащейся в определяемых пользователем исходных текстах на С++. В процессе работы анализатор осуществляет контроль правильности исходных текстов и диагностику ошибок. Модель, полученная в результате его работы, может целиком или фрагментарно использоваться в различных проектах. Анализатор обладает широкими возможностями настройки по входу и выходу. Например, можно определить типы исходных файлов, базовый компилятор, задать, какая информация должна быть включена в формируемую модель и какие элементы выходной модели следует выводить на экран. Таким образом, Rational Rose/С++ обеспечивает возможность повторного использования программных компонентов. |
| Интегра-ция | Rational Rose интегрируется со средством управления конфигурацией и контроля версий PVCS, необходимым для организации групповой работы и управления проектом, и со средством SoDA документирования проектов. Интеграция Rational Rose и SoDA обеспечивается средствами SoDA. |
| Группо-вая работа | Для организации групповой работы в Rational Rose возможно разбиение модели на управляемые подмодели. Каждая из них независимо сохраняется на диске или загружается в модель. В качестве подмодели может выступать категория классов или подсистема.  Для управляемой подмодели предусмотрены операции:   * загрузка подмодели в память; * выгрузка подмодели из памяти; * сохранение подмодели на диске в виде отдельного файла; * установка защиты от модификации; * замена подмодели в памяти на новую.   Наиболее эффективно групповая работа организуется при интеграции Rational Rose со специальными средствами управления конфигурацией и контроля версий (PVCS). В этом случае защита от модификации устанавливается на все управляемые подмодели, кроме тех, которые выделены конкретному разработчику. В этом случае признак защиты от записи устанавливается для файлов, которые содержат подмодели, поэтому при считывании "чужих" подмоделей защита их от модификации сохраняется, т.е. случайные воздействия окажутся невозможными. |
| Среда функ-цииони-рования | Rational Rose функционирует на различных платформах: IBM PC (в среде Windows), Sun SPARC stations (UNIX, Solaris, SunOS), Hewlett-Packard (HP UX), IBM RS/6000 (AIX). Для работы системы необходимо выполнение следующих требований: Платформа Windows - процессор 80486 или выше, память12Mб, пространство на диске 8Mб + 1-3Mб для одной модели.  Платформа UNIX - память 32+(16\*число пользователей) Mб, пространство на диске 30Mб + 20 при инсталляции + 1-3Mб для одной модели. |

**6.2.5 Обзор CASE-средства Sparx Enterprise Architect (SEA)**

Sparx Enterprise Architect v.7.5 (США) – высокопроизводительный инструмент объектно-ориентированного анализа и проектирования, основанный на стандарте UML 2.1 и используемый для моделирования и создания АС и их ПО. Покрывает весь процесс разработки от формирования требований к системе до её полной реализации. Представляет собой средства надежной и эффективной визуализации и организации взаимодействия в коллективе. Поддерживает все аспекты цикла разработки, обеспечивая полную трассировку от начала проектирования до размещения и поддержки. Обеспечивает поддержку тестирования, управления сопровождением и изменениями. Отличается низкими издержками на установку, высокой производительностью и интуитивно понятным интерфейсом.

Краткая характеристика возможностей Sparx Enterprise Architect дана в табл. 6.21.

Таблица 6.21

Краткая характеристика возможностей Sparx Enterprise Architect

|  |  |
| --- | --- |
| Объект | Описание |
| Организация работы | Архитектурно Enterprise Architect представляет собой программу – рабочее место EA, из которого осуществляется соединение через собственный драйвер БД с проектным репозиторием, организованным в виде базы данных. На рабочем месте хранятся пользовательские настройки этого рабочего места, такие как настройки отображения панелей инструментов, набор горячих клавиш и т.д. |
| Репозиторий | В проектном репозитории хранятся следующие элементы моделирования:  -       объекты модели, такие как UML-элементы и пакеты;  -       коннекторы, которые связывают взаимодействующие объекты;  -       диаграммы, отображающие объекты, коннекторы и ссылки на другие диаграммы.  При этом один элемент может быть отображен на нескольких диаграммах, но физически как объект базы данных он хранится только в одном экземпляре. Т.о. удаление элемента на диаграмме не вызывает удаление объекта из репозитория.  Также в проектном репозитории хранится дополнительная и служебная информация:  -       дополнительные справочники, такие как глоссарий, авторов моделей, задач, проблем, дефектов и пр.;  -       настроечные справочники, такие как типы стереотипов, пользовательских тегов, шаблонов отчетов и т.п.  -       шаблоны проектирования, такие как UML-паттерны и UML-профили, позволяющие сохранять и быстро воспроизводить типовые решения, смоделированные ранее.  -       базовые линии, т.е. моментальные снимки состояния пакетов в XML-формате.  Для обмена информацией между репозиториями используется экспорт/импорт файлов XML-формата. |
| Диаграммы | Поддерживает все 13 диаграмм UML 2 (Классов, Объектов, Композитных структур, Пакетов, Компонент, Развертывания, Вариантов использования, Коммуникаций, Последовательностей, Схем взаимодействия, Деятельностей, Состояний, Временные, Анализа, Пользовательские) |
| Моделирова-ние | Позволяет моделировать бизнес-процессы, веб-сайты, пользовательские интерфейсы, сети, конфигурации аппаратного обеспечения, сообщения и т.д., оценивать размер трудозатрат проектных работ в часах, фиксировать и трассировать требования, ресурсы, тест-планы, дефекты и запросы на изменения. |
| Генерация кодов | Поддерживает генерацию и обратный инжиниринг исходного кода для многих популярных языков, включая: C++, C#, Java, Delphi, VB.Net, Visual Basic, ActionScript, PHP, Python и др.; поддерживает обратный инжиниринг файлов Java .jar и даже бинарных сборок .Net. |
| Создание документации | Имеет мощные инструменты для создания документации и разного рода отчетов. Имеет WYSIWYG редактор шаблонов. Из моделей может быть быстро создана документация в стандартном rtf-формате и импортирована в Word для финального редактирования, так же доступна генерация HTML-документов. |
| Работа с БД | Позволяет проектировать и генерировать элементы баз данных. В качестве сервера БД могут использоваться SQL Server, MySQL, Oracle 9i и 10g, PostgreSQL, Adaptive Server Anywhere, MSDE Server, Progress OpenEdge. |

**6.2.6 Обзор вспомогательных средств поддержки жизненного цикла автоматизированных систем**

**6.2.6.1 Средства конфигурационного управления**

Цель конфигурационного управления (КУ) - обеспечить управляемость и контролируемость процессов разработки и сопровождения АС. Для этого необходима точная и достоверная информация о состоянии проекта и его компонентов в каждый момент времени, а также обо всех предполагаемых и выполненных изменениях.

Для решения задач КУ применяются методы и средства, обеспечивающие идентификацию состояния компонентов, учет номенклатуры всех компонентов и модификаций системы в целом, контроль за вносимыми изменениями в компоненты, структуру системы и ее функции, а также координированное управление развитием функций и улучшением характеристик системы.

Наиболее распространенным средством конфигурационного управления является PVCS (Project Version Control System) фирмы Intersolv (США), включающее ряд самостоятельных продуктов. Их краткая характеристика дана в табл. 6.22.

Таблица 6.22

Краткая характеристика компонентов средства КУ PVCS фирмы Intersolv

|  |  |
| --- | --- |
| Компонент | Описание |
| PVCS Version Manager | Предназначен для управления всеми компонентами проекта и ведения планомерной многоверсионной и многоплатформенной разработки силами команды разработчиков в условиях одной или нескольких локальных сетей. Понятие "проект" трактуется как совокупность файлов. В процессе работы над проектом промежуточное состояние файлов периодически сохраняется в архиве проекта, ведутся записи о времени сохранения, соответствии друг другу нескольких вариантов разных файлов проекта. Кроме этого, фиксируются имена разработчиков, ответственных за тот или иной файл, состав файлов промежуточных версий проекта и др. Это позволяет вернуться при необходимости к какому-либо из предыдущих состояний файла (например, при обнаружении ошибки, которую в данный момент трудно исправить).  Используется в рабочих группах. Имеющаяся система блокировок позволяет предотвратить одновременное внесение изменений в один и тот же файл. В то же время позволяет разработчикам работать с собственными версиями общего файла с полуавтоматическим разрешением конфликтов между ними.  Результатом работы PVCS Version Manager является созданный средствами файловой системы репозиторий, хранящий в компактной форме все рабочие версии программного продукта вместе с необходимыми комментариями и метками. |
| PVCS Tracker | Специализированная надстройка над офисной электронной почтой, предназначенная для обработки сообщений об ошибках в проекте, доставке их исполнителям и контроля за исполнением. Интеграция с PVCS Version Manager дает возможность связывать с сообщениями те или иные компоненты проекта. Отчетные возможности PVCS Tracker включают множество разновидностей графиков и диаграмм, отражающих состояние проекта и процесса его отладки, срезы по различным компонентам проекта, разработчикам и тестировщикам. С их помощью можно наглядно показать текущее состояние работы над проектом и ее временные тенденции.  Персонал, работающий с PVCS Tracker делится на пять групп в зависимости от их обязанностей: пользователи, разработчики, группа тестирования и контроля качества, группа технической поддержки и сопровождения, управленческий персонал. Этим пяти группам персонала соответствуют пять предопределенных групп PVCS Tracker. В дополнение к этим пяти предопределенным группам, существует группа администратора базы данных и 11 дополнительных групп, которые могут быть настроены в соответствии со специфическими должностными обязанностями сотрудников, использующих PVCS Tracker.  Требование или замечание поступающее в PVCS Tracker проходит четыре этапа обработки:  ·   регистрация - внесение замечания в базу данных;  ·   распределение - назначение исполнителя и сроков исполнения;  ·   исполнение - устранение замечания, что может вызвать дополнительные замечания или требования на дополнительные работы;  ·   приемка - приемка работ и снятие их с контроля или направление на доработку. |
| PVCS Configura-tion Builder | Предназначен для сборки окончательного продукта из компонентов проекта. Позволяет описывать процесс сборки как на стандартном языке MAKE, так и на собственном внутреннем языке, имеющем существенно большие возможности. Позволяет осуществлять сборку программного продукта на основании файлов, хранящихся в репозитории PVCS Version Manager. Обычная процедура сборки программного продукта с помощью PVCS Configuration Builder состоит из трех шагов:  ·   строится файл зависимостей между исходными модулями;  ·   в этот файл вносятся изменения с целью его настройки и оптимизации;  ·   осуществляется сборка программного продукта из исходных модулей.  Результатом работы является специальный файл, описывающий оптимальный алгоритм сборки программного продукта, построенный на основе анализа дерева зависимостей между исходными модулями. |
| PVCS Notify | Обеспечивает автоматическую рассылку сообщений об ошибках из базы данных пакета PVCS Tracker по рабочим станциям назначения. При этом используется офисная система электронной почты cc:Mail или Microsoft Mail. Расширяет возможности PVCS Tracker и используется только совместно с ним. Настраивается из среды PVCS Tracker. Настройка включает в себя определение интервала времени, через который PVCS Notify проверяет содержимое базы данных, определение критериев отбора записей для рассылки уведомлений, определение списков адресов для рассылки. После настройки PVCS Notify начинает работу в автономном режиме, автоматически рассылая уведомления об изменениях в базе данных PVCS Tracker.  Предназначен для использования в больших рабочих группах, часть членов которых хотя и доступна только через средства электронной почты, однако должна иметь оперативную информацию о требованиях на изменение проекта, замечаниях, ошибках, ходе и результатах его тестирования.  Результатом работы являются оформленные в соответствии с одним из стандартов почтовые сообщения, готовые для рассылки посредством e-mail. |

**6.2.6.2 Средства документирования**

Для создания документации в процессе разработки АС используются разнообразные средства формирования отчетов, а также компоненты издательских систем. Обычно средства документирования встроены в конкретные CASE-средства. Исключением являются некоторые пакеты, предоставляющие дополнительный сервис при документировании. Из них наиболее активно используется SoDA (Software Document Аutomation).

Продукт SoDA предназначен для автоматизации разработки проектной документации на всех фазах ЖЦ АС. Он позволяет автоматически извлекать разнообразную информацию, получаемую на разных стадиях разработки проекта, и включать ее в выходные документы. При этом контролируется соответствие документации проекту, взаимосвязь документов, обеспечивается их своевременное обновление. Результирующая документация автоматически формируется из множества источников, число которых не ограничено. Краткая характеристика SoDA дана в табл. 6.23.

Таблица 6.23

Краткая характеристика средства документирования SoDA

|  |  |
| --- | --- |
| Объект | Описание |
| База реализации | SoDA реализована на базе издательской системы FrameBuilder и предоставляет полный набор средств по редактированию и верстке выпускаемой документации. Разные версии документации могут быть для наглядности отмечены своими отличительными признаками. В системе создаются таблицы требований к проекту, по которым можно проследить, как реализуются эти требования. Разные виды документации, сопровождающие различные этапы ЖЦ, связаны между собой, и можно проследить состояние проекта от первоначальных требований до анализа, проектирования, кодирования и тестирования программного продукта. |
| Связь с приложе-ниями | SoDA не зависит от применяемых инструментальных средств. Связь с приложениями осуществляется через стандартный программный интерфейс API. Переход на новые инструментальные средства не влечет за собой дополнительных затрат по документированию проекта. |
| Набор шаблонов | SoDA содержит набор шаблонов документов, определяемых стандартом на программное обеспечение DOD 2167A. На их основе можно без специального программирования создавать новые формы документов, определяемые пользователями. |
| Графиче-ский редактор | Пакет включает в себя графический редактор для подготовки шаблонов документов. Он позволяет задавать необходимый стиль, фон, шрифт, определять расположение заголовков, резервировать места, где будет размещаться извлекаемая из разнообразных источников информация. Изменения автоматически вносятся только в те части документации, на которые они повлияли в программе. Это сокращает время подготовки документации за счет отказа от перегенерации всей документации. |
| Результат работы | Итоговым результатом работы системы SoDA является готовый документ (или книга). Документ может храниться в файле формата SoDA (Frame Builder), который получается в результате генерации документа. Вывод на печать этого документа (или его части) возможен из системы SoDA. |

**6.2.6.3 Средства тестирования**

Под тестированием понимается процесс исполнения программы с целью обнаружения ошибок. Регрессионное тестирование - это тестирование, проводимое после усовершенствования функций программы или внесения в нее изменений.

Одно из наиболее развитых средств тестирования QA (новое название - Quality Works) представляет собой интегрированную, многоплатформенную среду для разработки автоматизированных тестов любого уровня, включая тесты регрессии для приложений с графическим интерфейсом пользователя.

Краткая характеристика QA дана в табл. 6.24.

Таблица 6.24

Краткая характеристика средства тестирования QA

|  |  |
| --- | --- |
| Объект | Описание |
| Процесс тестирова-ния | QA позволяет начинать тестирование на любой фазе ЖЦ, планировать и управлять процессом тестирования, отображать изменения в приложении и повторно использовать тесты для более, чем 25 различных платформ. Одним из атрибутов теста является имя его разработчика, что позволяет при необходимости выполнять тесты, созданные конкретным разработчиком. |
| Основные компоненты | ·   QA Partner - среда для разработки, компиляции и выполнения тестов;  ·   QA Planner - модуль для разработки планов тестирования и обработки результатов. Для создания и выполнения тестов в процессе работы QA Planner вызывается QA Partner;  ·   Agent - модуль, поддерживающий работу в сети. |
| Этапы тестирова-ния | ·   создание плана тестирования;  ·   связывание плана с тестами;  ·   пометка и выполнение тестов;  ·   получение отчетов о тестировании и управление результатами.  Создание тестового плана в включает в себя составление схемы тестовых требований и выделение уровней детализации. Для этого необходимо определить все, что должно быть протестировано, подготовить функциональную декомпозицию приложения, оценить, сколько тестов необходимо для каждой функции и характеристики, определить, сколько из них будет реализовано в зависимости от доступных ресурсов и времени. Эта информация используется для создания схемы тестовых требований.  Для связывания плана с тестами необходимо создать управляющие предложения (скрипты) на специальном языке 4Test и тесты, которые выполняют требования плана, и связать компоненты любым способом. Во избежание перегруженности тестов используют управление тестовыми данными.  При выполнении плана результаты записываются в формате, похожем на план. Все результаты связаны с планом. Есть возможность просмотреть или скрыть общую информацию о выполнении, слить файлы результатов, разметить неудавшиеся тесты, сравнить результаты предыдущего выполнения тестов, выполнить или отменить отчет. |

**6.2.6.4 Проблема рационального выбора CASE-средств**

Спектр существующих в настоящее время CASE-средств поддержки процессов анализа и проектирования АС очень велик и постоянно расширяется. В связи с этим на практике приходится решать проблему рационального выбора CASE-средств для использования в работе. Данная проблема является сложной и трудно формализуемой.

Модель процесса оценки и выбора представлена на рис. 6.14. Она описывает наиболее общую ситуацию оценки и выбора, а также показывает зависимость между ними. Как можно видеть, оценка и выбор могут выполняться независимо друг от друга или вместе, каждый из этих процессов требует применения определенных критериев.

Рис. 6.14 Модель процесса оценки и выбора

Наиболее общие характеристики единого процесса оценки и выбора представлены в табл. 6.25. Основные характеристики собственно процесса оценки и процесса выбора представлены в табл. 6.26 - 6.27, соответственно.

Таблица 6.25

Общие характеристики процесса оценки и выбора CASE-средств

|  |  |
| --- | --- |
| Объект | Описание |
| Цель | Процесс оценки и выбора может преследовать несколько целей, включая одну или более из следующих:   * оценка нескольких CASE-средств и выбор одного или более из них; * оценка одного или более CASE-средств и сохранение результатов для последующего использования; * выбор одного или более CASE-средств с использованием результатов предыдущих оценок. |
| Элементы процесса | ·        цели, предположения и ограничения, которые могут уточняться в ходе процесса;  ·        потребности пользователей, отражающие количественные и качественные требования пользователей к CASE-средствам;  ·        критерии, определяющие набор параметров, в соответствии с которыми производится оценка и принятие решения о выборе;  ·        формализованные результаты оценок одного или более средств;  ·        рекомендуемое решение (обычно либо решение о выборе, либо дальнейшая оценка). |
| Условия начала процесса | Процесс оценки и/или выбора может быть начат только тогда, когда лицо, группа или организация полностью определила для себя конкретные потребности и формализовала их в виде количественных и качественных требований в заданной предметной области. Термин "пользовательские требования" далее означает именно такие формализованные требования.  Пользователь должен определить конкретный порядок действий и принятия решений с любыми необходимыми итерациями. Например, процесс может быть представлен в виде дерева решений с его последовательным обходом и выбором подмножеств кандидатов для более детальной оценки. Описание последовательности действий должно определять поток данных между ними. |
| Выбор критериев | Определение списка критериев основано на пользовательских требованиях и включает:   * выбор критериев для использования из приведенного далее перечня; * определение дополнительных критериев; * определение области использования каждого критерия (оценка, выбор или оба процесса); * определение одной или более метрик для каждого критерия оценки; * назначение веса каждому критерию при выборе. |

Таблица 6.26

Характеристики процесса оценки CASE-средств

|  |  |
| --- | --- |
| Объект | Описание |
| Цель оценки | Определение функциональности и качества CASE-средств для последующего выбора. Оценка выполняется в соответствии с конкретными критериями, ее результаты включают как объективные, так и субъективные данные по каждому средству. |
| Действия для оценки | ·        формулировка задачи оценки, включая информацию о цели и масштабах оценки;  ·        определение критериев оценки, вытекающее из определения задачи;  ·        определение средств-кандидатов путем просмотра списка кандидатов и анализа информации о конкретных средствах;  ·        оценка средств-кандидатов в контексте выбранных критериев. Необходимые для этого данные могут быть получены путем анализа самих средств и их документации, опроса пользователей, работы с демо-версиями, выполнения тестовых примеров, экспериментального применения средств и анализа результатов предшествующих оценок;  ·        подготовка отчета по результатам оценки. |
| Способы оценки | Оценка и накопление соответствующих данных может выполняться следующими способами:   * анализ CASE-средств и документации поставщика; * опрос реальных пользователей; * анализ результатов проектов, использовавших данные CASE-средства; * просмотр демонстраций и опрос демонстраторов; * выполнение тестовых примеров; * применение CASE-средств в пилотных проектах; * анализ любых доступных результатов предыдущих оценок. |
| Требования к процессу и результатам оценки | Результаты оценки в соответствии с конкретным критерием могут быть двоичными, находиться в некотором числовом диапазоне, представлять собой просто числовое значение или иметь какую-либо другую форму.  Для объективных критериев оценка должна выполняться путем воспроизводимой процедуры, чтобы любой другой специалист, выполняющий оценку, мог получить такие же результаты. Если используются тестовые примеры, их набор должен быть заранее определен, унифицирован и документирован.  По субъективным критериям CASE-средство должно оцениваться группой специалистов, использующих одни и те же критерии. Необходимый уровень опыта специалистов или групп должен быть заранее определен.  Результаты оценки должны быть стандартным образом документированы (для облегчения последующего использования) и, при необходимости, утверждены |

Таблица 6.27

Характеристики процесса выбора CASE-средств

|  |  |
| --- | --- |
| Объект | Описание |
| Действия для выбора | Процесс выбора тесно взаимосвязан с процессом оценки и включает следующие действия:   * формулировка задач выбора, включая цели, предположения и ограничения; * выполнение всех необходимых действий по выбору, включая определение и ранжирование критериев, определение средств-кандидатов, сбор необходимых данных и применение ранжированных критериев к результатам оценки для определения средств с наилучшими показателями. Для многих пользователей важным критерием выбора является интегрируемость CASE-средства с существующей средой; * выполнение необходимого количества итераций с тем, чтобы выбрать (или отвергнуть) средства, имеющие сходные показатели; * подготовка отчета по результатам выбора. |
| Типовой набор критериев для выбора | 1.     Поддержка полного жизненного цикла АС с обеспечением эволюционности ее развития  2.     Обеспечение целостности проекта и контроля за его состоянием  3.     Независимость от программно-аппаратной платформы и СУБД  4.     Поддержка одновременной работы групп разработчиков  5.     Возможность разработки приложений "клиент-сервер" требуемой конфигурации  6.     Открытая архитектура и возможности экспорта/импорта  7.     Качество технической поддержки продавцом-производителем, стоимость приобретения и поддержки, опыт успешного использования  8.     Простота освоения и использования  9.     Обеспечение качества проектной документации  10.  Использование общепринятых, стандартных нотаций и соглашений |
| Возможные результаты | В процессе выбора возможно получение двух результатов:   * рекомендаций по выбору конкретного CASE-средства; * запроса на получение дополнительной информации к процессу оценки. |
| Использу-емые алгоритмы | Алгоритмы, обычно используемые для выбора, могут быть основаны на масштабе или ранге. Алгоритмы, основанные на масштабе, вычисляют единственное значение для каждого CASE-средства путем умножения веса каждого критерия на его значение (с учетом масштаба) и сложения всех произведений. CASE-средство с наивысшим результатом получает первый ранг. Алгоритмы, основанные на ранге, используют ранжирование CASE-средств - кандидатов по отдельным критериям или группам критериев в соответствии со значениями критериев в заданном масштабе. Затем, аналогично предыдущему, ранги сводятся вместе и вычисляются общие значения рангов. |
| Анализ результатов выбора | При анализе результатов выбора предполагается, что процесс выбора завершен, CASE-средство выбрано и рекомендовано к использованию. Тем не менее, может потребоваться более точный анализ для определения степени зависимости значений ключевых критериев от различий в значениях характеристик CASE-средств - кандидатов. Такой анализ позволит определить, насколько результат ранжирования CASE-средств зависит от оптимальности выбора весовых коэффициентов критериев. Он также может использоваться для определения существенных различий между CASE-средствами с очень близкими значениями критериев или рангами. Если ни одно из CASE-средств не удовлетворяет минимальным критериям, выбор (возможно, вместе с оценкой) может быть повторен для других CASE-средств - кандидатов.  Если различия между самыми предпочтительными кандидатами несущественны, дополнительная информация может быть получена путем повторного выбора (возможно, вместе с оценкой) с использованием дополнительных или других критериев.  Рекомендации по выбору должны быть строго обоснованы. В случае отсутствия адекватных CASE-средств рекомендуется разработать новое CASE-средство, модифицировать существующее или отказаться от внедрения. |

Было установлено, что нецелесообразно сравнивать отдельно взятые CASE-средства, поскольку ни одно из них не решает в целом все проблемы создания и сопровождения АС и их ПО. Сравниваться могут комплексы методологически и технологически согласованных инструментальных средств, поддерживающие полный ЖЦ АС и обеспеченные необходимой технической и методической поддержкой со стороны фирм-поставщиков.

Примером популярного в СНГ и в мире комплекса такого рода является комплекс CASE-средств, основанный на методологии DATARUN и представленный в табл. 6.28.

Таблица 6.28

Состав комплекса CASE-средств, основанного на методологии DATARUN

|  |  |
| --- | --- |
| № | Описание |
| 1 | CASE-средство Silverrun |
| 2 | Средство разработки приложений JAM |
| 3 | Мост Silverrun-RDM <-> JAM |
| 4 | Комплекс средств тестирования QA |
| 5 | Менеджер транзакций Tuxedo |
| 6 | Комплекс средств планирования и управления проектом SE Companion |
| 7 | Комплекс средств конфигурационного управления PVCS |
| 8 | Объектно-ориентированное CASE-средство Rational Rose |
| 9 | Средство документирования SoDA |

[6 CASE-СРЕДСТВА ПРОЕКТИРОВАНИЯ АС (CASE)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541659)

[*6.1 Общие положения по CASE*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541660)

[6.1.1 Эволюция CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541661)

[6.1.2 Состав, структура и функциональные особенности CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541662)

[6.1.3 Поддержка графических моделей в CASE-средствах](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541663)

[6.1.4 Контроль ошибок в CASE-средствах](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541664)

[6.1.5 Организация и поддержка репозитория](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541665)

[6.1.6 Поддержка процесса проектирования и разработки в CASE](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541666)

[*6.2 Характеристика отдельных CASE-средств и их комплексов*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541667)

[6.2.1 Обзор CASE-средств Silverrun+JAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541668)

[6.2.1.1 Silverrun](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541669)

[6.2.1.2 JAM](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541670)

[6.2.2 Обзор CASE-средств Designer/2000 + Developer/2000](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541671)

[6.2.3 Обзор CASE-средств ERwin + BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541672)

[6.2.3.1 ERwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541673)

[6.2.3.2 BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541674)

[6.2.3.3 Функционально-стоимостный анализ в BPwin](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541675)

[6.2.3.4 Свойства, определяемые пользователем (UDP)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541676)

[6.2.4 Обзор CASE-средства Rational Rose](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541677)

[6.2.5 Обзор CASE-средства Sparx Enterprise Architect (SEA)](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541678)

[6.2.6 Обзор вспомогательных средств поддержки жизненного цикла автоматизированных систем](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541679)

[6.2.6.1 Средства конфигурационного управления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541680)

[6.2.6.2 Средства документирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541681)

[6.2.6.3 Средства тестирования](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541682)

[6.2.6.4 Проблема рационального выбора CASE-средств](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk6\6%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc466541683)

[7  ОРГАНИЗАЦИОННО-ЭКОНОМИЧЕСКИЕ И СОЦИАЛЬНО-ПСИХОЛОГИЧЕСКИЕ АСПЕКТЫ ВНЕДРЕНИЯ АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876441)

[*7.1 COBIT - стандарт организации и управления информационными технологиями*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876442)

[7.1.1   Общая характеристика стандарта COBIT](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876443)

[7.1.2 Уровни зрелости системы управления информационными технологиями в организации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876444)

[7.1.3 Процесс внедрения COBIT в деятельность организации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876445)

[*7.2 Экономические и социально-психологические аспекты создания АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876446)

[7.2.1 Факторы повышения эффективности управления в условиях автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876447)

[7.2.2 Психологический барьер и пути его преодоления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876448)

[7.2.3 Требования к руководителю ИТ-службы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876449)

# 7  ОРГАНИЗАЦИОННО-ЭКОНОМИЧЕСКИЕ И СОЦИАЛЬНО-ПСИХОЛОГИЧЕСКИЕ АСПЕКТЫ ВНЕДРЕНИЯ АС

## 7.1 COBIT - стандарт организации и управления информационными технологиями

### 7.1.1   Общая характеристика стандарта COBIT

Построение грамотной структуры управления организацией, создание эффективной вертикали принятия решений и системы контроля напрямую зависят от состояния информационных технологий (ИТ), от их эффективности, производительности, безопасности, надежности и других не менее важных показателей. Эффективное управление ИТ (включающее их внедрение, сопровождение, оценку состояния и развитие) — важнейший фактор для успеха каждой современной организации. Для упорядочения процессов решения соответствующих задач управления ИТ разработан международный стандарт COBIT.

Аббревиатура COBIT (от англ. Control OBjectives for Information and related Technology) расшифровывается как Контрольные ОБъекты для Информационных и смежных Технологий. За этой аббревиатурой скрывается набор документов, в которых изложены принципы управления и аудита информационных технологий. COBIT позиционируют как международный открытый стандарт "де-факто", определяющий набор универсальных задач управления ИТ, ориентированных, прежде всего, на руководство организации и на ИТ-аудиторов. Основная ценность COBIT заключается в том, что он предлагает модель, обеспечивающую взаимосвязь между бизнес-целями и ИТ-процессами.

В состав третьей редакции COBIT входит несколько книг, общее описание которых дано в табл. 7.1.

Таблица 7.1

Общее описание книг третьей редакции COBIT

|  |  |
| --- | --- |
| Книга | Описание |
| Резюме для руководства | Служит введением в остальные разделы стандарта. |
| Концептуаль-ное ядро COBIT | Представляет собой набор основополагающих принципов и понятий, а также модель управления ИТ, на базе которых строятся все положения стандарта. Концептуальное ядро COBIT сформировано из набора 34 высокоуровневых задач управления (одна задача для каждого ИТ-процесса), сгруппированных в четыре домена: планирование и организация; комплектование и внедрение; предоставление и поддержка; мониторинг. |
| Детальные задачи управления | Для каждого из ИТ-процессов, описанных в концептуальном ядре, определяется набор «Детальных задач управления»; всего их насчитывается 318. Каждая задача управления содержит формулировку ожидаемых результатов, которые необходимо достигнуть путем реализации конкретных процедур управления в рамках ИТ-процесса. |
| Руководство по менеджменту | Позволяет руководству предприятия реализовать более эффективные стратегии управления ИТ, установить контроль над использованием информационных ресурсов и соответствующими процессами, осуществлять мониторинг, давать сравнительную оценку достижениям бизнес-целей и оценивать производительность в рамках каждого ИТ-процесса |
| Руководство по аудиту« | Позволяет производить проверку реализации каждого из 34 высокоуровневых ИТ-процессов на предмет достижения каждой из 318 детальных задач управления, что дает возможность аудитору гарантировать руководству предприятия адекватность реализованной системы управления ИТ и формировать рекомендации по ее улучшению. |
| Набор инструментов внедрения | Дает разъяснения ключевых концепций, пошаговое описание и примеры успешного внедрения COBIT в организациях по всему миру. |

COBIT является инструментом, позволяющим руководству предприятия обеспечить переход от постановки бизнес-задач к вопросам управления ИТ, помогая установить должный уровень понимания рисков и преимуществ, связанных с использованием ИТ, а также реализовать эффективную систему управления ИТ, направленную на достижение бизнес-целей предприятия.

Миссия COBIT состоит в исследовании, разработке, рекламе и продвижении международного набора авторитетных, отвечающих современным требованиям, общепризнанных задач управления ИТ для повседневного использования бизнес-менеджерами и аудиторами.

Основной методологический принцип COBIT состоит в следующем: «Для своевременного и полного получения информации, необходимой организации для достижения бизнес-целей, управление ИТ-ресурсами должно осуществляться при помощи набора естественным образом сгруппированных процессов».

### 7.1.2 Уровни зрелости системы управления информационными технологиями в организации

Определяемые в COBIT модели зрелости организации (Maturity Model) позволяют руководству организации дать оценку текущему состоянию ИТ-процессов в сравнении с лучшими примерами в данной отрасли и найти возможности их совершенствования.

В целом в «незрелых» организациях успешность ИТ-проектов полностью зависит от личности администратора-«универсала». Система управления и преемственность в деятельности ИТ-отдела практически отсутствуют. Вплоть до того, что с уходом «универсала» вообще все может перестать работать. С повышением уровня зрелости организации увеличивается роль системы управления и понижается роль администратора. Администраторы становятся взаимозаменяемыми, а наличие отточенных навыков в узкой предметной области и специальных знаний, начинает цениться выше универсальности. Такая система выгодна для организации, но не выгодна для администратора-«универсала». Этим объясняется то сопротивление со стороны ИТ-специалистов, с которым сталкиваются руководители ИТ-отделов, внедряющие систему управления ИТ.

Всего определяется шесть уровней зрелости системы управления ИТ-организации (табл. 7.2).

Таблица 7.2

Уровни зрелости системы управления ИТ-организации согласно COBIT

|  |  |
| --- | --- |
| Уровень | Описание |
| 0 - Нулевой  (самый низкий) | О зрелости речь вообще не идет, системы управления ИТ как таковой не существует, а необходимость ее создания не осознается. Наблюдается полное отсутствие управляемых ИТ-процессов. Все ключевые ИТ-роли выполняются "незаменимыми" сотрудниками, часто являющимися "универсалами", мастерами на все руки. Общая стратегия развития ИТ отсутствует. Часто действия "универсалов" между собой не согласованы. При необоснованно завышенных расходах, например, на обеспечение информационной безопасности в такой организации совокупный результат, как правило, близок к нулю. Если зависимость предприятия от ИТ большая, то в такую организацию инвесторам не выгодно вкладывать деньги. |
| 1- Начальный | Руководство начинает осознавать необходимость реализации комплексного подхода к управлению ИТ, что вызвано большой зависимостью этих организаций от ИТ и значительными расходами, которые не дают видимых результатов. На этом уровне еще не существует стандартизированных ИТ-процессов и преобладает фрагментарный подход к их реализации. Руководство только начинает задумываться о получении возврата инвестиций, сделанных в ИТ, не располагая, однако, методикой оценки их эффективности. Для решения каждой задачи ИТ-специалистами вырабатываются собственные подходы. Связь между бизнес-целями и деятельностью ИТ-отдела отсутствует. На этом уровне в настоящее время находятся многие отечественные предприятия, вкладывающие большие средства в развитие и поддержание работоспособности своих АС. |
| 2-й уровень зрелости | Уже существуют стандартизированные ИТ-процессы, однако они не документированы и пока не являются частью системы управления. Фактически они реализуются в виде стандартной практики отдельных ИТ-специалистов. На этом уровне необходимость планомерного внедрения системы управления ИТ уже ни у кого не вызывает сомнений, активно разрабатываются показатели эффективности ИТ-процессов, внедряются процессы планирования, мониторинга и предоставления ИТ-услуг, устанавливаются взаимосвязи между ИТ и бизнес-процессами, разрабатывается стратегия развития ИТ. Руководство организации принимает активное участие в формировании управляемых ИТ-процессов, для которых уже существуют базовые методы оценки эффективности. Однако сказывается недостаток опыта управления ИТ, используется ограниченное количество механизмов управления и показателей эффективности. |
| 3-й уровень зрелости | Если на предыдущих уровнях зрелости преобладает администратор-"универсал", то, *начиная с третьего уровня,* доминирующей становится роль системы управления. Здесь все процедуры стандартизированы и документированы, а сотрудники обучены их использованию. Деятельность ИТ-отдела регламентирована этими процедурами. Однако механизмы контроля качества выполнения процедур пока не работают, а сами процедуры далеки от совершенства, и об их оптимизации говорить не приходится. |
| 4-й уровень зрелости | характеризуется наличием системы контроля качества ИТ-процессов. Эта система осуществляет непрерывный мониторинг ИТ-процессов, устанавливает стандарты качества и контролирует соответствие ИТ-процессов данным стандартам. Наличие системы контроля качества позволяет выявлять неэффективно действующие механизмы управления ИТ-системой и постоянно работать над повышением их эффективности. Четвертый уровень - это уровень, на котором существуют управляемые ИТ-системы. |
| 5-й  высший уровень зрелости (оптимизи-руемый) | Система управления ИТ отличается от предыдущего по существу лишь более высоким уровнем оптимизации ИТ-процессов, которые являются управляемыми и измеряемыми. Информация о выполнении каждого ИТ-процесса фиксируется. ИТ являются эффективным инструментом бизнеса, а система управления ими - одной из составных частей системы управления организацией. |

### 7.1.3 Процесс внедрения COBIT в деятельность организации

Шаги процесса внедрения COBIT в деятельность организации представлены в табл. 7.3.

Таблица 7.3

Шаги процесса внедрения COBIT в деятельность организации

|  |  |
| --- | --- |
| Шаг | Описание |
| 1 | Определение бизнес-целей при помощи концептуального ядра |
| 2 | Выбор ИТ-процессов и механизмов управления с использованием высокоуровневых и детальных задач управления |
| 3 | Согласование программы действий с бизнес-планом |
| 4 | Оценка существующих процедур и результатов внедрения механизмов управления при помощи *"Руководства по аудиту"* |
| 5 | Оценка текущего статуса организации, идентификация критичных действий и измерение производительности в достижении целей организации при помощи *"Руководства по менеджменту"* |

Уроки внедрения COBIT на предприятиях по всему миру указывают на необходимость вовлечения высшего руководства в проект внедрения уже на ранней его стадии. Следует быть готовым дать разъяснения основных концепций COBIT, а также привести примеры успешных внедрений в других организациях.

## 7.2 Экономические и социально-психологические аспекты создания АС

### 7.2.1 Факторы повышения эффективности управления в условиях автоматизации

В табл. 7.4 представлены факторы повышения эффективности управления в условиях автоматизации для обычных АС и дополнительные факторы для больших интегрированных АС.

Таблица 7.4

Факторы повышения эффективности управления в условиях автоматизации

|  |  |
| --- | --- |
| Вид АС | Описание |
| Обычные АС | -  Оптимизации планирования и оперативности управления;  -  Более полному и своевременному использованию информации об объекте управления;  -  Повышению уровня аналитической и организационной работы аппарата управления, освобожденного от рутинной работы по составлению, учету и обработке массовых документов и др. |
| Интегриро-ванные АС | -  Повышением оперативности обмена информацией между элементами ИАСУ и, как следствие, повышение эффективности принятия управленческих решений и сокращение длительности цикла «исследование-производство»;  -  Повышением точности управления за счет устранения информационных нестыковок элементов ИАСУ;  -  Устранением промежуточных носителе информации и, как следствие, сокращение численности персонала, занятого подготовкой и обработкой информации;  -  Увеличением загрузки и производительности вычислительной техники |

Затраты на разработку и эксплуатацию АС велики, поэтому важно уметь рассчитать экономическую эффективность этих затрат.

*Основные показатели экономической эффективности АС в целом* следующие:

Годовой экономический эффект;

Годовой прирост прибыли;

Коэффициент экономической эффективности;

Срок окупаемости затрат на создание АС.

В итоге основным источником экономической эффективности ИАСУ является дополнительная прибыль, образующаяся за счет увеличения объема выпуска продукции, уменьшения ее себестоимости, уменьшения непроизводительных потерь и уменьшения платы за производственные фонды вследствие их сокращения до рациональных минимально необходимых объемов.

### 7.2.2 Психологический барьер и пути его преодоления

ИАСУ затрагивает интересы большинства работников предприятия, а не только управленческого персонала. Поэтому проектировщику ИАСУ необходимо уделять большое внимание социально-психологическим аспектам, тщательно продумывая, как будут реализованы проектные решения в действующей системе с учетом человеческого фактора. Это особенно важно в связи с возникновением в ходе автоматизации объектов трудностей психологического характера или ***психологических барьеров***. Различают психологические барьеры I и II рода (табл. 7.5).

Таблица 7.5

Виды психологических барьеров

|  |  |
| --- | --- |
| Вид | Описание |
| Барьер  I рода | Означает боязнь человеком потери рабочего места, сокращения или перевода на другую работу. |
| Барьер  II рода | Означает несоответствие экономических показателей и возможностей ИАСУ тем экономическим рамкам, в которых находится персонал объекта, охваченного автоматизацией (нежелание работать лучше и больше). |

Преодолением психологических барьеров должно заниматься как руководство, так и разработчики АС. Если не подходить профессионально к этому вопросу, то противоборство между персоналом и АС приобретает характер острого конфликта. В итоге можно гарантировать, что создать и внедрить АС не удастся. Пути преодоления психологического барьера описаны в табл. 7.6.

Таблица 7.6

Пути преодоления психологического барьера

|  |  |
| --- | --- |
| № пути | Описание |
| 1 | Нецелесообразно ставить перед разработчиками АС для действующей организации задачу значительного сокращения численности персонала организации. |
| 2 | При проектировании АС следует обеспечивать создание дружественной среды для пользователя с учетом требований инженерной психологии. |
| 3 | Важную роль в обеспечении благоприятного социального климата играет закладываемая на стадии проектирования надежность АС. |
| 4 | Устранение отчуждения работников сферы управления от технических средств АС. |
| 5 | Повышение гибкости и оперативности АС. |
| 6 | Обязательное соблюдение принципа первого руководителя при проектировании АС. |
| 7 | Внедрение АС должно сопровождаться общим и специальным обучением персонала предприятия. Цель ***общего обучения*** – моральная и психологическая подготовка коллектива к переходу к работе в условиях автоматизации, удовлетворение любопытства, защита от неизбежных  слухов и предположений. Общее обучение проводится в период подготовки к внедрению АС. Цель ***специального обучения*** состоит в обеспечении конкретной производственной деятельности в условиях функционирования АС. Подготовка достигает своей цели, если окончание обучения персонала совпадает с началом работы в новых условиях. |

### 7.2.3 Требования к руководителю ИТ-службы

Успех или неудача ИТ-директора определяется главным образом его навыками лидера и человеческими качествами.

ИТ-директор должен держать руку на пульсе операционной деятельности, так как в условиях жестких бюджетов необходимо постоянно помнить о пресловутом «поезде, движущемся по расписанию». Однако наиболее эффективные ИТ-руководители при этом еще и выстраивают устойчивые партнерские отношения со своими подчиненными, коллегами и внешними партнерами, быстро адаптируясь к складывающейся ситуации.

Чтобы достичь результата и успешно решить ключевые задачи, необходимо организовать эффективное управление ИТ-процессами. В то же время поставленных целей невозможно добиться без грамотного руководства людьми. Навыки общения становятся мощной движущей силой и помогают людям оправдывать возложенные на них ожидания и добиваться максимальной отдачи от ИТ.

Можно выделить пять поведенческих стилей и ключевых навыков, определяющих успех ИТ-директора (табл. 7.7).

Таблица 7.7

Требования к руководителю ИТ-службы

|  |  |
| --- | --- |
| № | Описание требования |
| 1 | Лидерские навыки: компетентность, решительность, сила воли, умение брать ответственность на себя. |
| 2 | Понимание того, что руководство — это коллективный процесс, что все вопросы решаются людьми, путем взаимодействия с ними и с их помощью. В процессе руководства нельзя опираться только на собственные мысли. В процессе руководства и при выборе оптимального решения ИТ-директор опирается не только на собственный «высочайший интеллект» и аналитические способности. |
| 3 | Психологические навыки: умение формировать правильные межличностные взаимоотношения. Эффективный ИТ-директор для успеха в работе значительную часть своего времени и сил тратит на управление взаимоотношениями с коллегами, внешними поставщиками и клиентами. Иногда он приобретает еще большее влияние, ослабляя контроль и открывая тем самым уязвимые места управления. Это помогает ему создавать прочные личные связи, позволяющие оказывать нужное воздействие на людей как внутри организации, так и за ее пределами. |
| 4 | Культура общения и умение влиять на других. Лучшие ИТ-директора понимают, что подчиненные внимательно наблюдают за их действиями. Руководитель всегда находится на виду. Демонстрируя ясность своих намерений, последовательность, непротиворечивость и нужную страсть, ИТ-директор доносит информацию таким образом, чтобы окружающие ее не только поняли, но и прочувствовали. При должном внимании и стимулировании большинство людей увеличивают производительность труда. Лучшие ИТ-директора создают атмосферу, объединяющую людей, помогающую им почувствовать значимость своего вклада в общее дело и понять, за счет чего предприятие может укрепить свои позиции. Необходимо формировать эмоциональный фон, который помогает в работе. |
| 5 | Выстраивание людей, а не систем: умение обучать и воспитывать. Совершенствуя людей, находящихся вокруг, ИТ-директора повышают их потенциал и создают условия для достижения высоких результатов. Воспитать следующее поколение лидеров — лучшее, что они могут сделать для организации, сохраняя при этом долгую память о себе. |

[7  ОРГАНИЗАЦИОННО-ЭКОНОМИЧЕСКИЕ И СОЦИАЛЬНО-ПСИХОЛОГИЧЕСКИЕ АСПЕКТЫ ВНЕДРЕНИЯ АС](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876472)

[*7.1 COBIT - стандарт организации и управления информационными технологиями*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876473)

[7.1.1   Общая характеристика стандарта COBIT](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876474)

[7.1.2 Уровни зрелости системы управления информационными технологиями в организации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876475)

[7.1.3 Процесс внедрения COBIT в деятельность организации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876476)

[*7.2 Экономические и социально-психологические аспекты создания АС*](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876477)

[7.2.1 Факторы повышения эффективности управления в условиях автоматизации](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876478)

[7.2.2 Психологический барьер и пути его преодоления](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876479)

[7.2.3 Требования к руководителю ИТ-службы](file:///C:\Users\ritaa\Downloads\Telegram%20Desktop\PAS\%D0%AD%D0%A0%D0%A3%D0%94%20%D0%9F%D0%90%D0%A1\%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F\content\lk7\7%20%D0%9F%D0%90%D0%A1_16%D0%B3%20%D1%87%D0%B8%D1%81%D1%82.htm#_Toc456876480)