МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

КУРГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

А.Г. КОКИН

ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Учебное пособие

Курган 2011

УДК 681.3.06(075.8)
К60

Рецензенты: кафедра прикладной информатики и экономики ГО ВПО Шадринского государственного педагогического института (зав.кафедрой - канд. физ.-мат. наук, доцент **В.Ю. Пирогов**); зав. кафедрой информатики и вычислительной техники Курганской государственной сельскохозяйственной академии канд. эконом. наук, доцент **А.Х. Голованова**.

Печатается по решению методического совета
Курганского государственного университета.

К60 Кокин А.Г. Технология разработки программного обеспечения: Учебное пособие. – Курган: Изд-во Курганского гос. ун-та, 2011. - 100 с.

В учебном пособии излагаются вопросы, связанные с использование CASE - технологий, средств и процессного подхода для разработки программного обеспечения бизнес-процессов на основе создания информационных систем. Уделяется внимание логистическому подходу при проектировании бизнес-процессов, а также имитационным потоковым моделям при исследовании бизнес-процессов.

Учебное пособие предназначено для студентов, обучающихся по специальности 230105 “Программное обеспечение вычислительной техники и автоматизированных систем”.

Рис. - 89, табл. - 11, библ. - 23 назв.

УДК 681.3.06(075.8)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

При проектировании программного обеспечения основным является ***наличие трех основных компонентов*** [1]***:***

- описание предметной области проектирования;

- исходные данные, методологии, методы, технологии, средства, условия;

- описание требуемого результата, модели, проекта, цели проектирования*.*

Первым ключевым компонентом является описание предметной области*. Возможна следующая классификация сведений о предметной области (рисунок 1).*

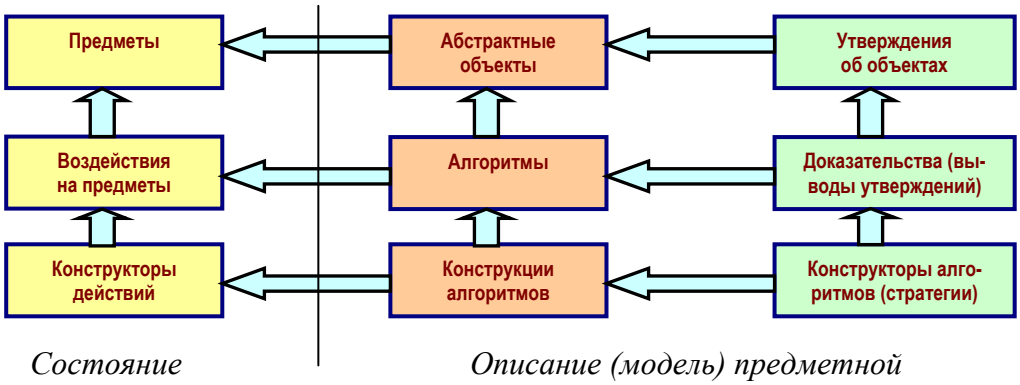


Рисунок 1 – Классификация сведений о предметной области

Левый столбик представляет состояние предметной области как части реального мира, а правый – знания о ней. Естественно, что реальные объекты, действия над ними (процессы) и способы комбинирования не могут быть формализованы.

Основу предметной области *составляют предметы, сущности, т.е. выделенные для исследования объекты внешнего мира (материальные или идеальные).* ***Остальные компоненты*** *состояния предметной области и ее модели характеризуются тем, какую роль по отношению к сущностям они выполняют.* ***Всякое воздействие вызывает изменение сущностей, т.е. превращение одной сущности в другую****. Исходная сущность называется объектом воздействия, созданная сущность – результатом. Когда объект и результат воздействия рассматриваются как разные состояния одной сущности, то говорят, что результат замещает объект.*

Сущность – это внутреннее содержание предмета, выражающееся в единстве всех его многообразных свойств и отношений.

Существенно, что из некоторых элементов или совокупностей элементов двух правых столбцов могут быть получены элементы, расположенные слева от них.

Например, *некоторым совокупностям утверждений соответствуют модели. Существуют выводы (доказательства), из которых может быть извлечен алгоритм, а алгоритму может быть сопоставлено некоторое воздействие на предметы.*

При хорошем уровне формализации возможно представление объектов одной категории объектами другой – кодирование.

Второй компонент проектирования *– собственно условие. Описание того, что дано:* ***методологии, методы, технологии и средства****. По своей природе этот компонент не отличается от рассмотренного состояния предметной области и ее описания. Состояние предметной области, входящее во второй компонент называют актуальным (имеющим место, начальным, заданным) состоянием.* ***Выделение актуального состояния*** *в отдельную компоненту связано с тем, что предметная область и наши знания о ней существуют независимо от конкретного проекта и являются общими для целого класса задач, актуальное состояние предметной области и его описание связаны с конкретной задачей и служат для уточнения и пополнения сведений о ней.*

Условие обычно включает некоторую совокупность предметов, заданных описанием и совокупность утверждений о них. *Утверждения могут носить характер фактов или утверждений общего характера – законов переметной области.*

Третий компонент – требование или цель проектирования *представляет собой некоторый императив. Он включает указание типа императива (спроектировать, построить), указание категории искомого элемента предметной области, его описание.*

Используя приведенную выше формализацию предметной области (рисунок 1), можно утверждать, что процесс проектирования состоит в конструировании элементов или предметов предметной области из заданных элементов по его описанию.

Таким образом, решение задачи представляется как движение влево и вверх от известных объектов, алгоритмов, методов и описаний к искомому результату*. При этом необходимо учитывать ресурсы, необходимые для создания проекта.*

АКТУАЛЬНОСТЬ ПОСТАНОВКИ ЗАДАЧИ

В современных условиях динамично развивается рынок комплексных интегрированных систем автоматизации предприятий и учреждений самого различного профиля (финансовых, промышленных) от малых предприятий до крупных корпораций. Такие системы предназначены для решения задач предприятия; управление финансовыми ресурсами, управление запасами, планирование и производство, сбыт и снабжение, техническое обслуживание и ремонт оборудования, управление персоналом и т.п.

Фактически ***проблема комплексной автоматизации*** стала актуальной для каждого предприятия. Уже не стоит вопрос “надо или не надо автоматизировать”, предприятия столкнулись с проблемой: каким образом это осуществить [2]. Подобная переориентация предприятий объясняется следующими ***основными причинами***:

- ***повышением степени*** организационной и финансовой самостоятельности;
- ***выходом*** на зарубежный рынок;
- ***завершением*** периода “островковой” автоматизации;
- ***возрастающей*** ориентацией предприятий на бизнес-процессы, т.е. деятельности, имеющие ценность для клиента;
- ***появлением*** на рынке как зарубежных, так и отечественных систем автоматизации.

Главная особенность систем автоматизации различных предприятий и учреждений, характеризующихся широкой номенклатурой входных данных с различными маршрутами их обработки, состоит в концентрации сложности на начальных этапах анализа требований и проектирования спецификаций системы. Фактически здесь и приходит понимание того, что будет делать будущая система и каким образом.

Следует отметить, что для большинства предприятий необходим и предваряющий автоматизацию этап - наведение порядка в их деятельности, создание рациональных технологий и бизнес-процессов.

Самостоятельно с задачей выбора и разработки собственной системы предприятие справиться не в состоянии, так как отсутствует концепция автоматизации. Возникает необходимость в услугах независимых консалтинговых фирм.

Консалтинг - это деятельность специалиста или целой фирмы, занимающихся стратегическим планированием проекта, анализом и формализацией требований к информационной системе, созданием системного проекта, иногда - проектированием приложений. Консалтинг предваряет и регламентирует названные этапы.

Фактически консультантом выполняется два вида работ. ***Прежде всего, это элементарное наведение порядка в организации***: бизнес-анализ и реструктуризация (реинжиниринг бизнес-процессов). Это направление получило название “бизнес-консалтинг”.

Другой вид работ - системный анализ и проектирование. Выявление и согласование требований заказчика приводит к пониманию того, что же 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.

Основными целями разработки консалтинговых проектов являются:

- представление деятельности предприятия и принятых в нем технологий в виде иерархии диаграмм, обеспечивающих наглядность и полноту их отображения;
- формирование на основании анализа предложений реорганизации организационно-управленческой структуры;
- упорядочивание информационных потоков внутри предприятия;
- выработка рекомендаций по построению рациональных технологий работы подразделений предприятия и его взаимодействию с внешним миром;
- анализ требований и проектирование спецификаций информационных систем;
- рекомендации и предложения по применимости и внедрению существующих систем управления предприятиями (ERP - enterprise resource planning).

ЭТАПЫ РАЗРАБОТКИ

1 Анализ первичных требований и планирование работ

Его основными задачами являются: анализ первичных бизнес-требований, предварительная экономическая оценка проекта, построение план-графика выполнения работ.

2 Проведение обследования деятельности предприятия

В рамках данного этапа осуществляется: предварительное выявление требований, предъявляемых к будущей системе; определение оргштатной и топологической структур предприятия; определение перечня целевых задач (функций) предприятия; анализ распределения функций по подразделениям и сотрудникам; определение перечня применяемых на предприятии средств автоматизации.

В качестве исходной информации при проведении обследования служат: данные по оргштатной структуре предприятия; информация о принятых технологиях деятельности; стратегические цели и перспективы развития; результаты интервьюирования сотрудников; нормативно-справочная документация; опыт системных аналитиков в части наличия типовых решений.

3 Построение моделей деятельности предприятия

На данном этапе осуществляется обработка результатов обследования и построение моделей деятельности предприятия следующих двух видов:

- ***модели “как есть”***, представляющей собой "снимок" положения дел на предприятии на момент обследования и позволяющей понять, что делает и как функционирует данное предприятие с позиций системного анализа;
- ***модели “как должно быть”***, интегрирующей перспективные предложения руководства и сотрудников, экспертов и системных аналитиков и позволяющей сформировать новые рациональные технологии работы предприятия.

Переход от модели “как есть” к модели ”как должно быть” осуществляется на основе совершенствования существующих технологий - “легкий” реинжиниринг и радикального изменения технологий и бизнес-процессов - “жесткий” реинжиниринг**.**

4 Разработка системного проекта

На данном этапе (фазе анализа требований к системе)требования заказчика уточняются, формализуются и документируются. Фактически на этом этапе дается ответ на вопрос: "Что должна делать будущая система?"

На этом этапе определяются:

- архитектура системы, ее функции, внешние условия ее функционирования, распределение функций между аппаратной и программной частями;
- интерфейсы и распределение функций между человеком и системой;
- требования к программным и информационным компонентам системы, необходимые аппаратные ресурсы, требования к базе данных, физические характеристики;
- состав людей и работ, имеющих отношение к системе;
- ограничения в процессе разработки (сроки, имеющиеся ресурсы).

Системный проект строится на основе модели “как должно быть” и включает функциональную модель будущей системы, информационную модель, а также техническое задание на создание автоматизированной системы.

5 Разработка технического проекта

На данном этапе осуществляется проектирование системы. Фактически здесь дается ответ на вопрос: "Как (каким образом) будет строиться система, чтобы она удовлетворяла предъявленным к ней требованиям?"

Этот этап разделяется на два этапа:

- проектирование архитектуры системы, разработка структуры и интерфейсов ее компонент, согласование функций и технических требований к компонентам, определение информационных потоков и связей между ними и внешними объектами;
- детальное проектирование включает разработку спецификаций каждой компоненты, разработку требований к тестам и интеграции компонент, а также построение иерархии программных модулей.

6 Последующие этапы

Последующие этапы являются традиционными: по спецификациям технического проекта осуществляется программирование модулей, их тестирование, отладка и последующая комплексация в автоматизированные рабочие места и в систему в целом.

CASE – ТЕХНОЛОГИИ

За последнее десятилетие сформировалось новое направление в проектировании информационных систем - **CASE** (**C**omputer-**A**ided **S**oftware/**S**ystem **E**ngineering). CASE - технология представляет собой совокупность методологий анализа, проектирования, разработки и сопровождения сложных систем программного обеспечения (ПО), поддержанную комплексом взаимоувязанных средств автоматизации. CASE - это инструментарий для системных аналитиков, разработчиков и программистов, заменяющий им бумагу и карандаш на компьютер для автоматизации процесса проектирования и разработки ПО.

Несмотря на то, что структурные методологии зарождались как средства анализа и проектирования ПО, CASE-технологии успешно применяются для моделирования практически всех предметных областей, однако ***устойчивое положение они занимают в следующих областях:***

- ***бизнес-анализ*** (фактически, модели деятельности предприятий “как есть” и ”как должно быть” строятся с применением методов структурного системного анализа и поддерживающих их CASE-средств);
- ***системный анализ и проектирование*** (практически любая современная крупная программная система разрабатывается с применением CASE-технологий по крайней мере на этапах анализа и проектирования, что связано с большой сложностью данной проблематики и со стремлением повысить эффективность работ).

Большинство CASE-средств основано на парадигме методология/ метод/ нотация/ средство. ***Методология*** определяет руководящие указания для оценки и выбора проекта разрабатываемого ПО, шаги работы и их последовательность**.** ***Метод*** - это систематическая процедура или техника генерации описаний компонент ПО (например, проектирование потоков и структур данных). ***Нотации*** предназначены для описания структуры системы, элементов данных, этапов обработки и включают графы, диаграммы, таблицы, блок-схемы. ***Средства*** - инструментарий для поддержки и усиления методов.

ЖИЗНЕННЫЙ ЦИКЛ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Разработка программного обеспечения базируется на модели его функционирования. Средства описания модели служат тем языком, с помощью которого разработчик формулирует и обосновывает решения, принимаемые в процессе проектирования. Средства описания модели дают совокупность понятий, то концептуальное пространство, в рамках которого развивается процесс проектирования.

ЭТАПЫ ЖИЗНЕННОГО ЦИКЛА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Создание информационной системы (ИС) представляет собой процесс построения ряда согласованных моделей на всех этапах жизненного цикла (ЖЦ) ИС.

Методология проектирования информационных систем описывает процесс создания и сопровождения систем в виде жизненного цикла ИС, представляя его как некоторую последовательность стадий и выполняемых на них процессов. Для каждого этапа определяются состав и последовательность выполняемых работ, получаемые результаты, методы и средства, необходимые для выполнения работ. На каждом этапе ЖЦ создаются специфичные для него модели, отражающие его различные состояния, начиная с создания ИС и заканчивая моментом ее неиспользования.

Выделяются следующие основные этапы ЖЦ ИС:

- анализ требований,
- проектирование,
- кодирование (программирование),
- тестирование и отладка,
- эксплуатация и сопровождение.

АНАЛИЗ ТРЕБОВАНИЙ

Начальным этапом процесса создания ИС является моделирование бизнес-процессов, протекающих в исследуемой системе и реализующих ее цели и задачи. Здесь формулируются основные требования к ИС (рисунок 2). Это фундаментальное положение методологии обеспечивает объективность в выработке требований к проектированию системы. Из множества моделей формируются модели требований к программной системе (ПО) и информационному обеспечению (ИО). Затем формируется архитектура ПО и ИО, выделяются БД и отдельные приложения.

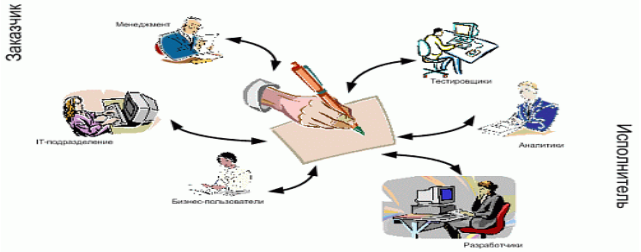


Рисунок 2 - Основные участники процесса разработки требований

Целью начальных этапов создания ИС на стадии анализа является формирование требований к ИС, корректно и точно отражающих цели и задачи организации-заказчика. Нужно выяснить и четко сформулировать, в чем заключаются эти потребности.

Для этого необходимо определить требования заказчиков к ИС и отобразить их на языке моделей в требования к разработке проекта ИС (рисунок 3). Требования должны быть собраны, проанализированы, структурированы, формализованы и представлены в виде законченного документа, который должен быть согласован и утвержден. Как со стороны тех, кто эти требования предъявляет (они выражают свое согласие с тем, что в документе представлено именно то, что им нужно), так и со стороны тех, кто будет разрабатывать систему (они подписываются под тем, что данные требования им понятны, реализуемы и достаточны для разработки системы).

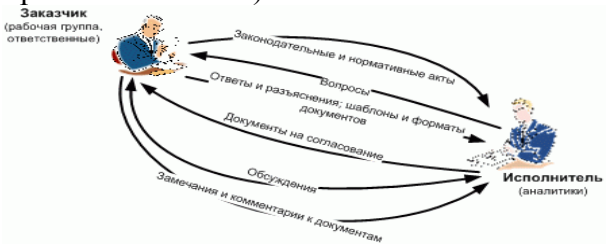


Рисунок 3 - Взаимодействие между исполнителем и заказчиком

Задача формирования требований к ИС является одной из наиболее ответственных, трудно формализуемых и наиболее дорогих и тяжелых для исправления в случае ошибки.

ПРОЕКТИРОВАНИЕ

Проектирование информационных систем всегда начинается с определения цели проекта. В общем виде ***цель проекта*** можно определить как ***решение ряда взаимосвязанных задач, включающих в себя обеспечение***: требуемой ***функциональности системы*** и уровня ее адаптивности к изменяющимся условиям функционирования; требуемой ***пропускной способности*** системы; требуемого ***времени реакции*** системы на запрос; ***безотказной работы*** системы; необходимого ***уровня безопасности***; ***простоты*** эксплуатации и поддержки системы.

Модели деятельности организации создаются в двух видах:

- модель "***как есть"*** отражает существующие на момент обследования в организации бизнес-процессы, положение дел в организации, и позволяющая понять, каким образом функционирует данная организация, а также выявить узкие места и сформулировать предложения по улучшению.

- модель "***как должно быть"*** отражает представление о новых технологиях работы организации, необходимые изменения бизнес-процессов с учетом внедрения ИС.

Каждая из моделей включает функциональную модель и модель данных деятельности организации, а также модель, описывающую динамику поведения организации.

Согласно современной методологии процесс создания ИС представляет собой процесс построения и последовательного преобразования ряда согласованных моделей на всех этапах жизненного цикла ИС. На каждом этапе создаются специфичные модели организации, требований к ИС, проект ИС, требований к приложениям и т.д. Модели формируются, сохраняются и накапливаются в репозитории проекта. Создание моделей осуществляется с использованием специальных программных инструментов - CASE-средств.

На этапе проектирования формируются модели данных. Проектировщики в качестве исходной информации получают результаты анализа. Построение логической и физической моделей данных является основной частью проектирования базы данных.

Параллельно выполняется проектирование процессов, чтобы получить спецификации (описания) всех модулей ИС. Оба эти процесса проектирования тесно связаны.

Главная цель проектирования процессов заключается в отображении функций в модули информационной системы. 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*** (на основании ER-модели, разработанной на этапе анализа);
- ***набор спецификаций модулей системы*** (они строятся на базе моделей функций).

Кроме того, на этапе проектирования осуществляется также разработка архитектуры ИС, включающая в себя выбор платформы (платформ) и операционной системы (операционных систем).

Характеристики архитектуры: архитектура "файл-сервер" или "клиент-сервер"; 2 - уровневая или 3-уровневая архитектура со следующими слоями: сервер, ПО промежуточного слоя (сервер приложений), клиентское ПО; база данных: централизованная или распределенная; база данных: однородная или неоднородная, то есть, будут ли все серверы баз данных продуктами одного и того же производителя (например, все серверы только Oracle или все серверы только DB2 UDB).

Этап проектирования завершается разработкой технического проекта ИС.

ЭТАП РЕАЛИЗАЦИИ

На этапе реализации осуществляется создание программного обеспечения системы, установка технических средств, разработка эксплуатационной документации.

МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА

Модель жизненного цикла отражает различные состояния системы, с момента возникновения необходимости в данной ИС и заканчивая моментом ее полного выхода из употребления. Существует международный стандарт ISO/IEC 12207, который под моделью ЖЦ представляет структуру, определяющую последовательность выполнения и взаимосвязи процессов, действий и задач, выполняемых на протяжении ЖЦ. К настоящему времени наибольшее распространение получили следующие ***основные модели ЖЦ.***

Каскадная модель (рисунок 4) предусматривает последовательное выполнение всех этапов проекта в строго фиксированном порядке. Переход на следующий этап означает полное завершение работ на предыдущем этапе.

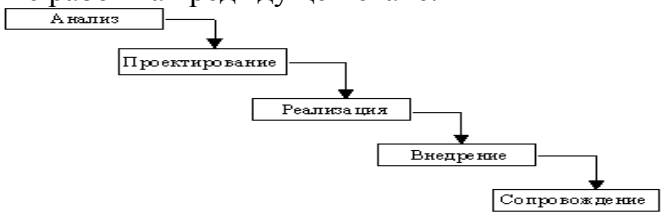


Рисунок 4 - Каскадная модель разработки ПО

Поэтапная модель с промежуточным контролем (рисунок 5). Разработка ИС ведется итерациями с циклами обратной связи между этапами. Межэтапные корректировки позволяют учитывать реально существующее влияние результатов разработки на различных этапах; время жизни каждого из этапов растягивается на весь период разработки.

Рисунок 5 – Поэтапная модель разработки ПО

Спиральная модель (рисунок 6). Основным недостатком каскадного подхода является существенное запаздывание с получением результатов. Пользователи могут внести свои замечания после того, как работа над системой будет полностью завершена.

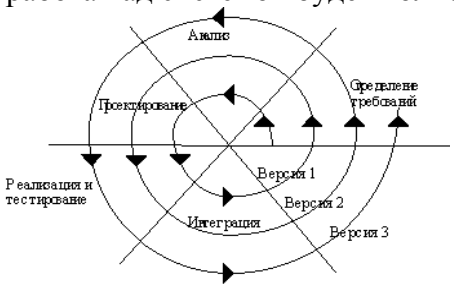


Рисунок 6 - Спиральная модель ЖЦ

Для преодоления перечисленных проблем была предложена спиральная модель ЖЦ, 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 ПО, уточняются цели и характеристики проекта, определяются его качество и планируются работы следующего витка спирали. Выбирается обоснованный вариант, который доводится до реализации. 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, составленным на основе статистических данных, полученных в предыдущих проектах.

ПРОЦЕССНЫЙ ПОДХОД

На основе практики ведения консалтинговых проектов был разработан методологический подход оказания консалтинговых услуг по построению процессной системы управления предприятием.

Под процессным подходом к организации и управлению деятельностью предприятия понимается ориентация деятельности предприятия на бизнес-процессы; системы управления предприятием на управление как каждым бизнес-процессом в отдельности, так и всеми бизнес-процессами в целом; системы качества предприятия на обеспечение качества технологий выполнения бизнес-процессов в рамках существующей или перспективной организационно-штатной структуры.

Под бизнес-процессом понимают совокупность различных видов деятельности предприятия, которые, вместе взятые, создают результат (продукт, услугу), имеющий ценность для потребителя, клиента или заказчика.

ХАРАКТЕРИСТИКА ПРОЦЕССНОГО ПОДХОДА

ПРЕИМУЩЕСТВА ПРОЦЕССНОГО ПОДХОДА

"Хотя люди, вовлеченные в процессы производства, мыслили в терминах таких процессов многие десятилетия, теория бизнес-процессов появилась менее двадцати лет назад и сначала была встречена практически с полным равнодушием. Даже те немногие, кто заинтересовался идеей, выражали скептицизм в отношении ее реальных достоинств, и только в связи с массовым внедрением реинжиниринга идея управления бизнес-процессами начала набирать обороты".

Чтобы "процесс пошел", надо, чтобы он был… Процесс – одна из возможных форм описания деятельности организации. Форма, введенная в действие международных стандартов ИСО 9000:2000, возникла не случайно, расширив наши возможности в понимании бизнеса. Теперь она создает ***горизонтальный срез функционирования организации***, привлекая наше внимание не только к структурным единицам и объектам организации, но и к их взаимодействию (рисунок 7).

Именно взаимодействия элементов часто порождают узкие места и проблемы, а процессы облегчают описание взаимодействий. В этом одна из причин их нынешней популярности. С этим связано и разнообразие форм представления и способов описания процессов и программных продуктов, предназначенных для автоматизации деятельности по описанию процессов.

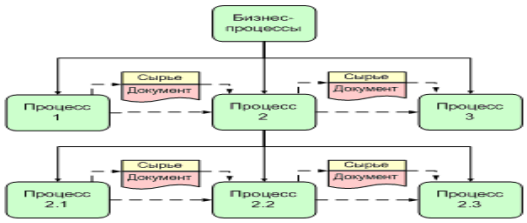


Рисунок 7 - Вертикальное и горизонтальное описание процессов

Стандарт ИСО 9000:2000 определяет ***процесс как: "совокупность взаимосвязанных и взаимодействующих действий, преобразующих входы в выходы". Определяющим является термин "взаимодействие".*** Действительно, фундаментальные взаимодействия бизнеса – это взаимодействия с заинтересованными сторонами - с клиентами и поставщиками, начиная с их выбора и одобрения, и вплоть до тесного взаимовыгодного общения. Эти процессы, в свою очередь, ведут к взаимодействию с участниками логистической системы и соответствующих логистических процессов. Процессы порождаются и взаимодействиями с сотрудниками, владельцами, партнерами, и всеми другими представителями заинтересованных сторон.

Это значит, что для разумной организации деятельности существенно выделить требования потребителя (цели, по которым оценивается результат деятельности), ресурсы (для получения которых понадобятся поставщики) и, собственно, последовательность операций, направленных на преобразование ресурсов в продукцию или услуги, соответствующие заявленным целям. Эта последовательность действий и представляет собой процесс, который допускает описание с различной степенью подробности. Можно говорить о процессе на уровне организации, на уровне структурного подразделения организации, на уровне команды и на уровне рабочего места.

Появление процессного подхода открыло дополнительные преимущества.

Первое преимущество связано с возникновением языка описания деятельности, доступного и понятного всем вовлеченным в процесс. Это позволило разрушить или, по крайней мере, снизить барьеры, обусловленные различиями в терминологии, в статусе внутри организации, в образовании и т.п.

Второе преимущество – возможность простой и наглядной графической интерпретации деятельности, что существенно облегчает, например, реализацию такого важного требования стандарта ИСО 9000:2000 как прослеживаемость. Графическое представление процессов создает простой и понятный язык описания, существенно облегчающий обмен информацией по вертикали и по горизонтали.

Третье преимущество вытекает из предыдущего и связано с ***выделением зон ответственности*** и рабочих зон, что облегчает формулировку требований на выполнение определенной работы, и облегчает задачу составления штатного расписания.

Четвертое преимущество обусловлено более ***простым и надежным определением точек контроля*** и критических точек в процессе за счет разбиения процесса на подпроцессы. ***Благодаря локализации точек контроля*** процессный подход помогает организации информационных потоков бизнес-процессов. ***Становится понятным***, где важно собирать информацию, с помощью каких средств измерения, и какими статистическими методами стоит пользоваться при ее свертке и представлении. А ***также*** проясняется организация механизмов обмена информации, ее накопления и хранения.

Процессный подход облегчает описание взаимодействий бизнес-процесса и вспомогательных процессов, прежде всего таких, как процессы обслуживания и ремонта. Упрощается и взаимодействие с процессами управления.

Управление качеством является одной из важных задач. При этом надо учитывать ***гибкость управления,*** чтобы при изменении структуры бизнес-процесса ***оставалась возможность перестройки системы учета, контроля анализа*** без значительных затрат.

ОБЩАЯ КЛАССИФИКАЦИЯ И УРОВНИ ОПИСАНИЯ ПРОЦЕССОВ

Стандарт ИСО 15504 связывает все процессы с жизненным циклом. В рамках жизненного цикла этот стандарт выделяет ***три группы процессов***.

Первую образуют основные процессы, которые включают процессы “поставщик-потребитель” и инженерные процессы (разработка и сопровождение).

Вспомогательные процессы – это вторая группа, а ***организационные процессы*** (организация и управление) – третья.

Необходимо выделить несколько ***уровней описания процессов***.

Стратегический уровень - верхний уровень, на котором организация рассматривается как "черный ящик", взаимодействующий с заинтересованными сторонами.

Основной критерий классификации процессов на этом уровне – "клиент" (заинтересованная сторона), ради которого этот процесс запускается. ***Условились называть процессы менеджмента процессами управления; процессы удовлетворения интересов сотрудников – социальными процессами; процессы удовлетворения потребностей потребителя – "бизнес-процессами".***

Второй классификационный критерий делит процессы на те, что направлены на создание ценностей для заинтересованных сторон, и те, что обеспечивают процессы создания ценностей, например, процессы управления и вспомогательные процессы.

Иерархический уровень – тактический – предполагает развертывание процессов, связанных с этапами жизненного цикла продукции. Для программных продуктов эти процессы международного стандарта ИСО/МЭК 12207.

Цикл начинается с маркетинга. Поставщики этого процесса находятся вне организации. К ним относятся структуры, которые перехватывают у маркетинга "эстафетную палочку" и несут ее дальше – в исследования, разработки и менеджмент.

На следующем этапе возникает процесс исследований и разработок. Технологические службы перехватывают эстафету на следующем этапе, порождая соответствующие процессы. А их, в свою очередь, уже ждут ***производственники***, главные исполнители собственно производственных процессов. Далее ***распространение товара или услуги*** (продажа), и это тоже процесс. За ним идет ***послепродажное обслуживание и, наконец, утилизация***. После чего все начинается сначала.

Оперативный уровень – выделяет процессы, которые определяют действия отдельного сотрудника на рабочем месте. Именно свойство "фрактальности" позволяет строить такие процессы. Из этого следует вложенность процессов один в другой и образование единой системы, ансамбля. ***Менеджмент нужен для того, чтобы управлять этим ансамблем процессов.***

БИЗНЕС – ПРОЦЕССЫ

В качестве предметной области для разработки программного обеспечения рассматриваются бизнес-предприятия*. Подразделения бизнес-предприятий выполняют определенные функции и связаны между собой и внешними сущностями потоками данных. Моделями такой предметной области являются информационные системы.*

ОПИСАНИЕ

Бизнес-процесс - это цепочка работ (операций, функций), результатом которой является какой-либо продукт или услуга*.*

Определение бизнес-процессов компании, их описание, анализ и оптимизация (реинжиниринг системы управления) в первую очередь преследуют цель повышения конкурентоспособности компании путём организации ее целесообразной, эффективной, слаженной деятельности. ***Процессно-ориентированный подход в управлении, в противовес функциональному подходу, является более сложным с точки зрения реализации, но в то же время более эффективным***. Поскольку специфика современного рынка такова, что конкурентное преимущество определяется не качеством продукции, а качеством бизнес-процессов, производящих продукцию.

Основой процессно-ориентированного подхода в управлении является понятие бизнес-процесса. Под этим термином понимается ряд взаимосвязанных видов деятельности, преобразующих входы в выходы, полезные потребителю. Бизнес-процессы существуют вне зависимости от того, какой подход к управлению существует в организации.

О бизнес-процессе следует знать, что он:

имеет свои границы. Границы бизнес-процесса определяются структурой компании, своим продуктом (выходом), цепочкой создания стоимости (процессом, создающим ценность продукта, - основным или вспомогательным);

имеет конечного потребителя (другой бизнес-процесс или конечный потребитель). Бизнес-процесс без потребителя не имеет смысла, поскольку всякая деятельность в компании выполняется ради производства чего-либо для кого-либо;

имеет своего владельца, отвечающего за его выполнение, за своевременное получение потребителем качественного выхода процесса.

ЭФФЕКТИВНОСТЬ БИЗНЕС-ПРОЦЕССОВ

Что же даёт внедрение процессного подхода в управлении?

Ориентация на результат. При процессном подходе в управлении становится очевидным, что конечный продукт бизнес-процесса не является простым "средним ариф-

метическим" работ всех сотрудников. Если один сотрудник работает хорошо, а второй плохо, то конечный продукт обоих сотрудников будет равен нулю. Компании не нужны "самородки", равно как и "бракоделы" - достаточно хорошо организовать сотрудников с удовлетворительным (средним) качеством работы, чтобы получить стабильный и конкурентоспособный продукт на выходе. ***Деятельность перестаёт выполняться ради самой деятельности - она становится деятельностью ради потребителя.***

Упрощение системы управления. При использовании процессного подхода, оргструктура становится более плоской за счет сокращения лишних потоков информации на верхние уровни управления. Если четко определено, кто за что отвечает – руководитель может отказаться от промежуточных звеньев в цепи без потери качества управления.

Стирание граней между подразделениями. При процессном подходе каждый точно знает, что, кому и когда должен передать в качестве выхода своего процесса. Это значительно уменьшает количество лишних информационных потоков.

Устранение барьеров в управленческой иерархии. Деятельность каждого сотрудника становится проще оценивать – по результатам деятельности бизнес-процессов, которые выражаются в сбалансированной системе показателей (ССП).

Таким образом, процессно-ориентированный подход в управлении - это управленческий инструмент, снижающий непроизводительные затраты, повышающий качество продукции, позволяющий иметь полную информацию о текущем процессе бизнеса и принимать своевременные и стратегически верные решения.

БИЗНЕС – МОДЕЛЬ КОМПАНИИ, ОРГАНИЗАЦИОННЫЙ АНАЛИЗ

Организационный анализ компании при инжиниринговом подходе проводится по определенной схеме с помощью ***полной бизнес-модели компании.*** Компания рассматривается как целевая, открытая социально-экономическая система, взаимодействующая с внешними системами (рынок, государственные учреждения) и состоящая из подсистем (отделы, цеха, бригады и пр.). Возможности компании определяются характеристиками ее структурных подразделений и организацией их взаимодействия. На рисунке 8 представлена обобщенная схема организационной бизнес-модели.

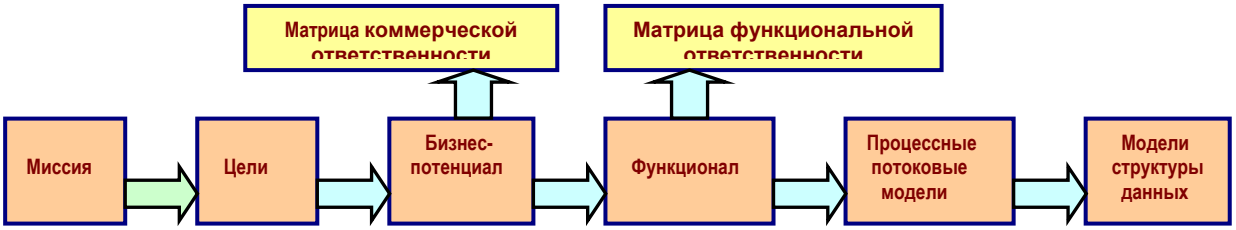


Рисунок 8 - Обобщенная схема организационного бизнес – моделирования

Построение ***бизнес-модели компании*** начинается с описания модели взаимодействия с внешней средой, то есть с определения ***миссии компании***.

Миссия, согласно ISO-15704, - это деятельность, осуществляемая предприятием для предоставления заказчикам продукта или услуги, с помощью которого предприятие реализует свои цели и задачи.

Миссия компании по удовлетворению социально-значимых потребностей рынка определяется как ***компромисс интересов рынка и компании***. При этом ***миссия*** разрабатывается, с одной стороны, исходя из рыночной конъюнктуры, а с другой - исходя из объективных внутренних возможностей компании. ***Миссия*** является своеобразной мерой устремлений компании и, в частности, определяет рыночные претензии компании. Определение ***миссии*** позволяет сформировать ***дерево целей компании***.

Дерево целей формирует ***дерево стратегий*** - иерархические списки достижения целей. При этом разрабатываются стратегии роста, интеграции и инвестиции бизнеса.

Блок бизнес-стратегий определяет продуктовые и конкурентные стратегии, а также стратегии сегментации и продвижения.

Ресурсные стратегии определяют стратегии привлечения материальных, финансовых, человеческих и информационных ресурсов.

Функциональные стратегии определяют стратегии в организации компонентов управления и этапов жизненного цикла продукции. Одновременно выясняется потребность и партнерские отношения (субподряд, сервисные услуги, продвижение и пр.). Это позволяет обеспечить заказчикам необходимый продукт требуемого качества, в нужном количестве, в нужном месте, в нужное время и по приемлемой цене.

Бизнес-потенциал компании. Стратегии дают возможность сформировать ***бизнес-потенциал компании*** - набор видов коммерческой деятельности, направленный на удовлетворение потребностей конкретных сегментов рынка. Исходя из специфики каналов сбыта, формируется представление об организационной структуре (определяются центры коммерческой ответственности).

Функционал компании. Бизнес-потенциал, в свою очередь, определяет ***функционал компании*** - перечень бизнес-функций (рисунок 9), функций менеджмента и функций обеспечения, требуемых для поддержания на регулярной основе указанных видов коммерческой деятельности. Кроме того, уточняются необходимые для этого ресурсы (материальные, человеческие, информационные) и структура компании.

Рисунок 9 - Шаблон формирования основных бизнес-функций

Построение ***бизнес-потенциала*** и ***функционала компании*** позволяет с помощью ***матриц проекций*** определить зоны ответственности менеджмента (рисунок 10). ***Матрица проекций*** - модель, представленная в виде матрицы, задающей систему отношений между классификаторами в любой их комбинации.

Рисунок 10 - Шаблон формирования основных функций менеджмента

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.

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

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***.

Процессные потоковые модели - это 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.

Модели структуры данных завершают организационное бизнес-моделирование разработкой ***модели структур данных,*** которая определяет перечень и форматы документов, сопровождающих процессы в компании, а также задает форматы описания объектов внешней среды, компонентов и регламентов самой компании**.**

Такой подход позволяет описать деятельность компании с помощью множества управленческих регистров: цели, стратегии, продукты, функции, организационные звенья.

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***. При этом происходит процессно-целевое описание компании, позволяющее получить взаимосвязанные ответы на вопросы: зачем-что-где-кто-как-когда-кому-сколько.

Таким образом, организационный анализ предполагает построение ***комплекса взаимосвязанных информационных моделей компании,*** который включает:

- ***стратегическую модель*** (отвечает на вопросы: зачем компания занимается именно этим бизнесом, почему предполагает быть конкурентоспособной, какие цели и стратегии для этого необходимо реализовать);
- ***организационно-функциональную модель*** (отвечает на вопрос: кто-что делает в компании и кто за что отвечает);
- ***функционально-технологическую модель*** (отвечает на вопрос: что-как реализуется в компании);
- ***процессно-потоковую 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 описания компании и позволяет вырабатывать понятные требования к проектируемой ИС.

CASE – МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ ИС

АРХИТЕКТУРА СОВРЕМЕННЫХ СИСТЕМ И МЕТОДОЛОГИЙ

Методологии, технологии и инструментальные средства проектирования CASE-средства составляют основу проекта любой информационной системы ИС.

Под методологией понимается регламентация процесса проектирования ИС и обеспечение управления этим процессом для выполнения требований к проектируемой ИС. *Методология реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые обеспечивают выполнение процессов ЖЦ.*

В центре любой методологии находится некоторая системная архитектура, и лишь затем совокупность стратегий и методов анализа и проектирования. ***Архитектура современных систем является трехслойной и имеет следующие характеристики*:**

- четко определенные слои,
- формальные и явные интерфейсы между слоями,

- скрытые и защищенные детали внутри каждого слоя.

Три слоя (база данных, правила бизнеса, документы) отражают возрастание уровня абстракции в рассматриваемой системной архитектуре.

Наиболее детальным слоем является база данных, более высокий уровень абстракции - слой правил бизнеса, наивысший уровень абстракции - слой документов. В данной архитектуре ***слой правил бизнеса*** является относительно новой концепцией. ***Процессы данного слоя отражают***:

- ***выполнение требуемых задач,***
- ***принятие решений в соответствующей компетенции,***
- ***запуск других задач в слое правил бизнеса и других слоях***.

Независимость слоев трехслойной системной архитектуры обеспечивает следующие основные преимущества:

- ***улучшение базы данных*** - отделение базы данных от изменений в технологиях, а следовательно, поддержка согласованности и осмысленности данных в течение длительного периода времени;
- ***гибкость интерфейсов пользователя*** - изменение интерфейсов без влияния на бизнес-процессы и наоборот;
- ***разделение усилий*** коллектива разработчиков.

Таким образом, в современном проекте основным центром является бизнес-процесс, база данных - менее важный из двух центров, т.е. процесс становится первичным и во многом определяет весь проект. Модель процесса является средством для размышлений и совместной работы над перспективами развития предприятия и системной разработкой. Тем не менее, информационная модель продолжает оставаться важной и соответствующим образом влиять на разрабатываемую функциональную модель.

В таблице 1 представлена трехслойная системная архитектура в разрезе регламентируемых этапов разработки (анализ требований, проектирование, реализация).

Таблица 1

Слои	Анализ	Проектирование	Реализация
Документы	Поток работ	Поток форм	Формы
Правила бизнеса	Поток процессов	Модель компонентов	Программы
База данных	Модель данных	Схема базы данных	Таблицы и т.п.

В настоящее время наибольшее распространение получили методологии SADT (Structured Analysis and Design Technique), структурного системного анализа Гейна-Сарсона, структурного анализа и проектирования Йордана Де Марко, структурного анализа модулей Джексона и Константайна, развития структурных систем Варнье-Орра, информационного моделирования Мартина и другие.

Структурные методологии предлагают методику трансляции проектных спецификаций в модель реализации, в дальнейшем используемую при кодогенерации. Кодогенерация предполагает наличие кодовых стандартов, специфицирующих формат заголовков подпрограмм, ступенчатый вид вложенных блоков, номенклатуру для спецификации переменных и имен подпрограмм и т.п.

МЕТОДОЛОГИЯ DFD

Классические стандарты DFD - Data Flow Diagram и WFD - Work Flow Diagram содержат набор символов или обозначений, с помощью которых описывается бизнес-процесс. Эти обозначения принято называть языком или методологией описания процессов. В данном случае этот язык или методология являются классическими.

В настоящее время в мире появилось много других языков или методологий описания бизнес-процессов, содержащих несколько иные обозначения. Причем каждая методология содержит свой язык и имеет свое название. ***Несмотря на различия, связанные с названием диаграмм и видов используемых объектов современные методологии описания бизнес-процессов практически идентичны и представляют незначительные видоизменения двух классических схем - DFD и WFD.***

Стандарт описания бизнес-процессов был разработан на основе развития классической методологии DFD. Он представлен нотациями Гейна-Сарсона и Йордана-Де Марко.

ДИАГРАММЫ ПОТОКОВ ДАННЫХ

Нотация Йордана Де Марко или Гейна-Сарсона обеспечивает анализ требований и функциональное проектирование информационных систем. Интеграция DFD-ERD осуществляется с использованием хранилища данных, структура которых описывается с помощью ERD и согласуется по соответствующим потокам на DFD. DFD могут быть легко преобразованы в модели проектирования - структурные карты. ***Наличие миниспецификаций DFD-процессов нижнего уровня позволяет преодолеть логическую незавершенность модели, когда дальнейшая ее детализация становится бессмысленной и построить полную функциональную спецификацию разрабатываемой системы для создания на ее основе программной системы.*** Для динамического моделирования в настоящий момент доступен ряд методологий и продуктов (INCOME Mobile, CPN-AMI и др.), базирующихся на сетях Петри различного вида и интегрируемых с DFD-моделью, которые позволяют успешно решать задачи бизнес-консалтинга.

В основе данной методологии (Gane/Sarson) лежит построение модели анализируемой ИС - проектируемой или реально существующей. В соответствии с методологией модель системы определяется как иерархия диаграмм потоков данных (DFD), описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю. ***Диаграммы верхних уровней иерархии*** (контекстные диаграммы) определяют основные процессы или подсистемы ИС с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором возможно проведение формализации процессов.

Источники информации (внешние сущности) порождают информационные потоки (потоки данных), переносящие информацию к подсистемам или процессам. Те в свою очередь преобразуют информацию и порождают новые потоки, которые переносят информацию к другим процессам или подсистемам, накопителям данных или внешним сущностям - потребителям информации. Таким образом, ***основными компонентами диаграмм потоков данных являются: внешние сущности***; ***системы/подсистемы***; ***процессы***; ***накопители данных***; ***потоки данных***.

Диаграммы потоков данных (DFD) являются основным средством моделирования функциональных требований проектируемой системы. С их помощью эти требования разбиваются на функциональные компоненты (процессы) и представляются в виде сети, связанной потоками данных. Главная цель - продемонстрировать, как каждый процесс преобразует свои входные данные в выходные и выявить отношения между этими процессами.

В качестве примера рассмотрим фрагмент функциональной модели компании, занимающейся распределением товаров по заказам (рисунок 11). Заказы подвергаются входному контролю и сортировке. Если заказ не отвечает номенклатуре товаров или оформлен

неправильно, то он аннулируется с уведомлением заказчика. Если заказ не аннулирован, то определяется, имеется ли на складе соответствующий товар.

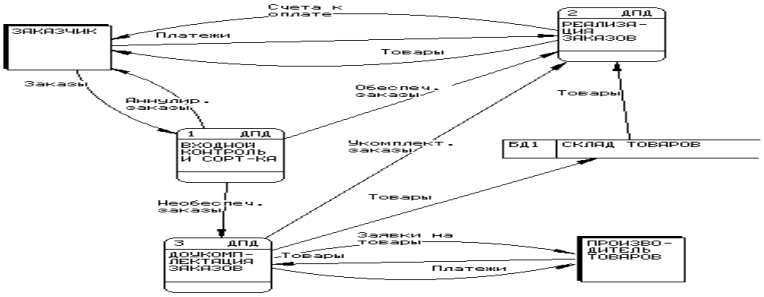


Рисунок 11 - Пример диаграммы Гейна-Сарсона

В случае положительного ответа выписывается счет к оплате и предъявляется заказчику. Если заказ не обеспечен складскими запасами, то отправляется заявка на товар производителю. После поступления требуемого товара на склад компании заказ становится обеспеченным и повторяет вышеописанный маршрут.

ОСНОВНЫЕ ПОНЯТИЯ

Основные символы DFD изображены на рисунке 11. Функциональные требования представляются с помощью процессов и хранилищ, связанных потоками данных.

ПОТОКИ ДАННЫХ являются механизмами, использующимися для моделирования передачи информации (или даже физических компонент) из одной части системы в другую. Потоки на диаграммах обычно изображаются именованными стрелками, ориентация которых указывает направление движения информации, например, ***ПЛАТЕЖИ, ТОВАРЫ***.

ПРОЦЕСС. Назначение ***ПРОЦЕССА*** состоит в продуцировании выходных потоков из входных в соответствии с действием, задаваемым именем процесса. Это имя должно содержать глагол в неопределенной форме, например, ***РЕАЛИЗАЦИЯ ЗАКАЗОВ, ДОУКОМПЛЕКТАЦИЯ ЗАКАЗОВ***. Кроме того, 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 (НАКОПИТЕЛЬ) ДАННЫХ позволяет на определенных участках определять данные, которые будут сохраняться в памяти между процессами. Фактически хранилище представляет "срезы" потоков данных во времени. Имя хранилища должно идентифицировать его содержимое и быть существительным, например, ***СКЛАД ТОВАРОВ***. В случае, когда поток данных входит или выходит в/из хранилища, и его структура соответствует структуре хранилища, он должен иметь то же самое имя, которое нет необходимости отражать на диаграмме.

ВНЕШНЯЯ СУЩНОСТЬ представляет сущность вне контекста системы, являющуюся источником или приемником системных данных. Ее имя должно содержать существительное, например, ***ЗАКАЗЧИК***.

МИНИСПЕЦИФИКАЦИИ ПРОЦЕССОВ описывают процессы нижнего уровня. Миниспецификации представляют собой алгоритмы описания задач, выполняемых процессами, множество всех миниспецификаций является полной спецификацией системы.

КОНТЕКСТНАЯ ДИАГРАММА И ДЕТАЛИЗАЦИЯ ПРОЦЕССОВ

Важную специфическую роль в модели играет специальный вид DFD - ***контекстная диаграмма,*** моделирующая систему наиболее общим образом. Контекстная диаграмма отражает интерфейс системы с внешним миром, а именно, информационные потоки между системой и внешними сущностями, с которыми она должна быть связана. Она идентифицирует эти внешние сущности, а также единственный процесс, отражающий главную цель. Ее полезность заключается в том, что она устанавливает границы анализируемой системы. Каждый проект должен иметь ровно одну контекстную диаграмму.

DFD первого уровня строится как декомпозиция процесса, который присутствует на контекстной диаграмме. Построенная диаграмма первого уровня также имеет множество процессов, которые в свою очередь могут быть декомпозированы в DFD нижнего уровня. Таким образом, строится иерархия DFD с контекстной диаграммой в корне дерева. Этот процесс декомпозиции продолжается до тех пор, пока процессы могут быть эффективно описаны с помощью коротких миниспецификаций (спецификаций процессов).

СЛОВАРЬ ДАННЫХ

Словарь данных представляет собой определенным образом организованный список всех элементов данных системы с их точными определениями, что дает возможность различным категориям пользователей (от системного аналитика до программиста) иметь общее понимание всех входных и выходных потоков и компонент хранилищ.

Для каждого потока данных в словаре хранится имя потока, его тип и атрибуты. Информация по каждому потоку состоит из ряда словарных статей, каждая из которых начинается с ключевого слова - заголовка статьи, которому предшествует символ “***@***”.

По типу потока в словаре содержится информация, идентифицирующая: простые или групповые потоки; внутренние или внешние потоки; потоки данных или потоки управления; непрерывные или дискретные потоки.

Атрибуты потока данных включают: имена-синонимы потока данных в соответствии с узлами изменения имени; единицы измерения потока; диапазон значений для непрерывного потока, типичное его значение; список значений и их смысл для дискретного потока; список номеров диаграмм различных типов, в которых поток встречается; комментарий, включающий дополнительную информацию.

МЕТОДЫ ЗАДАНИЯ СПЕЦИФИКАЦИИ ПРОЦЕССОВ

Спецификация процесса (СП) используется для описания функционирования процесса в случае отсутствия необходимости детализировать его с помощью DFD. Фактически СП представляют собой алгоритмы описания задач, выполняемых процессами: множество всех СП является полной спецификацией системы. СП содержат номер и/или имя процесса, списки входных и выходных данных и описание процесса, являющееся спецификацией алгоритма или операции, трансформирующей входные потоки данных в выходные. Известно большое число 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.

Независимо от используемой нотации спецификация процесса должна начинаться с ключевого слова, например, ***@СПЕЦПРОЦ***. Требуемые входные и выходные данные должны быть специфицированы следующим образом:

@ВХОД = <имя символа данных>
@ВЫХОД = <имя символа данных>
@ВХОДВЫХОД = <имя символа данных>,
где <имя символа данных> - соответствующее имя из словаря данных.

Спецификации должны удовлетворять следующим требованиям:

- для каждого процесса нижнего уровня должна существовать одна спецификация;
- спецификация определяют способ преобразования входных потоков в выходные;
- спецификация должна стремиться к ограничению избыточности;

Ниже рассматриваются некоторые наиболее часто используемые методы задания спецификаций процессов.

СТРУКТУРИРОВАННЫЙ ЕСТЕСТВЕННЫЙ ЯЗЫК

Структурированный естественный язык применяется для читабельного, строгого описания спецификаций процессов. Он является разумной комбинацией строгости языка программирования и читабельности естественного языка и состоит из подмножества слов, организованных в логические структуры, арифметических выражений и диаграмм.

Управляющие структуры языка имеют один вход и один выход. К ним относятся:

1) последовательная конструкция: ***ВЫПОЛНИТЬ*** *функция1*
ВЫПОЛНИТЬ *функция2*

2) конструкция выбора: ***ЕСЛИ*** *<условие>* ***ТО***
ВЫПОЛНИТЬ *функция1*
ИНАЧЕ
ВЫПОЛНИТЬ *функция2*
КОНЕЦЕСЛИ

3) итерация: ***ДЛЯ*** *<условие>*
ВЫПОЛНИТЬ *функция*
КОНЕЦДЛЯ

или ***ПОКА*** *<условие>*
ВЫПОЛНИТЬ *функция*
КОНЕЦПОКА

ТАБЛИЦЫ И ДЕРЕВЬЯ РЕШЕНИЙ

Для описания некоторых типов преобразований традиционно используются таблицы и деревья решений. Проектирование спецификаций процессов с помощью ***таблиц решений*** (ТР) заключается в задании матрицы, отображающей множество входных **условий** в множество **действий**. Верхняя часть таблицы используется для определения условий. Обычно условие ***ЕСЛИ*** является частью оператора ***ЕСЛИ-ТО*** и требует ответа "да-нет". Однако иногда в условии может присутствовать и ограниченное множество значений, например, ***ЯВЛЯЕТСЯ ЛИ ДЛИНА СТРОКИ БОЛЬШЕЙ ИЛИ РАВНОЙ ГРАНИЧНОМУ ЗНАЧЕНИЮ****?*

Нижняя часть ТР используется для определения действий, т.е. ***ТО*** - части оператора ***ЕСЛИ - ТО***. Так, в конструкции: ***ЕСЛИ*** *ИДЕТ ДОЖДЬ,* ***ТО*** *РАСКРЫТЬ ЗОНТ, ИДЕТ ДОЖДЬ* является условием, а *РАСКРЫТЬ ЗОНТ* - действием.

Вариантом таблицы решений является дерево решений (ДР), позволяющее взглянуть на процесс условного выбора с позиции схемы. Обычно ДР используется при малом числе действий и когда не все комбинации условий возможны.

ВИЗУАЛЬНЫЕ ЯЗЫКИ ПРОЕКТИРОВАНИЯ СПЕЦИФИКАЦИЙ

Визуальные языки проектирования 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 является подход с использованием FLOW-форм. 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 (рисунок 12).

Рисунок 12 - Символы FLOW-форм

Каждый символ является блоком обработки. Каждый прямоугольник внутри символа также представляет блок обработки. На рисунке 13 приведен пример использования данного подхода при проектировании спецификации процесса, обеспечивающего упорядочивание элементов массива методом "поплавка".

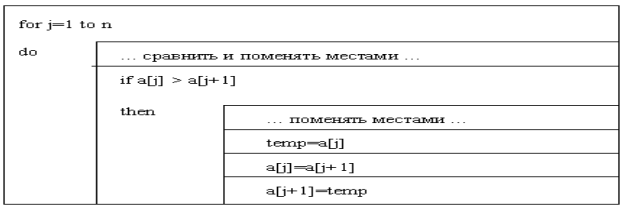


Рисунок 13 - Пример FLOW-формы

СРАВНЕНИЕ МЕТОДОВ

Методы задания спецификаций процессов в соответствии с увеличением трудности их проектирования приведены на рисунке 14. Наиболее трудным методом задания СП являются языки программирования. Языки программирования концентрируют внимание на деталях реализации, а потоки данных в DFD представляются абстрактно (их фактическая композиция определяется в словаре данных). ***Поэтому сложность - не в написании СП, а в их синхронизации и согласовании с DFD, поскольку при редактировании DFD должны корректироваться и спецификации процессов***.

Текстовое описание	*Структурированный естественный язык*	*Таблица решений*	*Дерево решений*	*Визуальный язык*	*Язык программирования*

Рисунок 14 - Спектр методов задания спецификаций процессов

Структурированный естественный язык применяется в случаях, когда детали СП известны не полностью. Он обеспечивает быстрое проектирование СП, прост в использовании, легко понимаем проектировщиками, программистами и пользователем. К его недостаткам относятся отсутствие процедурных возможностей и неспособность к автоматической кодогенерации из-за наличия неоднозначностей.

Таблицы и деревья решений позволяют управлять сложными комбинациями условий и действий, обеспечивают визуальное (табличное и графическое) представление СП и легко понимаемы конечным пользователем. Главным недостатком методов является отсутствие процедурных возможностей.

Визуальные языки проектирования поддерживаются автоматической кодогенерацией. Их недостаток - трудность модификации СП при изменении деталей.

ДИАГРАММЫ “СУЩНОСТЬ-СВЯЗЬ”

Диаграммы "сущность-связь" (ERD) предназначены для разработки моделей данных и обеспечивают стандартный способ определения данных и отношений между ними. Фактически с помощью ERD осуществляется детализация хранилищ данных проектируемой системы, а также документируются сущности системы и способы их взаимодействия, включая идентификацию объектов (сущностей), свойств этих объектов (атрибутов) и их отношений с другими объектами (связей). Данная нотация была введена Ченом (Chen) и получила дальнейшее развитие в работах Баркера (Barker).

НОТАЦИЯ БАРКЕРА

Баркер предложил 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 себя имя сущности, список имен атрибутов (возможно, неполный) и указатели ключевых атрибутов (знак "#" перед именем атрибута).

Связи являются бинарными и представляются линиями с двумя концами (соединяющими сущности), для которых должно быть определено имя, степень множественности (один или много объектов участвуют в связи) и степень обязательности (обязательная или необязательная связь между сущностями). Для множественной связи линия присоединяется к прямоугольнику сущности в трех точках, а для одиночной связи - в одной точке. При обязательной связи рисуется непрерывная линия до середины связи, при необязательной - пунктирная линия. На рисунке 15 приведен фрагмент ERD в нотации Баркера.

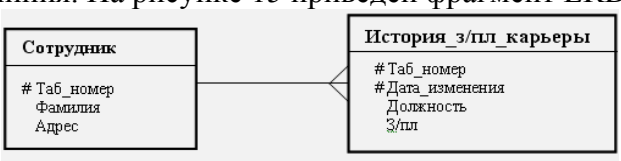


Рисунок 15 - Нотация Баркера

Каждый СОТРУДНИК может ИМЕТЬ одну или более ИСТОРИЯ_З/ПЛ_КАРЬЕРЫ* или *Каждая ИСТОРИЯ_З/ПЛ_КАРЬЕРЫ должна ПРИНАДЛЕЖАТЬ ровно одному СОТРУДНИКУ.

ДИАГРАММЫ ПЕРЕХОДОВ СОСТОЯНИЙ

Диаграммы переходов состояний (STD) позволяют осуществлять декомпозицию управляющих процессов и описывают отношения между входными и выходными управляющими потоками на управляющем процессе-предке (рисунок 16).

STD состоит из следующих объектов:

СОСТОЯНИЕ - как условие устойчивости системы. Можно определить очередное состояние в зависимости от текущих входных событий. Имя состояния должно отражать реальную ситуацию, например, *НАГРЕВАНИЕ, ОХЛАЖДЕНИЕ* и т.п.

НАЧАЛЬНОЕ СОСТОЯНИЕ - узел STD, являющийся стартовой

ПЕРЕХОД определяет перемещение моделируемой системы из одного состояния в другое, он идентифицирует ***событие***, возникающее при выполнении некоторого ***условия.***

УСЛОВИЕ представляет собой событие, вызывающее переход и идентифицируемое именем перехода. Кроме условия с переходом может связываться ***действие***.

ДЕЙСТВИЕ - это операция при выполнении перехода.

В STD состояния представляются узлами, а переходы - дугами. Условия идентифицируются именем перехода и возбуждают выполнение перехода. Действия или отклики на события привязываются к переходам и записываются под соответствующим условием.

Рисунок 16 – Диаграмма переходов-состояний

СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ

Проектирование - фаза ЖЦ, на которой вырабатывается, как реализуются требования пользователя, которые порождены и зафиксированы на фазе анализа. На этом этапе осуществляется построение ***модели реализации***, демонстрирующей, **как система будет удовлетворять предъявленным к ней требованиям**.

Техника структурных карт (схем) используется на фазе проектирования для того, чтобы продемонстрировать, каким образом системные требования будут отражаться комбинацией программных структур. При этом наиболее часто применяются две техники: структурные карты Константайна (Constantine), предназначенные для описания отношений между модулями, и структурные карты Джексона (Jackson), предназначенные для описания внутренней структуры модулей.

СТРУКТУРНЫЕ КАРТЫ КОНСТАНТАЙНА

Структурные карты Константайна являются моделью отношений между программными модулями. Базовым элементом структурной карты является модуль.

МОДУЛЬ - используется для представления обрабатывающего фрагмента.

ПОДСИСТЕМА - ранее определенный детализированный модуль.

БИБЛИОТЕКА - определена вне проекта данной системы.

ОБЛАСТЬ ДАННЫХ - используется для указания модулей, содержащих области глобальных/распределенных данных.

СТРУКТУРНЫЕ КАРТЫ ДЖЕКСОНА

Техника ***структурных карт Джексона*** основана на методологии структурного программирования Джексона и заключается в продуцировании диаграмм (структурных карт) для графического иллюстрирования внутримодульных (а иногда и межмодульных) связей и документирования проекта архитектуры системы ПО. При этом техника позволяет осуществлять проектирование нижнего уровня структуры ПО.

По аналогии со структурными картами Константайна диаграмма Джексона может включать объекты следующих типов:

СТРУКТУРНЫЙ блок (базовая компонента методологии) представляет частную функцию или блок кодов с одним входом и одним выходом.

ПРОЦЕДУРНЫЙ блок является специальным видом структурного блока, представляющим вызов ранее определенной процедуры.

БИБЛИОТЕЧНЫЙ блок аналогичен процедурному блоку.

Для взаимоувязывания блоков используются связи следующих типов: последовательная связь; параллельная связь; условная связь; итерационная связь.

ХАРАКТЕРИСТИКИ МОДУЛЕЙ

Один из фундаментальных принципов структурного проектирования заключается в том, что большая система должна быть расчленена на модули. При этом существенным является то, что это расчленение должно быть выполнено таким образом, чтобы модули были как можно более независимы (критерий сцепления - **coupling**), и чтобы каждый модуль выполнял единственную (связанную с общей задачей) функцию (критерий связности - **cohesion**). Кроме этих двух взаимно дополняющих друг друга критериев в структурном проектировании существует целый ряд других руководящих принципов, которые могут применяться для оценки и улучшения качества проекта на основании структурных карт.

СЦЕПЛЕНИЕ

Одним из способов оценки качества проекта является анализ сцепления модулей. Сцепление является мерой взаимозависимости модулей. В хорошем проекте сцепления должны быть минимизированы, т.е. модули должны быть слабозависимыми (независимыми) настолько, насколько это возможно. Слабое сцепление между модулями служит признаком хорошо спроектированной системы по ***следующим причинам***:

- ***уменьшение количества соединений*** между двумя модулями приводит к уменьшению вероятности появления “волнового эффекта” (ошибка в одном модуле влияет на работу других модулей);
- ***минимизация риска появления “эффекта ряби***” (внесение изменений, например, при исправлении ошибки, приводит к появлению новых ошибок), т.к. изменение влияет на минимальное количество модулей;

Слабое сцепление может быть достигнуто за счет комбинирования трех следующих ***способов действий***: ***удаления*** необязательных связей; ***уменьшения*** количества необходимых связей; ***упрощением*** необходимых связей.

Практические рекомендации для ослабления сцепления модулей: минимальные по количеству параметров межмодульные связи; прямые, а не косвенные межмодульные связи; локализованные связи; явные связи; гибкие связи для облегчения модификаций.

На практике ***существуют три основных типа сцепления***, используемых системными проектировщиками для связи модулей: ***нормальное сцепление, сцепление по общей области и сцепление по содержимому.*** С позиций структурного проектирования эти типы являются, соответственно, приемлемым, неприемлемым и запрещенным.

Два модуля А и В являются нормально сцепленными, если А вызывает В; В возвращает управление А; вся информация, передаваемая между А и В, представляется значениями параметров при вызове.

Существует три типа нормального сцепления: ***сцепление по данным, сцепление по образцу, сцепление по управлению.***

На практике наиболее часто используемым типом сцепления является ***сцепление по данным*** (data coupling). Два модуля сцеплены по данным, если они взаимодействуют через передачу параметров и при этом каждый параметр является элементарным информационным объектом.

Два модуля ***сцеплены по образцу***, если один посылает другому составной информационный объект. Примером составного объекта может быть объект *Данные о клиенте*, включающий в себе поля: *Название организации, Почтовый адрес, Номер счета*.

Два модуля ***сцеплены по управлению***, если один посылает другому информационный объект - флаг, предназначенный для управления его внутренней логикой.

Два модуля являются сцепленными по общей области, если они ссылаются к одной и той же области глобальных данных.

Два модуля являются сцепленными по содержимому, если один модуль передает управление или выполняет переход в другой модуль.

Необходимо отметить, что любые два модуля могут быть сцеплены более чем одним способом. В этом случае ***тип их сцепления определяется худшим типом сцепления***. Например, если два модуля сцеплены по образцу и общей области, то они характеризуются как сцепленные по общей области. Они по-прежнему сцеплены по образцу, но это сцепление выше, чем сцепление по общей области.

В таблице 2 приведены конкретные характеристики каждого типа сцепления.

Таблица 2

Тип сцепления	Устойчивость к волн. эффекту	Модифицируемость	Понятность	Используемость в других системах
data coupling	-	хорошая	хорошая	хорошая
stamp coupling	-	средняя	средняя	средняя
control coupling	средняя	плохая	плохая	плохая
common coupling	плохая	средняя	плохая	плохая
content coupling	плохая	плохая	плохая	плохая

СВЯЗНОСТЬ

Сцепление является лишь одним из критериев оценки качества разбиения системы на части: он оценивает, насколько хорошо модули отделены друг от друга. Другим критерием оценки качества расчленения системы является критерий связности, контролирующий, как действия в одном модуле связаны друг с другом. Фактически сцепление и связность являются двумя взаимозависимыми способами измерения расчленения системы на части: связность модуля часто определяет качество его сцепления с другими модулями.

Связность - это мера прочности соединения функциональных и информационных объектов внутри одного модуля. Размещение сильно связанных объектов в одном и том же модуле уменьшает межмодульные взаимосвязи и взаимовлияния. Специалисты выделяют следующие ***уровни связности: функциональная, последовательная, информационная, процедурная, временная, логическая и случайная.***

Функционально связный модуль содержит объекты, предназначенные для выполнения только одной задачи, например: *Расчет заработанной платы, Считывание данных кредитной карты.*

Модуль имеет последовательную связность, если его объекты охватывают подзадачи, для которых выходные данные одной из подзадач служат входными данными следующей, например: *Открыть файл - Прочитать запись - Закрыть файл*.

Информационно связный модуль содержит объекты, использующие одни и те же входные или выходные данные.

Процедурно связный модуль является модулем, объекты которого включены в различные (и возможно несвязные) подзадачи, в которых управление переходит от каждой подзадачи к последующей (отметим, что в последовательно связном модуле данные, а не управление, переходили от одной подзадачи к последующей).

Временно связным является модуль, объекты которого включены в подзадачи, связанные временем исполнения.

Модулем с логической связностью является модуль, объекты которого содействуют решению одной общей подзадачи, для которой эти объекты отобраны во внешнем по отношению к модулю мире. Логически связный модуль содержит некоторое количество подзадач (действий) одного и того же общего вида.

Случайно связным является модуль, объекты которого соответствуют подзадачам, незначительно связанным друг с другом. Случайно связный модуль подобен логиче-

ски связному модулю, его объекты не связаны ни потоками данных, ни потоками управления. Однако, подзадачи в логически связном модуле являются по крайней мере одной категории; для случайно связного модуля даже это неверно.

В таблице 3 приведены конкретные характеристики каждого уровня связности.

Таблица 3

Уровень связности	Сцепление	Модифицир.	Понятность	Сопровожд.
функциональная	хорошее	хорошая	хорошая	хорошая
последовательная	хорошее	хорошая	ближе к хорошей	хорошая
информационная	среднее	средняя	средняя	средняя
процедурная	переменная	переменная	переменная	плохая
временная	плохое	средняя	средняя	плохая
логическая	плохое	плохая	плохая	плохая
случайная	плохое	плохая	плохая	плохая

Таким образом, связность является мерой функциональной зависимости объектов (исполняемых операторов, областей данных и т.д.) внутри одного модуля. В хорошем проекте связность каждого модуля является высокой. Вместе со сцеплением, связность является одним из лучших критериев оценки качества проекта.

ДРУГИЕ ПРИНЦИПЫ ПРОЕКТИРОВАНИЯ

Очевидно, что для оценки качества проектируемой системы критериев сцепления и связности недостаточно. Например, если бы мы осуществляли оценку только по критерию сцепления, мы бы всегда получали системы, состоящие из одного модуля. Связность этого единственного модуля также была бы вполне приемлемой. Ниже кратко рассматриваются другие принципы проектирования, позволяющие получать качественные системы.

1 ***Принцип факторизации***. Под факторизацией понимается выделение подзадачи, реализуемой некоторым модулем, в новый самостоятельный модуль. Это может быть сделано по следующим причинам: чтобы уменьшить размеры модуля; чтобы обеспечить преимущества классического проектирования сверху-вниз; чтобы избежать дублирования подзадачи, выполняемой более чем одним модулем; чтобы отделить собственно вычисления от управления; чтобы обеспечить пригодность модулей для их использования в различных частях системы; чтобы упростить реализацию.

2 ***Принцип единства решения***. Процесс любого решения состоит из двух частей: распознавания того, какое действие выбрать, и исполнения этого действия. Поскольку эти две составляющие решения различны, то информационные объекты, используемые при распознавании и исполнении также могут существенно различаться и, следовательно, могут быть недоступными в одном модуле. Такая ситуация получила название ***расщепления решения (decision split)***. Сильное расщепление решения (хотя иногда расщепления не удается избежать) обычно является симптомом плохой организации модулей. Исполнительная часть решения должна располагаться как можно ближе к распознавательной части для того, чтобы быть обработанной.

3 ***Обработка ошибок***. Сообщения об ошибках целесообразно формировать и визуализировать в модуле, который ошибку обнаруживает и “знает”, что это за ошибка. Тексты сообщений рекомендуется хранить вместе.

4 ***Принцип отсутствия памяти***. Вызванный модуль возвращает управление вызвавшему его модулю после выполнения своей функции и “умирает”, оставляя после себя результат. При повторном вызове он делает свою работу так, как будто бы он родился впервые. Модуль не помнит, что происходило в его предыдущих жизнях.

5 ***Инициализация и завершение***. Как правило, модули инициализации и завершения являются трудными для сопровождения из-за их плохой (временной) связности и сильного сцепления. Общая рекомендация по решению этой проблемы - инициализацию каждой функции желательно выполнять как можно позже, а действия по завершению каждой функции должны производиться как можно раньше.

6 ***Нагрузка по входу и выходу***. Под нагрузкой модуля по входу - количество непосредственных вызывающих его модулей, по выходу - количество непосредственно подчиненных ему модулей. Нагрузка по выходу не должна превышать 6-7 модулей. Высокая нагрузка по входу требует от модуля хорошей связности.

SADT- МЕТОДОЛОГИЯ СТРУКТУРНОГО АНАЛИЗА И ПРОЕКТИРОВАНИЯ

(Structured Analysis and Design Technique) - одна из самых известных методологий анализа и проектирования систем, *введенная Россом (Ross). SADT успешно использовалась в военных, промышленных и коммерческих организациях для решения широкого спектра задач. SADT-модель обеспечивает возможность исследования динамических характеристик бизнес-процессов, основанных на цветных (раскрашенных) сетях Петри CPN (Color Petri Nets).* ***Методология SADT успешно работает для реорганизации хорошо специфицированных и стандартизованных западных бизнес-процессов, поэтому она и принята на Западе в качестве типовой.***

Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Основные ***концепции***:

- ***графическое представление*** блочного моделирования. Графика блоков и дуг SADT-диаграммы отображает функцию в виде блока, а интерфейсы входа/выхода представляются дугами, соответственно входящими в блок и выходящими из него.

- ***строгость и точность***. Правила SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия аналитика.

Правила SADT включают: ***ограничение количества блоков на каждом уровне декомпозиции*** (3-6 блоков); ***связность*** диаграмм (номера блоков); ***уникальность меток*** и наименований (отсутствие повторяющихся имен); ***синтаксические правила*** для графики (блоков и дуг); ***разделение входов и управлений*** (правило определения роли данных); ***отделение организации от функции***.

МЕТОДОЛОГИЯ IDEF0

Методология IDEF0 незначительно отличается от классической схемы описания бизнес-процессов DFD. Основным отличием является наличие в языке дополнительной аналитики. Данный стандарт описания бизнес-процессов предлагает ввести три типа входов. ***Первый тип входов назвали так же входом, а два других входа назвали управлением и механизмами***.

В стандарте IDEF0 ***показывают объекты*** – информационные и материальные потоки, которые преобразуются в бизнес-процессе. ***С помощью управления*** показывают объекты – материальные и информационные потоки, которые не преобразуются в процессе, но нужны для его выполнения. ***С помощью механизмов*** стали показывать механизмы, при помощи которых бизнес-процесс реализуется: технические средства, люди, информационные системы и т.д. Выход бизнес-процесса в стандарте IDEF0 полностью соответствует по смыслу выходу процесса, описанному при помощи DFD-схемы. ***Четыре типа объектов,*** применяемых для описания входов и выходов в стандарте IDEF0: Вход (Input), Управление (Control), Выход (Output), Механизм (Mechanism).

Рассмотрим пример бизнес-процесса "Выточить деталь", который выполняет токарь. Входом является заготовка, из которой вытачивается деталь – она физически преобразуется в процессе. Для работы ему нужно дать задание и чертеж с размерами детали. Чертеж, задание ***по ходу выполнения процесса не преобразуются***. Согласно стандарту IDEF0 их относят к управлению. Для того, что бы выточить деталь, нужен токарь, станок – их относят к механизмам. Выходом бизнес-процесса является деталь (рисунок 17).

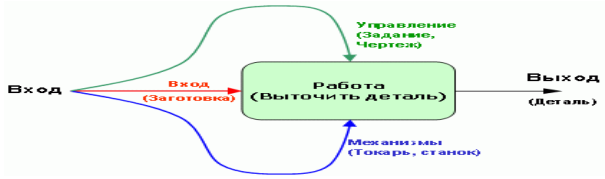


Рисунок 17 - Стандарт описания бизнес-процесса IDEF0

Тем не менее, стандарт IDEF0 имеет большое распространение, так как по нему существует много различных информационно-методических материалов и программные продукты, поддерживающие данный стандарт, овладеть которым несложно.

Методологию IDEF0 можно считать следующим этапом развития хорошо известного графического языка описания функциональных систем SADT (Structured Analysis and Design Teqnique). Исторически IDEF0 как стандарт был разработан в 1981 году в рамках обширной программы автоматизации промышленных предприятий, которая носила обозначение ICAM (Integrated Computer Aided Manufacturing). Семейство стандартов IDEF унаследовало свое обозначение от названия этой программы (IDEF=Icam DEFinition), и последняя его редакция была выпущена в декабре 1993 года Национальным Институтом по Стандартам и Технологиям США (NIST).

Работы (Activity), которые означают некие поименованные процессы, функции или задачи, изображаются в виде прямоугольников. Именем работы должен быть глагол или глагольная форма (например "Изготовление детали", "Прием заказа" и т.д.). Взаимодействие работ с внешним миром и между собой описывается в виде стрелок.

Стрелки управления, выхода, механизма и выхода на контекстной диаграмме служат для описания взаимодействия системы с окружающим миром. Они могут начинаться у границы диаграммы и заканчиваться у работы или наоборот. Имена вновь внесенных стрелок автоматически заносятся в словарь (Arrow Dictionary). Словарь стрелок можно распечатать в виде отчета (меню Report / Arrow Report...).

Стрелки представляют собой некую информацию и именуются существительными (например, "Заготовка", "Изделие", "Заказ"). В IDEF0 различают ***пять типов стрелок***:

- ***Вход - информация***, которая используется или преобразовывается работой.
- ***Управление - правила***, стратегии, процедуры или стандарты, которыми руководствуется работа. Каждая работа должна иметь хотя бы одну стрелку управления.
- ***Выход - материал или информация***, которая производится работой. Каждая работа должна иметь хотя бы одну стрелку выхода.
- ***Механизм - ресурсы***, которые выполняют работу, например, персонал.
- ***Вызов*** - специальная стрелка, указывающая на другую модель работы.

В IDEF0 различают несколько типов связей работ (рисунок 18):

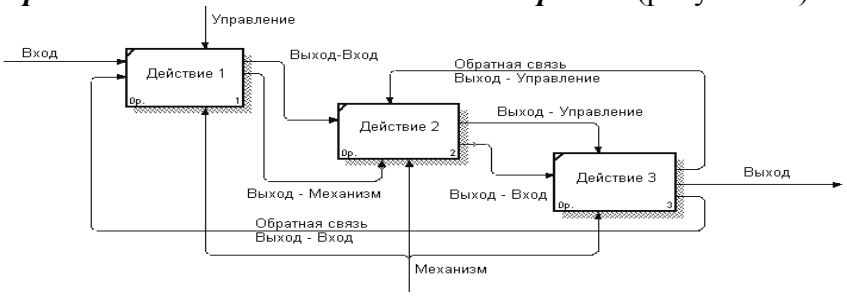


Рисунок 18 – Типы связей IDEF0

МЕТОДОЛОГИЯ IDEF3

Для описания временной последовательности и алгоритмов выполнения работ был разработан стандарт IDEF3, который входит в семейство стандартов IDEF.

Метод моделирования IDEF3, являющийся частью семейства стандартов IDEF, предназначен для моделирования последовательности выполнения действий и взаимозависимости между ними в рамках процессов. Метод приобрел широкое распространение

среди системных аналитиков как дополнение к методу функционального моделирования IDEF0 (модели IDEF3 могут использоваться для детализации функциональных блоков).

Основой модели IDEF3 служит так называемый сценарий процесса, который выделяет последовательность действий и подпроцессов анализируемой системы.

Как и в методе IDEF0, основной единицей модели IDEF3 является диаграмма. Другой важный компонент модели — действие, или в терминах IDEF3 "единица работы" (Unit of Work). Все связи в IDEF3 являются однонаправленными.

Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Смысл каждого типа приведен в таблице 4.

Таблица 4

Обозначение	Наименование	Слияние стрелок (Fan-in Junction)	Разветвления стрелок Fan-out Junction)
&	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
&	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
O	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
O	Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
X	XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

Стандарт IDEF3 предназначен для описания бизнес-процессов нижнего уровня и содержит объекты – логические операторы, с помощью которых показывают альтернативы и места принятия решений в бизнес-процессе, а также стрелки, с помощью которых показывают временную последовательность работ в бизнес-процессе (рисунок 19).

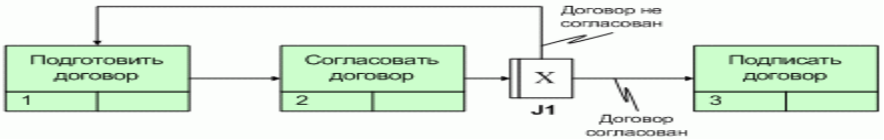


Рисунок 19 - Схема бизнес-процесса в стандарте IDEF3

Помимо наличия нескольких типов связей между работами в стандарте IDEF3 логические операторы, которые называются перекрестками, также делятся на несколько типов: "Исключающий ИЛИ", "И" и "ИЛИ".

Перекресток "Исключающий ИЛИ" обозначает, что после завершения работы "А" (рисунок 20), начинает выполняться только одна из трех расположенных параллельно работ B, C или D в зависимости от условий 1, 2 и 3.

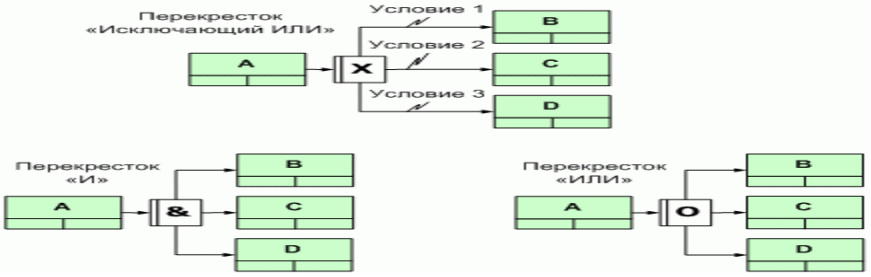


Рисунок 20 - Применение перекрестков "Исключающее ИЛИ", "И" и "ИЛИ"

Перекресток "И" обозначает, что после завершения работы "А", начинают выполняться одновременно три параллельно расположенные работы B, C и D. Перекресток "ИЛИ" обозначает, что после завершения работы "А", может запуститься любая комбинация трех параллельно расположенных работ B, C и D. Например может запуститься только одна из них, могут запуститься три работы, а также могут запуститься двойные комби-

нации B и C, либо C и D, либо B и D. Перекресток "Исключающий ИЛИ" является самым неопределенным, так как предполагает несколько возможных сценариев реализации бизнес-процесса и применяется для описания слабо формализованных ситуаций.

Перекрестки "И" и "ИЛИ" подразделяются еще на два подтипа – синхронные и асинхронные. Перекрестки синхронного типа обозначают, что работы B, C и D запускаются одновременно после завершения работы A. Перекрестки асинхронного типа требований к одновременности не предъявляют.

Приведенные на рисунке 22 схемы взаимосвязи работ и перекрестков называются схемами расхождения, так как от перекрестков расходятся несколько работ. Существуют и другие схемы взаимосвязи перекрестков и работ – это так называемые схемы схождения, когда к перекрестку подходит несколько работ (рисунок 21).

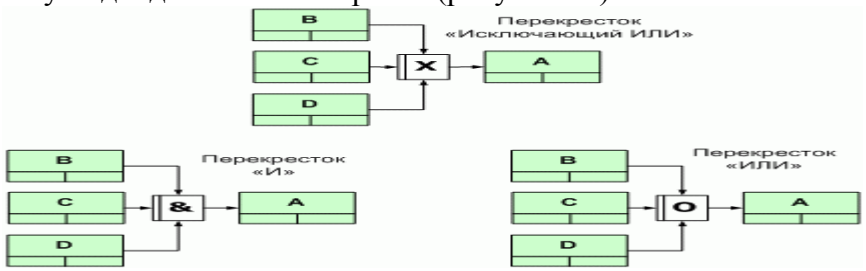


Рисунок 21 - Применение перекрестков "Исключающее ИЛИ", "И" и "ИЛИ"

Примеры. На рисунке 22 после обнаружения пожара инициируются включение пожарной сигнализации, вызов пожарной охраны, и начинается тушение пожара. Запись в журнал производится только тогда, когда все три перечисленных действия завершены.

Рисунок 22 - Соединения "и"

На рисунке 23 соединение "исключающее "или"" используется для отображения факта, что студент не может одновременно быть на лекции по двум разным курсам.

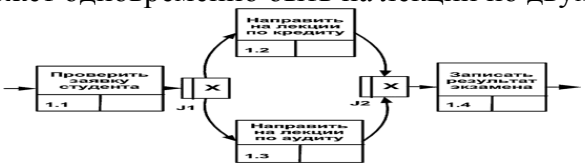


Рисунок 23 - Соединение "исключающее "или""

На рисунке 24 соединение J2 может активизировать проверку данных чека и/или проверку суммы наличных. Проверка чека инициируется, если покупатель желает расплатиться чеком, проверка суммы наличных — при оплате наличными. И то, и другое действие инициируются при частичной оплате как чеком, так и наличными.

Рисунок 24- Соединения "или"

В рассмотренных примерах все действия выполнялись асинхронно, т.е. они не инициировались одновременно. Однако существуют случаи, когда действия должны выполняться синхронно.

Все соединения на диаграммах должны быть парными, из чего следует, что любое разворачивающее соединение имеет парное себе сворачивающее. Однако типы соединений не обязательно должны совпадать. Соединения могут комбинироваться для создания более сложных ветвлений. Комбинации соединений следует использовать с осторожностью, чтобы диаграммы не оказались сложными для восприятия.

МОДЕЛЬ ДАННЫХ

ОБЩАЯ ХАРАКТЕРИСТИКА СУБД

Метод IDEF1, разработанный Т.Рэмей (T.Ramey), основан на подходе Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме. В настоящее время на основе совершенствования методологии IDEF1 создана ее новая версия - методология IDEF1X. IDEF1X разработана с учетом таких требований, как простота изучения и возможность автоматизации. IDEF1X-диаграммы используются рядом CASE-средств (в частности, ERwin, Design/IDEF).

Методология IDEF1X использует все основные концепции построения СУБД.

АРХИТЕКТУРА КЛИЕНТ-СЕРВЕР

Принцип централизации хранения и обработки данных является базовым принципом архитектуры “клиент-сервер”. Для его реализации используется так называемый сервер баз данных, выполненный как приложение или сервис операционной системы, и только он может реально манипулировать файлами, в которых хранятся данные.

Сервер баз данных осуществляет целый комплекс действий по управлению данными. ***Основными его обязанностями являются***: выполнение пользовательских запросов на выбор и модификацию данных и метаданных, получаемых от клиентских приложений, функционирующих на персональных компьютерах локальной сети; хранение и резервное копирование данных; поддержка ссылочной целостности данных по определенным в БД правилам; обеспечение авторизованного доступа к данным на основе проверки прав и привилегий пользователей; протоколирование операций и ведение журнала транзакций.

В простейшем случае ***клиент-серверная информационная система состоит из двух основных компонентов:*** сервера баз данных, управляющего данными и выполняющего запросы клиентских приложений; клиентских приложений, предоставляющих интерфейс пользователя и посылающих запросы к серверу.

Существуют и более сложные реализации архитектуры “клиент-сервер”, например многозвенные информационные системы с использованием серверов приложений, реализующих бизнес-логику и осуществляющих обработку данных.

ПРЕИМУЩЕСТВА АРХИТЕКТУРЫ КЛИЕНТ-СЕРВЕР

Информационные системы, использующие архитектуру “клиент-сервер”, обладают преимуществами: снижение сетевого трафика при выполнении запросов, например, при необходимости выбора пяти записей из таблицы, содержащей миллион записей; возможность хранения бизнес-правил (например, правил ссылочной целостности или ограничений на значения данных) на сервере, что позволяет избежать дублирования кода в различных клиентских приложениях, использующих общую базу данных; использование CASE-средства для создания диаграмм “сущность-связь”. Это позволяет описать подобные правила на уровне логической и физической “модели данных”, а затем сгенерировать код соответствующих серверных объектов — триггеров, хранимых процедур, серверных ограничений. В этом случае клиентские приложения будут избавлены от значительной части кода, связанного с реализацией бизнес-правил непосредственно в приложении. Отметим также, что часть кода, связанного с обработкой данных, также может быть реализована в виде хранимых процедур сервера, что позволяет еще больше “облегчить” клиентское приложение, а это означает снижение стоимости информационной системы.

Большинство серверных СУБД поддерживает так называемые роли, представляющие собой совокупность прав на доступ к тем или иным объектам базы данных. В этом случае каждый пользователь может иметь одну или несколько ролей и соответственно определенные в этих ролях привилегии.

ЯЗЫК SQL

Structured Query Language (SQL) — это непроцедурный язык, используемый для формулировки запросов к базам данных в большинстве современных СУБД и в настоящий момент являющийся индустриальным стандартом.

Непроцедурность языка означает, что на нем можно указать, что нужно сделать с базой данных, но нельзя описать алгоритм этого процесса. Все алгоритмы обработки SQL-запросов генерируются самой СУБД и не зависят от пользователя. ***Язык SQL состоит из набора операторов, которые можно разделить на несколько категорий:*** Data Definition Language (DDL) — язык определения данных, позволяющий создавать, удалять и изменять объекты в базах данных; Data Manipulation Language (DML) — язык управления данными, позволяющий модифицировать, добавлять и удалять данные в имеющихся объектах базы данных; Data Control Languages (DCL) — язык, используемый для управления пользовательскими привилегиями; Transaction Control Language (TCL) — язык для управления изменениями, сделанными группами операторов; Cursor Control Language (CCL) — операторы для определения курсора, подготовки операторов SQL к выполнению и некоторых других операций.

Триггеры и хранимые процедуры пишутся на *процедурном* языке, характерном для данной СУБД. В большинстве СУБД такие языки представляют собой процедурные расширения SQL и помимо обычных операторов SQL содержат некоторый набор алгоритмических конструкций, например begin…end, if…then…else и т.д.

В отличие от самого языка SQL, подчиняющегося стандарту ANSI, расширения SQL не стандартизованы. У каждой СУБД есть свой диалект процедурных расширений SQL (в СУБД Oracle это PL/SQL, в СУБД Microsoft SQL Server — Transact-SQL).

ЗАПРОСЫ К БАЗАМ ДАННЫХ

Модификация и выбор данных, изменение метаданных и некоторые другие операции осуществляются с помощью запросов (query). Большинство современных СУБД (и некоторые средства разработки приложений) содержат средства для генерации таких запросов. Один из способов манипуляции данными называется «queries by example» (QBE) — запрос по образцу. QBE представляет собой средство для визуального связывания таблиц и выбора полей, которые следует отобразить в результате запроса.

В большинстве СУБД визуальное построение запроса с помощью QBE приводит к генерации текста запроса с помощью специального языка запросов SQL.

ТРИГГЕРЫ И ХРАНИМЫЕ ПРОЦЕДУРЫ

Триггеры и хранимые процедуры, поддерживаемые в большинстве современных серверных СУБД, используются для хранения исполняемого кода.

Хранимая процедура — это специальный вид процедуры, который выполняется сервером баз данных. Хранимые процедуры пишутся на процедурном языке, который зависит от конкретной СУБД. Они могут вызывать друг друга, читать и изменять данные в таблицах, и их можно вызвать из клиентского приложения, работающего с базой данных.

Хранимые процедуры обычно используются при выполнении часто встречающихся задач (например, сведение бухгалтерского баланса). Они могут иметь аргументы, возвращать значения, коды ошибок и иногда наборы строк и колонок (такой набор данных иногда называется термином dataset).

Триггеры также содержат исполняемый код, но их нельзя вызвать из клиентского приложения или хранимой процедуры. Триггер всегда связан с конкретной таблицей и выполняется тогда, когда при редактировании этой таблицы наступает событие, с которым он связан (например, вставка, удаление или обновление записи).

ПАРАЛЛЕЛЬНАЯ ОБРАБОТКА ДАННЫХ

Повышение производительности с помощью параллельной обработки запросов в многопроцессорных системах связано с появлением первого продукта такого класса — Oracle Parallel Server. Серверы, поддерживающие параллельную обработку, разрешают нескольким процессорам обращаться к одной базе данных, что позволяет обеспечить высокую скорость обработки транзакций.

В настоящее время большинство производителей современных серверных СУБД поставляют на рынок версии, поддерживающие параллельную обработку данных. Обычно это версии для Windows NT и коммерческих версий UNIX.

OLAP И СОЗДАНИЕ ХРАНИЛИЩ ДАННЫХ

OLAP (On-Line Analytical Processing) представляет собой технологию построения многомерных хранилищ данных (Data Warehouses), как правило, агрегатных, то есть являющихся результатом обработки набора данных, нередко состоящего из нескольких таблиц. Такие хранилища данных в последнее время широко используются в системах поддержки принятия решений.

Многомерные хранилища данных могут быть реализованы как в виде набора обычных реляционных таблиц, так и в виде нереляционной многомерной базы данных. В последнем случае такое хранилище обычно управляется отдельным сервером. Многие производители серверных СУБД поставляют такие серверы отдельно (Oracle, Informix), некоторые включают их в состав сервера реляционных баз данных (Microsoft SQL Server 7.0). Нередко с целью повышения конкурентоспособности подобные OLAP-системы строят многомерные хранилища на основе данных из других СУБД, как это сделано, например, в Microsoft SQL Server OLAP Extensions и в Sybase Adaptive Server IQ.

НАИБОЛЕЕ ПОПУЛЯРНЫЕ СЕРВЕРНЫЕ СУБД

На сегодняшний день известно более двух десятков серверных СУБД, однако наиболее популярными, исходя из числа продаж и инсталляций, следует признать Oracle, Microsoft SQL Server, Informix, Sybase, DB2 (таблица 5).

Таблица 5

СУБД	Производитель	Url
Oracle	Oracle Corp.	http://www.oracle.com
Microsoft SQL Server	Microsoft	http://www.microsoft.com
Informix	Informix	http://www.informix.com
Sybase	Sybase	http://www.sybase.com
DB2	IBM	http://www-4.ibm.com

ORACLE

Oracle была первой коммерческой реляционной СУБД, поддерживающей ставший ныне индустриальным стандартом язык SQL; ее первая версия появилась в 1979 году. Фактически все это время Oracle является бессменным лидером на рынке производителей коммерческих СУБД и второй (после Microsoft) по величине компанией, производящей программное обеспечение.

Первоначально версии были предназначены для различных серверных платформ — различных версий UNIX, VMS и др. Позже были выпущены версии сервера Oracle для Novell NetWare. Первые версии этого сервера для персональных компьютеров появились в середине 90-х (Personal Oracle 7 for Windows 3.1, Personal Oracle 7 for Windows 95, Personal Oracle Lite, Oracle Workgroup Server 7 for Windows NT).

Oracle была первой компанией, создавшей СУБД, использовавшую предоставляемые некоторыми серверными платформами средства параллельных вычислений — Oracle Parallel Server (до его появления параллельные вычисления использовались только для решения научных задач). При использовании параллельных вычислений Oracle Parallel Server дает возможность нескольким процессорам обращаться к одной базе данных, что позволяет обеспечить высокую скорость обработки транзакций, а более поздние его версии дают возможность осуществить декомпозицию операций с большими объемами данных с целью параллельного выполнения их на нескольких процессорах.

Последней версией Oracle является версия Oracle 8i, отличительными свойствами которой являются: наличие объектных расширений и соответствующих типов данных, таких как вложенные таблицы, массивы, объекты и др. Иными словами, Oracle 8

и Oracle 8i являются объектно-ориентированными СУБД; наличие функций аналитической обработки данных (например, вычисления процентных соотношений, ранжирования, сравнения временных периодов); возможность создания таблиц, содержащих агрегатные данные (materialized views), и возможность частичного их обновления при изменении данных; поддержка Java, в частности JDK 1.2 и JDBC 2.0; поддержка XML, в частности в Oracle 8i включены XML Parser for Java, C/C++, PL/SQL, превращающие XML-данные в вид, пригодный для использования в Oracle 8; поддержка HTML- и XML-страниц с включенным в них кодом PL/SQL (для их выполнения требуются дополнительные продукты, например WebDB PL/SQL Gateway или Oracle Application Server PL/SQL Cartridge); поддержка хранения мультимедиа-данных с возможностью индексации, построения контекстных запросов, поддержки разных языков для хранимых документов; набор процедур и функций для обработки пространственной информации (Oracle Spatial); дополнительные возможности обеспечения безопасности, например шифрование данных, поддержка SSL, использование ролей уровня базы данных и уровня предприятия; возможность создания систем, устойчивых к сбоям, с использованием нескольких параллельных процессов; поддержка Microsoft Cluster Server; наличие OLE DB-провайдера для доступа к данным.

Oracle 8i существует в трех редакциях: Oracle 8i, Oracle 8i Enterprise Edition, Oracle 8i Personal Edition. Для создания многомерных хранилищ данных существует и отдельный продукт — Oracle Express OLAP. Помимо различных версий сервера баз данных среди продуктов Oracle имеется также Designer/2000 — ориентированный на эту СУБД, CASE-средство для анализа бизнес-процессов и проектирования данных, а также разработки клиентских приложений.

Производя собственные средства разработки, Oracle предоставляет своим пользователям возможность создавать клиентские приложения с помощью других средств. В частности, кроме стандартного клиентского API (Oracle Call Interface) клиентская часть Oracle содержит также объектную модель (Oracle Objects for OLE), позволяющую использовать клиентскую часть Oracle как набор COM-объектов для доступа к данным. Кроме того, клиентская часть Oracle содержит ODBC-драйвер для доступа к данным этой СУБД.

Из готовых информационных систем на базе Oracle следует отметить несколько крупных систем управления предприятием, в частности SAP/R3. На Западе нередко используются готовые решения от самой Oracle Corporation, объединенные под общим названием Oracle Applications, такие как Oracle Financials, Oracle Human Resources, Oracle Market Management, Oracle Project Systems и др.

МЕХАНИЗМЫ ДОСТУПА К ДАННЫМ

Существует несколько способов доступа к данным из средств разработки и клиентских приложений. Подавляющее большинство систем управления базами данных содержит в своем составе библиотеки, предоставляющие специальный ***прикладной программный интерфейс (Application Programming Interface, API)*** для доступа к данным этой СУБД. Обычно такой интерфейс представляет собой набор функций, вызываемых из клиентского приложения. Библиотеки, содержащие API для доступа к данным серверной СУБД, обычно входят в состав ее клиентского программного обеспечения, устанавливаемого на компьютерах, где функционируют клиентские приложения.

Использование клиентского API (или клиентских COM-объектов) является наиболее очевидным (и нередко самым эффективным с точки зрения производительности) способом манипуляции данными в приложении только СУБД этого производителя.

Другой способ манипуляции данными в приложении базируется на применении ***универсальных механизмов*** доступа к данным. Универсальный механизм доступа к данным обычно реализован в виде библиотек и дополнительных модулей, называемых ***драйверами*** или ***провайдерами*.** Библиотеки содержат некий стандартный набор функций или классов, нередко подчиняющийся той или иной спецификации. Дополнительные модули, специфичные для той или иной СУБД, реализуют непосредственное обращение к функциям клиентского API конкретных СУБД.

Достоинством универсальных механизмов является возможность применения одного и того же абстрактного API, а во многих случаях — COM-серверов, компонентов, классов для доступа к разным типам СУБД. Поэтому приложения, использующие универсальные механизмы доступа к данным, легко модифицировать, если необходима смена СУБД. При этом нередко модификация затрагивает не код приложения, а настройки доступа к данным, содержащиеся в реестре.

Наиболее популярные универсальные механизмы доступа к данным: Open Database Connectivity (ODBC), OLE DB, ActiveX Data Objects (ADO).

Универсальные механизмы ODBC, OLE DB и ADO фирмы Microsoft представляют собой по существу промышленные стандарты. ***В общем случае приложение может применять следующие механизмы доступа к ним***: непосредственный вызов функций клиентского API; вызов функций ODBC API; непосредственное обращение к интерфейсам OLEDB; применение ADO; применение ADO+OLEDB+ODBC.

ODBC

ODBC (Open Database Connectivity) — широко распространенный программный интерфейс фирмы Microsoft, удовлетворяющий стандартам ANSI и ISO для интерфейсов обращений к базам данных ***(Call Level Interface, CLI).*** Для доступа к данным конкретной СУБД с помощью ODBC, кроме собственно клиентской части этой СУБД, нужен ***ODBC Administrator*** (приложение, позволяющее определить, какие источники данных доступны для данного компьютера с помощью ODBC, и описать новые источники данных), и ***ODBC-драйвер*** для доступа к этой СУБД. ODBC-драйвер представляет собой динамически загружаемую библиотеку ***(DLL),*** которую клиентское приложение может загрузить в свое адресное пространство и использовать для доступа к источнику данных. Для каждой используемой СУБД нужен собственный ODBC-драйвер, так как ODBC-драйверы используют функции клиентских API, разные для различных СУБД.

С помощью ODBC можно манипулировать данными любой СУБД (и даже данными в файлах электронных таблиц или в текстовых файлах), если для них имеется ODBC-драйвер. Для манипуляции данными можно использовать как непосредственные вызовы ODBC API, так и другие универсальные механизмы доступа к данным, например OLE DB, ADO, реализующие стандартные функции или классы на основе вызовов ODBC API в драйверах или провайдерах, предназначенных для работы с любыми ODBC-источниками.

Спецификация ODBC подразумевает несколько стандартов на ODBC-драйверы (Level 1, Level 2 и т.д.). Эти стандарты отличаются различной функциональностью, которая должна быть реализована в таком драйвере. Например, драйверы, соответствующие стандарту Level 1, не обязаны поддерживать работу с хранимыми процедурами.

КЛАССИФИКАЦИЯ СТРУКТУРНЫХ МЕТОДОЛОГИЙ

Целью рассматриваемых методологий является преобразование общих, неясных знаний о требованиях к системе в точные (насколько это возможно) определения. Методологии фокусируют внимание на потоках данных, их главное назначение - создание базированных на графике документов по функциональным требованиям. Методологии поддерживаются традиционными нисходящими методами проектирования спецификаций и обеспечивают один из лучших способов связи между аналитиками, разработчиками и пользователями системы за счет интеграции множества различных средств.

Современные структурные методологии анализа и проектирования классифицируются по следующим признакам:

- ***по отношению к школам*** -Software Engineering (SE) и Information Engineering (IE);
- ***по порядку построения модели*** - процедурно-ориентированные, ориентированные на данные и информационно-ориентированные.

SE является нисходящим поэтапным подходом к разработке ПО, начинающейся с общего взгляда на его функционирование. Затем производится декомпозиция на подфункции, пока они не станут достаточно малы для их реализации кодированием. В ре-

зультате получается иерархическая, структурированная, модульная программа. SE является универсальной дисциплиной разработки ПО. Она успешно применяется ***как при разработке систем реального времени, так и при разработке информационных систем***.

IE - более новая дисциплина. С одной стороны, она имеет более широкую область применения, чем SE: ***IE является дисциплиной построения систем вообще*****,** и включает этапы более высокого уровня (стратегическое планирование), однако на этапе проектирования систем ПО эти дисциплины аналогичны. С другой стороны, IE - более узкая дисциплина, чем SE, т.к. IE используется только для построения информационных систем, а SE - для всех типов систем (таблица 6).

Таблица 6

Методологии	Школа	Порядок построения
Йодан-Де Марко Гейн-Сарсон	SE	Процедурно-ориентированная
Константайн	SE	Процедурно-ориентированная
Джексон	SE	Ориентированная на данные
Варнье-Орр	SE	Ориентированная на данные
Мартин	IE	Информационно-ориентированная
SADT	IE	Процедурно-ориентированная Ориентированная на данные

СОВРЕМЕННЫЕ МЕТОДОЛОГИИ ОПИСАНИЯ БИЗНЕС-ПРОЦЕССОВ

Oracle Designer. Компания Oracle http://www.oracle.com Представительство Oracle в России http://www.oracle.com/global/ru/index.html.

Функциональное средство для описания предметной области. Входит в комплекс инструментальных средств Oracle9i Developer Suite по проектированию программных систем и баз данных, реализующих технологию CASE и собственную методологию разработки ИС компании Oracle - "CDM", позволяющих команде разработчиков провести проект, начиная от анализа бизнес-процессов через моделирование к генерации кода и получению прототипа, а в дальнейшем и окончательного продукта.

Rational Rose. Компания IBM http://www.ibm.com Представительство IBM в России http://www.ibm.com/ru. Средство моделирования объектно-ориентированных информационных систем. Позволяет решать практически любые задачи в проектировании информационных систем: от анализа бизнес-процессов до кодогенерации на определенном языке программирования. Позволяет разрабатывать как высокоуровневые, так и низкоуровневые модели, осуществляя тем самым либо абстрактное проектирование, либо логическое.

System Architect. Компания Telelogic. http://www.telelogic.com. Компания Telelogic в России http://www.telelogic.com. System Architect представляет собой универсальное CASE-средство, позволяющее осуществить не только проектирование данных, но и структурное моделирование.

Power Designer. Компания Sybase http://www.sybase.com. Компания Sybase http://www.sybase.ru. PowerDesigner - средство моделирования бизнес-процессов, проектирования баз данных и объектного моделирования.

Re-Think. Компания Gensym. http://www.gensym.com. Графическая объектно-ориентированная среда создания и сопровождения интеллектуальных приложений мониторинга, диагностики и управления сложными динамическими системами.

Ithink Analyst. Компания High Performance Systems http://www.hps-inc.com. Компания Тора-центр http://www.tora-centre.ru. Пакет для ситуационного моделирования. Позволяет строить наглядные и точные модели самых сложных политических и экономических ситуаций, используя библиотеку базовых моделей и методы системной динамики. Также используется при анализе инвестиционных проектов и реинжиниринге.

Workflow Modeler. Компания Meta Software http://www.metasoftware.com. Пакет для функционального и информационного моделирования, анализа и проектирования биз-

нес-процессов. Используется в некоторых известных пакетах типа CIM (Computer Integrated Manufacturing) и CAE (Computer Aided Engineering) и принят в качестве стандарта для проектов, финансируемых американскими и европейскими спонсорами.

АНАЛИЗ СТРУКТУРНЫХ И ОБЪЕКТНО – ОРИЕНТИРОВАННЫХ МЕТОДОЛОГИЙ

Появление CASE-технологий способствовало бурному развитию моделирования систем в двух различных направлениях: структурном и объектном.

Моделирование бизнес-процессов может быть реализовано в рамках различных методологий, отличающихся своим подходом к тому, что представляет собой моделируемая организация. *В соответствии с различными представлениями об организации методологии принято делить на объектные и функциональные (структурные).*

Объектные методологии рассматривают моделируемую организацию как набор взаимодействующих объектов - производственных единиц. Объект определяется как осязаемая реальность - предмет или явление, имеющее четко определяемое поведение. Целью применения данной методологии является выделение объектов, составляющих организацию, и распределение между ними ответственностей за выполняемые действия.

Структурные методологии 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 от самих данных.

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.

Структурный подход хорошо показывает себя в тех случаях, когда организационная структура находится в процессе изменения или вообще слабо оформлена. Подход от выполняемых функций интуитивно лучше понимается исполнителями при получении от них информации об их текущей работе.

СРАВНЕНИЕ СУЩЕСТВУЮЩИХ МЕТОДОЛОГИЙ

В функциональных моделях (DFD-диаграммах потоков данных, SADT-диаграммах) главными структурными компонентами являются функции (процессы, действия, работы), которые на диаграммах связываются между собой потоками данных.

К преимуществам структурного подхода относятся: возможность однозначно определить внешние сущности; возможность проектирования сверху вниз, что облегчает построение модели "как должно быть"; наличие спецификаций процессов нижнего уровня, что позволяет построить полную функциональную спецификацию разрабатываемой системы; наглядности представления модели.

К недостаткам структурного подхода относятся: искусственный ввод управляющих процессов; процессы и данные существуют отдельно друг от друга.

Объектно-ориентированный подход обладает следующими преимуществами:

- объектная декомпозиция дает возможность создавать модели меньшего размера путем использования общих механизмов, обеспечивающих необходимую экономию выразительных средств. Использование объектного подхода существенно повышает уровень унификации разработки и пригодность для повторного использования;
- объектная декомпозиция позволяет избежать создания сложных моделей, так как она предполагает эволюционный путь развития модели на базе небольших подсистем;
- объектная модель естественна, ориентированна на человеческое восприятие.

К недостаткам объектно-ориентированного подхода относятся: высокие начальные затраты. Этот подход не дает немедленной отдачи. Эффект от его применения сказывается после разработки двух-трех проектов и накопления повторно используемых компонентов; диаграммы, отражающие специфику объектного подхода, менее наглядны.

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 ИС для различных классов задач могут требоваться различные виды моделей, описывающих одну и ту же проблемную область. В таком случае необходимо использовать комбинированные модели предметной области.

СИНТЕТИЧЕСКАЯ МЕТОДОЛОГИЯ

Из представленного обзора видно, что каждая из рассмотренных методологий позволяет решить задачу построения формального описания рабочих процедур исследуемой системы. Все методологии позволяют построить модель "как есть" и "как должно быть". С другой стороны, каждая из этих методологий обладает существенными недостатками. Их можно суммировать следующим образом: недостатки применения отдельной методологии лежат не в области описания реальных процессов, а в неполноте методического подхода.

Функциональные методологии в целом лучше дают представление о существующих функциях в организации, о методах их реализации, причем, чем выше степень детализации исследуемого процесса, тем лучше они позволяют описать систему. Под лучшим описанием понимается наименьшая ошибка при предсказании поведения реальной системы. На уровне отдельных процедур - описание практически однозначно совпадает с фактической реализацией в потоке работ.

Хорошо описать систему на этом уровне позволяет объектный подход, основанный на понятии сценария использования как о сеансе взаимодействия действующего лица с системой, в результате которого действующее лицо получает нечто, имеющее для него ценность. Использование критерия ценности для пользователя дает возможность отбросить не имеющие значения детали потоков работ и сосредоточиться на тех функциях системы, которые оправдывают ее существование. Однако и в этом случае задача определения границ системы, выделения внешних пользователей является сложной.

Наилучшим способом преодоления недостатков рассмотренных методологий является формирование синергетической методологии, объединяющей различные этапы отдельных методологий. При этом из каждой методологии необходимо взять часть, наиболее полно и формально изложенную, и обеспечить возможность обмена результатами на различных этапах применения синергетической методологии. Идея синтетической методологии заключается в последовательном применении функционального и объектного подхода с учетом возможности реинжиниринга существующей ситуации.

Пример. Рассмотрим применение синтетической методологии на примере разработки административного регламента. При этом выделяются следующие стадии:

1. Определение границ системы. При помощи анализа потоков данных выделяют внешние сущности и собственно моделируемую систему.
2. Выделение сценариев использования системы. При помощи критерия полезности строят для каждой внешней сущности набор сценариев использования системы.
3. Добавление системных сценариев использования. Определяют сценарии, необходимые для реализации целей системы, отличных от целей пользователей.
4. Построение диаграммы активностей по сценариям использования. Строят набор действий системы, приводящих к реализации сценариев использования;
5. Функциональная декомпозиция диаграмм активностей как контекстных диаграмм методологии IDEF0.
6. Формальное описание отдельных функциональных активностей в виде административного регламента (с применением различных нотаций).

ТЕХНОЛОГИИ И СРЕДСТВА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

ТРЕБОВАНИЯ К ТЕХНОЛОГИЯМ

Технология проектирования определяется совокупностью трех составляющих:

- ***пошаговой процедуры***, *определяющей последовательность технологических операций проектирования (рисунок 25);*

- ***критериев и правил***, *используемых для оценки результатов выполнения технологических операций;*

- ***нотаций*** *(графических и текстовых средств), используемых для описания проектируемой системы.*

Технология проектирования, разработки и сопровождения ***ИС должна удовлетворять следующим общим требованиям:***

- ***технология должна поддерживать полный ЖЦ ПО***;

- ***технология должна обеспечивать гарантированное достижение целей разработки ИС с заданным качеством и в установленное время;***

- ***технология должна обеспечивать возможность выполнения крупных проектов в виде подсистем;***

- ***технология должна обеспечивать возможность ведения работ по проектированию отдельных подсистем небольшими группами (3-7 человек);***

- ***технология должна обеспечивать минимальное время получения работоспособной ИС;***

- ***технология должна предусматривать возможность управления конфигурацией проекта;***

- ***технология должна обеспечивать независимость выполняемых проектных решений от средств реализации ИС;***

- ***технология должна быть поддержана комплексом согласованных CASE-средств.***

Рисунок 25 - Представление технологической операции проектирования

Реальное применение любой технологии проектирования, разработки и сопровождения ИС в конкретной организации и проекте невозможно без выработки ряда стандартов (правил, соглашений), которые должны соблюдаться всеми участниками проекта.

К таким стандартам относятся: ***стандарт проектирования; стандарт оформления проектной документации; стандарт пользовательского интерфейса***.

Стандарт проектирования должен устанавливать:

- ***набор необходимых моделей*** (диаграмм) на каждой стадии проектирования и степень их детализации;

- ***правила фиксации проектных решений*** на диаграммах: правила именования объектов, набор атрибутов для всех объектов и правила их заполнения;

- ***требования к конфигурации*** рабочих мест разработчиков, включая настройки операционной системы, настройки CASE-средств, общие настройки проекта и т. д.;

- ***механизм обеспечения*** совместной работы над проектом, в том числе: правила интеграции подсистем проекта, правила поддержания проекта в одинаковом для всех разработчиков состоянии, правила проверки проектных решений на непротиворечивость.

Стандарт оформления проектной документации должен устанавливать: ***комплектность***, состав и структуру на каждой стадии проектирования; ***требования к ее оформлению*** (разделов, подразделов, пунктов, таблиц); ***требования к настройке CASE-средств*** для обеспечения подготовки документации в соответствии с установленными требованиями.

Стандарт интерфейса пользователя должен устанавливать: ***правила оформления экранов***, состав, расположение окон, элементов управления; правила ***использования клавиатуры и мыши***; правила ***оформления текстов*** помощи; ***перечень стандартных сообщений***; правила ***обработки реакции пользователя***.

Технология создания крупных информационных систем (ИС) предъявляет особые ***требования к методикам реализации и программным инструментальным средствам***:

- ***реализацию крупных проектов принято разбивать на стадии анализа*** (прежде чем создавать ИС необходимо понять и описать бизнес-логику предметной области), ***проектирования*** (необходимо определить модули и архитектуру будущей системы), ***кодирования, тестирования и сопровождения.*** Исправление ошибок, допущенных на предыдущей стадии, обходится примерно в десять раз дороже, чем на текущей, откуда следует, что наиболее критичными являются первые стадии проекта. ***Поэтому важно иметь эффективные средства автоматизации ранних этапов реализации проекта;***

- ***крупный проект невозможно реализовать в одиночку***. ***Необходимо иметь средства координации и управления коллективом разработчиков;***

- ***жизненный цикл создания сложной ИС сопоставим с ожидаемым временем ее эксплуатации.*** Может оказаться, что к моменту сдачи ИС она уже никому не нужна, поскольку компания, ее заказавшая, вынуждена перейти на новую технологию работы. Следовательно, ***для создания крупной ИС необходимы инструментальные средства, значительно уменьшающие время разработки ИС и которые были бы достаточно гибкими к изменяющимся требованиям***.

CASE-технология представляет собой методы проектирования ИС, набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать модель на всех этапах разработки и сопровождения ИС и разрабатывать приложения в соответствии с информационными потребностями пользователей.

CASE-средства имеют характерные особенности:

- ***наличие мощных графических средств***, обеспечивающих интерфейс с разработчиком, позволяющих разработчикам в наглядном виде изучать существующую ИС, перестраивать ее в соответствии с поставленными целями и имеющимися ограничениями;

- ***интеграция отдельных компонентов CASE-средств***, обеспечивающая управляемость процессом разработки ИС;

- ***использование*** специальным образом организованного ***хранилища проектных метаданных (репозитория).***

ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ DFD

Практически любой класс систем успешно моделируется при помощи DFD-ориентированных технологий и методов: в этом случае вместо реальных объектов рассматриваются отношения, описывающие свойства этих объектов и их поведение.

Примерами таких систем служат системы документооборота, управления и другие системы. DFD с самого начала создавались как средство проектирования ИС.

Наличие миниспецификаций DFD-процессов нижнего уровня позволяет преодолеть логическую незавершенность и построить полную функциональную спецификацию разрабатываемой системы. ***DFD, ERD и STD взаимно дополняют друг друга*** и, по сути, являются согласованными представлениями различных аспектов одной и той же модели

Интеграция DFD-ERD осуществляется с использованием объекта - хранилища данных, структура которого описывается с помощью ERD и согласуется по соответствующим потокам и другим хранилищам на DFD.

DFD могут быть легко преобразованы в модели проектирования (структурные карты) - это близкие модели, что обеспечивает логичный и безболезненный переход от этапа анализа требований к проектированию системы.

Для разработки информационной системы и ПО используется технология последовательно связанных между собой методологий и программных продуктов (рисунок 26).

Рисунок 26 – Последовательность разработки ИС

МОДЕЛЬ ПОТОКОВ ДАННЫХ DFD

Последовательность построения модели потоков данных:

1 Расчленение множества требований к ИС и организация их в основные функциональные группы.
2 Идентификация внешних объектов, с которыми система должна быть связана.
3 Идентификация информации между системой и внешними объектами.
4 Построение контекстной диаграммы путем объединения функциональных групп в один процесс.
5 Декомпозиция и формирование DFD первого уровня.
6 Проверка основных требований по DFD первого уровня.
7 Декомпозиция каждого процесса текущей DFD с помощью детализирующей диаграммы с добавлением хранилищ данных.
8 Проверка основных требований по DFD соответствующего уровня.
9 Добавление определений новых потоков в словарь данных при каждом их появлении на диаграммах в хранилищах данных.
10 Параллельное (с декомпозицией) изучение требований (в том числе и вновь поступающих), идентификация процессов, соответствующих этим требованиям.
11 Проведение ревизии с целью проверки корректности и улучшения модели.
12 Построение спецификаций процессов.

МОДЕЛЬ БАНКОМАТА

Рассмотрим фрагмент проекта системы, организующей работу банкомата по обслуживанию клиента по его кредитной карте.

Контекстная диаграмма

Основные функции банкомата можно представить в виде процессов:

Процесс 1.1 (***ПОЛУЧИТЬ ПАРОЛЬ***).

Процесс 1.2 (***ПОЛУЧИТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ***).

Процесс 1.3 *(**ОБРАБОТАТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ**)*.

Процесс 1.4 (***ОБРАБОТАТЬ КРЕДИТНУЮ КАРТУ***).

Создадим контекстную диаграмму. На рисунке 27 приведена контекстная диаграмма системы с единственным процессом ***ОБСЛУЖИТЬ*****,** идентифицирующим внешние сущности ***КЛИЕНТ*** и ***КОМПЬЮТЕР БАНКА***. Опишем потоки данных, которыми обменивается проектируемая система с внешними объектами.

Для банковского обслуживания клиенту необходимо предоставить системе свою ***КРЕДИТНУЮ КАРТУ*** для автоматического считывания с нее информации *(ПАРОЛЬ, ЛИМИТ ДЕНЕГ, ДЕТАЛИ КЛИЕНТА),* а также сообщить свои ***КЛЮЧЕВЫЕ ДАННЫЕ,*** а именно *ПАРОЛЬ и ЗАПРОС НА ОБСЛУЖИВАНИЕ,* т.е. требуемую ему услугу (снятие со счета наличных денег).

Банковское обслуживание с позиций клиента, в свою очередь, должно:

- выдать ***СООБЩЕНИЕ***, приглашающее клиента ввести *КЛЮЧЕВЫЕ ДАННЫЕ*;
- выдать клиенту ***ДЕНЬГИ***;
- выдать клиенту ***ВЫПИСКУ*** по проведенному обслуживанию, включающую *ВЫПИСКУ О ДЕНЬГАХ, ВЫПИСКУ ПО БАЛАНСУ* и *ВЫПИСКУ ПО ОПЕРАЦИИ*.

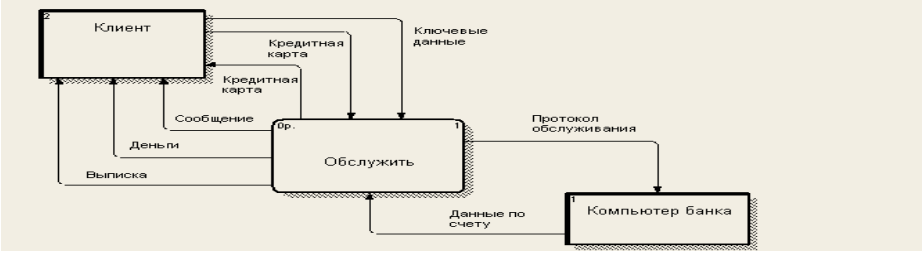


Рисунок 27 - Контекстная диаграмма банковской задачи

Контекстный процесс и *КОМПЬЮТЕР БАНКА* обмениваются информацией:

- ***ДАННЫЕ ПО СЧЕТУ*** клиента в банке*;*
- ***ПРОТОКОЛ ОБСЛУЖИВАНИЯ***, включающей информацию об *ОБРАБОТАННОЙ ДОКУМЕНТАЦИИ*, изымаемой *ДЕНЕЖНОЙ СУММЕ и ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА*.

DFD первого уровня

Контекстный процесс может быть детализирован DFD первого уровня, как показано на рисунке 28. Эта диаграмма содержит 4 процесса и хранилище ***ДАННЫЕ КРЕДИТНОЙ КАРТЫ***, которое изображено дважды на диаграмме для удобства.

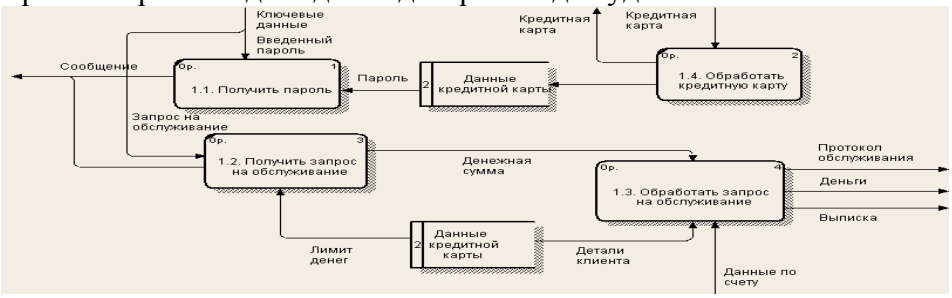


Рисунок 28 - Детализация процесса *ОБСЛУЖИТЬ*

Процесс 1.1 (***ПОЛУЧИТЬ ПАРОЛЬ***) осуществляет прием и проверку пароля клиента и имеет на входе/выходе следующие потоки: внешний выходной поток - *СООБЩЕНИЕ* для информирования клиента о своей готовности принять пароль; входной поток - *ВВЕДЕННЫЙ ПАРОЛЬ* как элемент внешнего потока *КЛЮЧЕВЫЕ ДАННЫЕ*; входной поток - *ПАРОЛЬ* из хранилища *ДАННЫЕ КРЕДИТНОЙ КАРТЫ* для проверки вводимого клиентом пароля.

Процесс 1.2 (***ПОЛУЧИТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ***) осуществляет прием и проверку запроса клиента на проведение необходимой ему банковской операции и имеет на входе/выходе потоки: внешний выходной поток *СООБЩЕНИЕ* для информирования клиента о готовности принять запрос на обслуживание; входной поток *ЗАПРОС НА ОБСЛУЖИВАНИЕ* как элемент внешнего потока *КЛЮЧЕВЫЕ ДАННЫЕ*; входной поток *ЛИМИТ ДЕНЕГ* из хранилища *ДАННЫЕ КРЕДИТНОЙ КАРТЫ* для контроля наличия денег на счете клиента.

Процесс 1.3 (ОБРАБОТАТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ) имеет внешний входной поток *ДАННЫЕ ПО СЧЕТУ* (из внешней сущности *КОМПЬЮТЕР БАНКА*), входной поток *ДЕТАЛИ КЛИЕНТА* (из хранилища), а также внешние выходные потоки *ВЫПИСКА, ДЕНЬГИ и ПРОТОКОЛ ОБСЛУЖИВАНИЯ*

Процесс 1.4 (***ОБРАБОТАТЬ КРЕДИТНУЮ КАРТУ***) осуществляет считывание информации с кредитной карты и имеет на входе внешний поток *КРЕДИТНАЯ КАРТА*, а на выходе поток *ДАННЫЕ КРЕДИТНОЙ КАРТЫ*. Нет необходимости в идентификации потока, т.к. идентифицировано соответствующее хранилище.

DFD второго уровня

Процессы 1.1, 1.2 и 1.4 являются элементарными (они будут раскрыты с помощью спецификаций процессов). Процесс 1.3 может быть детализирован с помощью DFD второго уровня. Диаграмма содержит 4 процесса (рисунок 29).

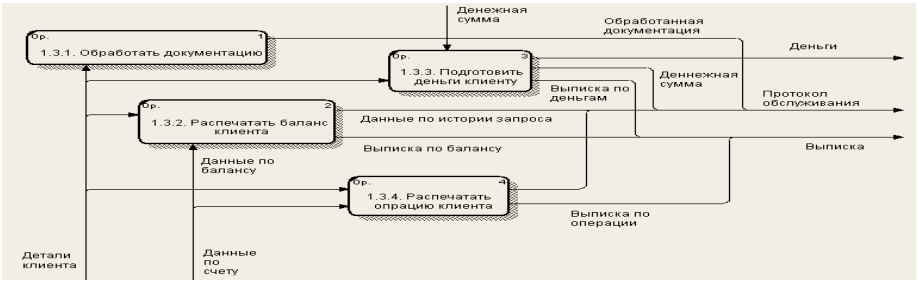


Рисунок 29 - Детализация процесса *ОБРАБОТАТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ*

Процесс 1.3.1 (***ОБРАБОТАТЬ ДОКУМЕНТАЦИЮ БАНКА***) осуществляет обработку внутренней банковской документации по клиенту и имеет входной поток *ДЕТАЛИ КЛИЕНТА* и выходной поток *ОБРАБОТАННАЯ ДОКУМЕНТАЦИЯ* (часть потока *ПРОТОКОЛ СДЕЛКИ*).

Процесс 1.3.2 (***РАСПЕЧАТАТЬ БАЛАНС КЛИЕНТА***) выдает справку по истории счета клиента и по балансу клиента. Входные потоки - *ДЕТАЛИ КЛИЕНТА* и *ДАННЫЕ ПО БАЛАНСУ* (часть внешнего потока *ДАННЫЕ ПО СЧЕТУ*), выходные потоки - *ВЫПИСКА ПО БАЛАНСУ* (часть внешнего потока *ВЫПИСКА*) и *ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА* (часть внешнего потока *ПРОТОКОЛ ОБСЛУЖИВАНИЯ*).

Процесс 1.3.3 (***ПРИГОТОВИТЬ ДЕНЬГИ КЛИЕНТУ***) обеспечивает выдачу наличных денег и информирование компьютера банка об изъятых из банка деньгах. Он имеет входные потоки *ДЕНЕЖНАЯ СУММА* и *ДЕТАЛИ КЛИЕНТА*, и выходные потоки *ДЕНЬГИ* и *ДЕНЕЖНАЯ СУММА (часть потока ПРОТОКОЛ ОБСЛУЖИВАНИЯ*).

Процесс 1.3.4 (***РАСПЕЧАТАТЬ ОПЕРАЦИЮ КЛИЕНТА***) выдает справку по истории счета и уведомление по проведенной операции. Входные потоки: *ДАННЫЕ ПО СЧЕТУ* и *ДЕТАЛИ КЛИЕНТА*, выходные потоки - *ВЫПИСКА ПО ОПЕРАЦИИ* (часть потока *ВЫПИСКА*) и *ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА* (часть потока *ПРОТОКОЛ ОБСЛУЖИВАНИЯ*).

Управляющий процесс

Для управления процессом вводится управляющий процесс и управляющие потоки, позволяющие описать ее функционирование в реальном времени (рисунок 30).

Управляющий процесс (***УПРАВЛЕНИЕ ОБСЛУЖИВАНИЕМ***), получив информацию о том, что кредитная карта введена (поток *ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА*), вызывает выполнение процесса 1.1 (поток *ПОЛУЧИТЬ ПАРОЛЬ*). Получив информацию о введенном пароле (поток *КОРРЕКТНЫЙ ПАРОЛЬ*), управляющий процесс информирует процесс 1.4 о необходимости удаления кредитной карты (поток *УДАЛЕННАЯ КРЕДИТНАЯ КАРТА*) и с помощью потока *ОБЕСПЕЧИТЬ ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ* вызывает выполнение процесса 1.2**,** затем процесса 1.3 (поток *ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ*).

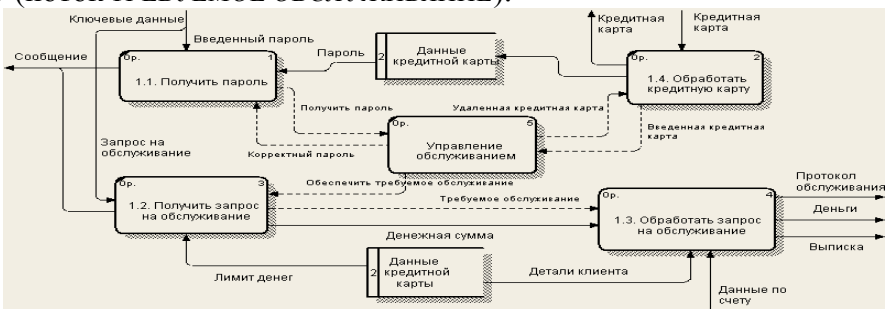


Рисунок 30 - Расширение диаграммы, детализирующей контекстный процесс

Спецификации процессов для примера банковской задачи

Спецификация процесса 1.1

@ВХОД *= ВВЕДЕННЫЙ ПАРОЛЬ*
@ВХОД *= ПАРОЛЬ*
@ВЫХОД *= СООБЩЕНИЕ*
@ВЫХОД *= КОРРЕКТНЫЙ ПАРОЛЬ*
@СПЕЦПРОЦ *1.1* ***ПОЛУЧИТЬ ПАРОЛЬ***
ВЫПОЛНИТЬ *выдать СООБЩЕНИЕ клиенту, запрашивающее ввод пароля*
Принять ВВЕДЕННЫЙ ПАРОЛЬ
ДОТЕХПОРПОКА *ВВЕДЕННЫЙ ПАРОЛЬ = ПАРОЛЬ или были сделаны три попытки ввода*

КОНЕЦВЫПОЛНИТЬ
ВЫПОЛНИТЬ *установить флаг КОРРЕКТНЫЙ ПАРОЛЬ в случае равенства*
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.1

Спецификация процесса 1.2

@ВХОД *= ЛИМИТ ДЕНЕГ*
@ВХОД *= ЗАПРОС НА ОБСЛУЖИВАНИЕ*
@ВЫХОД *= ДЕНЕЖНАЯ СУММА*
@ВЫХОД *= СООБЩЕНИЕ*
@ВЫХОД *= ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ*
@СПЕЦПРОЦ *1.2* ***ПОЛУЧИТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ***
ВЫПОЛНИТЬ *выдать СООБЩЕНИЕ клиенту по вводу запроса на обслуживание*
Принять ЗАПРОС НА ОБСЛУЖИВАНИЕ Обновить данные ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ (а именно, ЗАПРОС ДОКУМЕНТАЦИИ, ЗАПРОС ДЕНЕГ,ЗАПРОС БАЛАНСА, ЗАПРОС НА ОПЕРАЦИЮ)
ЕСЛИ *был сделан ЗАПРОС ДЕНЕГ*
ТО *ВЫПОЛНИТЬ запросить ДЕНЕЖНУЮ СУММУ*
Выдать требуемую ДЕНЕЖНУЮ СУММУ с учетом того, что она не должно превышать ЛИМИТ ДЕНЕГ
КОНЕЦЕСЛИ
ДОТЕХПОРПОКА *запрашивается продолжение обслуживания*
КОНЕЦВЫПОЛНИТЬ
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.2

Спецификация процесса 1.3

Спецификация процесса 1.3.1

@ВХОД *= ЗАПРОС ДОКУМЕНТАЦИИ*
@ВХОД *= ДЕТАЛИ КЛИЕНТА*
@ВЫХОД *= ОБРАБОТАННАЯ ДОКУМЕНТАЦИЯ*
@СПЕЦПРОЦ *1.3.1* ***ОБРАБОТАТЬ ДОКУМЕНТАЦИЮ БАНКА***
По получении ЗАПРОСА ДОКУМЕНТАЦИИ выдать ОБРАБОТАННУЮ ДОКУМЕНТАЦИЮ, содержащую ДЕТАЛИ КЛИЕНТА, КОМПЬЮТЕРУ БАНКА
@КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.3.1

Спецификация процесса 1.3.2

@ВХОД *= ДАННЫЕ ПО БАЛАНСУ*
@ВХОД *= ЗАПРОС БАЛАНСА*
@ВХОД *= ДЕТАЛИ КЛИЕНТА*
@ВЫХОД *= ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА*
@ВЫХОД *= ВЫПИСКА ПО БАЛАНСУ*
@СПЕЦПРОЦ *1.3.2* ***РАСПЕЧАТАТЬ БАЛАНС КЛИЕНТА***
По получении ЗАПРОСА БАЛАНСА выдать ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА
Затем выдать ВЫПИСКУ ПО БАЛАНСУ, содержащую ДАННЫЕ ПО БАЛАНСУ
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.3.2

Спецификация процесса 1.3.3

@ВХОД *= ДЕНЕЖНАЯ СУММА*
@ВХОД *= ЗАПРОС ДЕНЕГ*
@ВХОД *= ДЕТАЛИ КЛИЕНТА*
@ВЫХОД *= ДЕНЬГИ*
@ВЫХОД *= ВЫПИСКА О ДЕНЬГАХ*
@ВЫХОД *= ДЕНЕЖНАЯ СУММА*
@СПЕЦПРОЦ *1.3.3* ***ПРИГОТОВИТЬ ДЕНЬГИ ДЛЯ КЛИЕНТА***
По получении ЗАПРОСА ДЕНЕГ выдать ДЕНЬГИ по значению ДЕНЕЖНОЙ СУММЫ
Выдать ВЫПИСКУ О ДЕНЬГАХ, содержащую ДЕНЕЖНУЮ СУММУ
Передать КОМПЬЮТЕРУ БАНКА информацию о ДЕНЕЖНОЙ СУММЕ
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.3.3

Спецификация процесса 1.3.4

@ВХОД *= ДАННЫЕ ПО СЧЕТУ*
@ВХОД *= ЗАПРОС НА ОПЕРАЦИЮ*
@ВХОД *= ДЕТАЛИ КЛИЕНТА*
@ВЫХОД *= ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА*
@ВЫХОД *= ВЫПИСКА ПО ОПЕРАЦИИ*
@СПЕЦПРОЦ *1.3.4* ***РАСПЕЧАТАТЬ ОПЕРАЦИЮ КЛИЕНТА***
При получении ЗАПРОСА НА ОПЕРАЦИЮ выдать ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА для специфицирования ДЕТАЛЕЙ КЛИЕНТА, чтобы получить текущие ДАННЫЕ ПО СЧЕТУ
Выдать ВЫПИСКУ ПО ОПЕРАЦИИ, содержащую ДАННЫЕ ПО СЧЕТУ
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.3.4

Спецификация процесса 1.4.

@ВХОД *= УДАЛЕННАЯ КРЕДИТНАЯ КАРТА*
@ВХОДВЫХОД *= КРЕДИТНАЯ КАРТА*
@ВЫХОД *= ДАННЫЕ КРЕДИТНОЙ КАРТЫ*
@ВЫХОД *= ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА*
@СПЕЦПРОЦ *1.4* ***ОБРАБОТАТЬ КРЕДИТНУЮ КАРТУ***
ВЫПОЛНИТЬ считать КРЕДИТНУЮ КАРТУ записать в хранилище ДАННЫЕ КРЕДИТНОЙ КАРТЫ
Выдать управляющий поток ВВЕДЕННАЯ КРЕДИТНАЯ КАРТА
По получении управляющего потока УДАЛЕННАЯ КРЕДИТНАЯ КАРТА удалить КРЕДИТНУЮ КАРТУ
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1.4

МОДЕЛЬ “СУЩНОСТЬ-СВЯЗЬ”

Разработка ERD включает следующие основные этапы:

1. Анализ хранилищ данных, идентификация сущностей, их атрибутов, а также первичных и альтернативных ключей.
2. Идентификация отношений между сущностями и указание типов отношений.
3. Построение модели.

Банковская задача имеет одно хранилище данных ***ДАННЫЕ КРЕДИТНОЙ КАРТЫ*** с потоками данных *ПАРОЛЬ, ДЕТАЛИ КЛИЕНТА, ЛИМИТ ДЕНЕГ* .

Идентифицируем сущности: сущность ***КРЕДИТНАЯ КАРТА*** и внешняя сущность – ***КЛИЕНТ.*** На рисунке 31 приведен фрагмент ERD банковской задачи в нотации Баркера.

Рисунок 31 - Нотация Баркера

МОДЕЛЬ STD

Диаграмма STD для банковской задачи имеет вид (рисунок 32).

Рисунок 32 - Диаграмма STD

МОДЕЛЬ СТРУКТУРНЫХ КАРТ

Пример **структурной карты Константайна**, описывающей межмодульные отношения в рассмотренном ранее фрагменте банковской системы (рисунок 33):

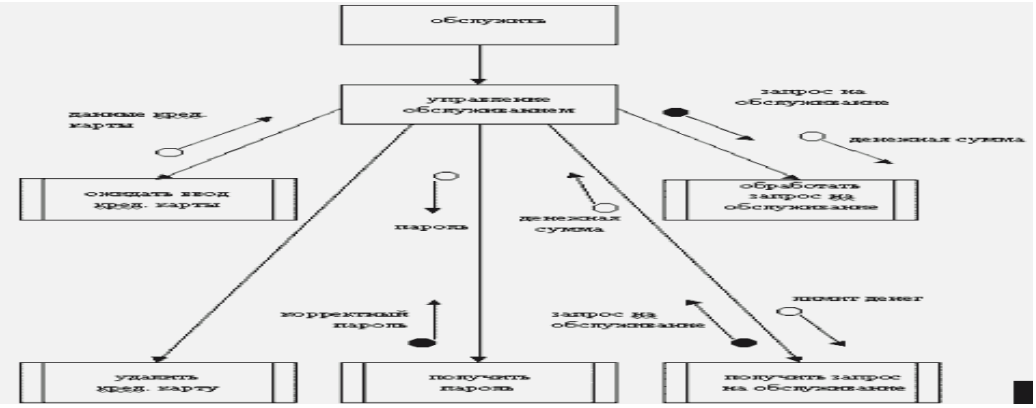


Рисунок 33 - Пример структурной карты Константайна

ОБОБЩЕННАЯ МОДЕЛЬ

Модель представляет совокупность программных модулей, связанных между собой и внешними сущностями через базу данных. Каждый модуль реализует определенную функцию (процесс) или группу функций. Обобщенная модель банковской задачи приведена на рисунке 34. Она содержит два модуля: модуль обработки кредитной карты и модуль обработки запроса на обслуживание. Модули связаны с БД – Данные кредитной карты и с внешними сущностями: Клиент и Компьютер банка.

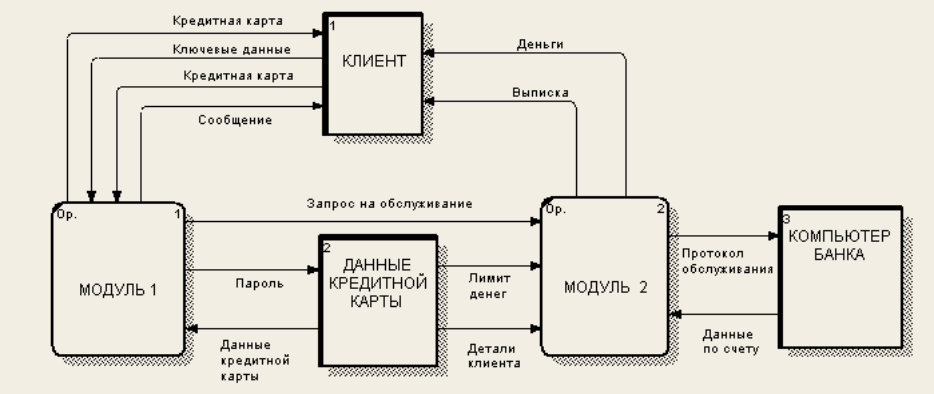


Рисунок 34 – Обобщенная модель

МОДУЛИ

Модуль 1 – ОБРАБОТАТЬ КРЕДИТНУЮ КАРТУ

ВХОДНЫЕ ДАННЫЕ*: ВВЕДЕННЫЙ ПАРОЛЬ (ДАННЫЕ КРЕДИТНОЙ КАРТЫ), ПАРОЛЬ, ДАННЫЕ КРЕДИТНОЙ КАРТЫ.*

ВЫХОДНЫЕ ДАННЫЕ*: СООБЩЕНИЕ, ДАННЫЕ КРЕДИТНОЙ КАРТЫ.*

ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ*:*

Выдать СООБЩЕНИЕ клиенту, запрашивающее ввод пароля.
Принять ВВЕДЕННЫЙ ПАРОЛЬ проверить ВВЕДЕННЫЙ ПАРОЛЬ = ПАРОЛЬ из БД.
Считать КРЕДИТНУЮ КАРТУ, записать в хранилище ДАННЫЕ КРЕДИТНОЙ КАРТЫ.

Модуль 2 – ОБРАБОТАТЬ ЗАПРОС НА ОБСЛУЖИВАНИЕ

ВХОДНЫЕ ДАННЫЕ*: ЗАПРОС НА ОБСЛУЖИВАНИЕ, ЛИМИТ ДЕНЕГ, ДАННЫЕ ПО СЧЕТУ, ДЕТАЛИ КЛИЕНТА.*

ВЫХОДНЫЕ ДАННЫЕ*: СООБЩЕНИЕ, ДЕНЕЖНАЯ СУММА, ВЫПИСКА ПО БАЛАНСУ, ВЫПИСКА О ДЕНЬГАХ.*

ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ*:*

Выдать СООБЩЕНИЕ клиенту по вводу запроса на обслуживание.
Принять ЗАПРОС НА ОБСЛУЖИВАНИЕ.
Обновить данные ТРЕБУЕМОЕ ОБСЛУЖИВАНИЕ (а именно, ЗАПРОС ДОКУМЕНТАЦИИ, ЗАПРОС ДЕНЕГ, ЗАПРОС БАЛАНСА, ЗАПРОС НА ОПЕРАЦИЮ).
Если был сделан ЗАПРОС ДЕНЕГ то запросить ДЕНЕЖНУЮ СУММУ.
Выдать требуемую ДЕНЕЖНУЮ СУММУ с учетом того, что она не должно превышать ЛИМИТ ДЕНЕГ.
По получении ЗАПРОСА ДОКУМЕНТАЦИИ выдать ОБРАБОТАННУЮ ДОКУМЕНТАЦИЮ, содержащую ДЕТАЛИ КЛИЕНТА КОМПЬЮТЕРУ БАНКА.
По получении ЗАПРОСА БАЛАНСА выдать ДАННЫЕ ПО ИСТОРИИ ЗАПРОСА.
Затем выдать ВЫПИСКУ ПО БАЛАНСУ, содержащую ДАННЫЕ ПО БАЛАНСУ.
По получении ЗАПРОСА ДЕНЕГ выдать ДЕНЬГИ по значению ДЕНЕЖНОЙ СУММЫ.
Выдать ВЫПИСКУ О ДЕНЬГАХ, содержащую ДЕНЕЖНУЮ СУММУ.
Передать КОМПЬЮТЕРУ БАНКА информацию о ДЕНЕЖНОЙ СУММЕ.

ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ SADT

AllFusion Process Modeler является частью полного комплекта продуктов CA, интегрирующихся друг с другом с формированием законченного сквозного решения разработки и управления данными (рисунок 35).

Широко используются три основных вида моделирования процессов:

- моделирование бизнес-процессов или функциональное моделирование (IDEF0);
- моделирование потоков процессов (IDEF3);
- моделирование потоков данных (DFD).

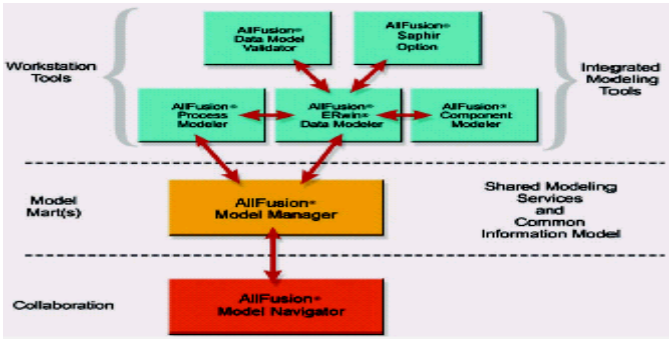


Рисунок 35 - Инструменты AllFusion Modeling Suite

Можно экспортировать информацию из AllFusion Process Modeler в AllFusion ERwin Data Modeler, с возможностью дальнейшего обновления имеющихся баз данных. Таким же образом, данные из AllFusion ERwin Data Modeler могут быть импортированы в AllFusion Process Modeler и привязаны к потокам и операциям моделей процессов.

AllFusion Process Modeler также интегрируется с AllFusion Model Manager, что улучшает координацию разработки и сопровождения крупных моделей несколькими сотрудниками, обеспечивает контроль версий, разрешение конфликтов, защиту информации и соблюдение стандартов. AllFusion Model Manager, в свою очередь, интегрируется с AllFusion Model Navigator, что позволяет многим сотрудникам компании просматривать модели с формированием единого корпоративного понимания бизнес-процессов.

С помощью разнообразных интерфейсов, AllFusion Process Modeler также обеспечивает поддержку сред моделирования, а также систем управления и отслеживания требований. Подобная расширяемость позволяет получать подробную информацию о процессах и данных из инструментов моделирования AllFusion.

ТЕХНОЛОГИЯ СОЗДАНИЯ ИНФОРМАЦИОННОЙ МОДЕЛИ

ОПИСАНИЕ ТЕХНОЛОГИИ

В описываемой технологии предлагается использование связки нескольких систем. Далее приведены этапы предлагаемой технологии структурирования требований.

Построение процессной модели (IDEF0/DFD/IDEF3)

На первом этапе производится формирование процессной модели системы (методология IDEF0/DFD/IDEF3) с использованием программного продукта AllFusion Process Modeler. По мере уточнения требований производится декомпозиция функций системы до необходимого уровня.

Построение ER модели (IDEF1X)

В рамках сбора требований к ПО производится определение требований к информации, которую будет обрабатывать система. Данные требования обычно включают в себя описание видов и атрибутики обрабатываемой информации.

Технология предполагает разработку связной ER модели сущностей предметной области на этапе сбора требований. ER модель разрабатывается с использование программного продукта AllFusion Data Modeler (ERwin).

Связывание процессной и ER моделей

Между процессной моделью и моделью сущностей устанавливается связь с использованием механизма связи, предоставляемого AllFusion Process Modeler (BPwin). При установлении связи в AllFusion Process Modeler модель копируется словарь сущностей и атрибутов, включая определения и типы данных.

СТРУКТУРА ИТОГОВОГО ДОКУМЕНТА

Описана структура разделов результирующего документа, которые автоматически заполняются при экспорте требований из AllFusion Process Modeler модели.

Краткая характеристика системы

В разделе приводится характеристика системы верхнего уровня.

Выводятся: текстовое описание системы (Definition для A0); перечень регламентирующей информации; перечень входящей информации; перечень ресурсов (механиз-

мов); перечень исходящей информации; краткое описание деятельностей первого уровня; перечень деятельностей первого уровня (диаграмма A0) вместе с их кратким текстовым описанием (Description);

Описание окружения

В разделе перечисляются Externals & Referents, которые являются источниками или приёмниками информационных потоков AllFusion Process Modeler модели и помечены как внешние системы (при помощи UDP переменной).

Описание ролей

В разделе перечисляются все роли, которые были указаны в каких-либо деятельностях AllFusion Process Modeler модели. Для каждой роли приводится список деятельностей, в которых она принимает участие.

Требования к функциям

Для каждой функции выводится: наименование функции; определение функции (Definition) и требования к функции; примечание функции (Note); перечень регламентирующей информации (управляющие стрелки); перечень входящей информации (наименования входящих стрелок); перечень ресурсов (наименования стрелок-механизмов); перечень исходящей информации (наименования исходящих стрелок); перечень ролей, участвующих в исполнении деятельности; перечень вызовов, осуществляемых функцией (Call стрелки); перечень переходов к исполнению определённых сценариев, при выполнении заданных условий (используется в IDEF3 диаграммах).

Требования к информации

Для каждой дуги выводится: наименование дуги (Name), описание дуги (Definition), примечание дуги (Note), перечень сущностей, ассоциированных с дугой.

Для каждой сущности выводится: наименование сущности (Name), описание сущности (Definition), перечень атрибутов, ассоциированных с дугой.

Для каждого атрибута выводится: наименование атрибута (Name), тип атрибута (тип, заданный в AllFusion Data Modeler модели), описание атрибута (Definition).

ПОСЛЕДОВАТЕЛЬНОСТЬ СОЗДАНИЯ МОДЕЛИ

Создание ИС отображается в виде последовательности (рисунок 36).

Рисунок 36 – Последовательность создания ИС

ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ

Обычно сначала строится модель существующей организации работы -"AS-IS" (как есть). Анализ функциональной модели позволяет понять, где находятся наиболее слабые места, в чем будут состоять преимущества новых бизнес-процессов. Найденные в модели "AS-IS" недостатки можно исправить при создании модели "TO-BE" (как должно быть) - модели новой организации бизнес-процессов. IDEF0-модель предполагает наличие четко сформулированной цели, единственного субъекта моделирования.

Построение контекстной диаграммы. Модель в нотации IDEF0 представляет собой совокупность иерархически упорядоченных и взаимосвязанных диаграмм. Вершина этой древовидной структуры, представляющая собой самое общее описание системы и ее взаимодействия с внешней средой, называется контекстной диаграммой.

Декомпозиция. После описания системы в целом проводится разбиение ее на крупные фрагменты - декомпозицией. После декомпозиции контекстной диаграммы проводится декомпозиция каждого большого фрагмента системы на более мелкие и так далее до достижения нужного уровня подробности описания.

Экспертиза. После каждого сеанса декомпозиции проводятся сеансы экспертизы на соответствие реальных бизнес-процессов созданным. Найденные несоответствия исправляются. Достигается соответствие модели реальным бизнес – процессам.

ПРОИЗВОДСТВО ИЗДЕЛИЙ. ПОСТРОЕНИЕ ФУНКЦИОНАЛЬНОЙ МОДЕЛИ AS-IS (“Как есть”)

ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Рассматривается система производство изделий. Для производства изделий требуется сырье, комплектующие. Из сырья по определенной технологии вырабатывается полуфабрикат, из которого изготавливаются детали. Из деталей и комплектующих производится сборка изделия. На каждом этапе производства изделий осуществляется контроль качества полуфабрикатов, изготовленных деталей, собранного изделия. Брак поступает в переработку.

Основная группа функций по производству изделий: ***ОБРАБОТКА СЫРЬЯ, ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ, СБОРКА ИЗДЕЛИЯ, КОНТРОЛЬ КАЧЕСТВА.***

ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ

Контекстная диаграмма

Контекстная диаграмма (рисунок 37) рассматривает систему в целом (работа ***ИЗГОТОВЛЕНИЕ ИЗДЕЛИЙ***) и ее окружение (поставщики, заказчики, исполнители, технологии, задания). Входные потоки – ***СЫРЬЕ*** **и** ***КОМПЛЕКТУЮЩИЕ***, выходной поток – ***ГОТОВОЕ ИЗДЕЛИЕ***. Управление включает потоки ***ЗАДАНИЕ*** **и** ***ЧЕРТЕЖИ***. Механизм реализуется потоком ***ПЕРСОНАЛ***.

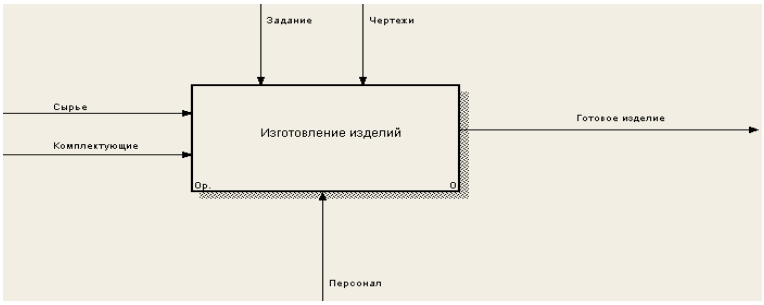


Рисунок 37 - Контекстная диаграмма

Первый уровень декомпозиции

Контекстная диаграмма декомпозируется на четыре работы: ***ОБРАБОТКА СЫРЬЯ, ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ, СБОРКА ИЗДЕЛИЯ, КОНТРОЛЬ КАЧЕСТВА*** (рисунок 38).

Работа (функция) ОБРАБОТКА СЫРЬЯ происходит в соответствии с *ЗАДАНИЕМ* и имеет на входе потоки *СЫРЬЕ* и *БРАК*, на выходе – *ПОЛУФАБРИКАТ*. Поток *БРАК* создается за счет бракованных деталей и бракованных изделий.

Работа ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ происходит в соответствии с *ЗАДАНИЕМ, ЧЕРТЕЖАМИ, РЕКОМЕНДАЦИЯМИ* на последнем этапе контроля качества и имеет на входе поток *ПОЛУФАБРИКАТ,* на выходе - *ДЕТАЛИ* и *БРАК*.

Работа СБОРКА ИЗДЕЛИЯ происходит в соответствии с *ЗАДАНИЕМ, ЧЕРТЕЖАМИ, РЕКОМЕНДАЦИЯМИ* и имеет на входе потоки *КОМПЛЕКТУЮЩИЕ, ДЕТАЛИ,* на выходе - *СОБРАННОЕ ИЗДЕЛИЕ.*

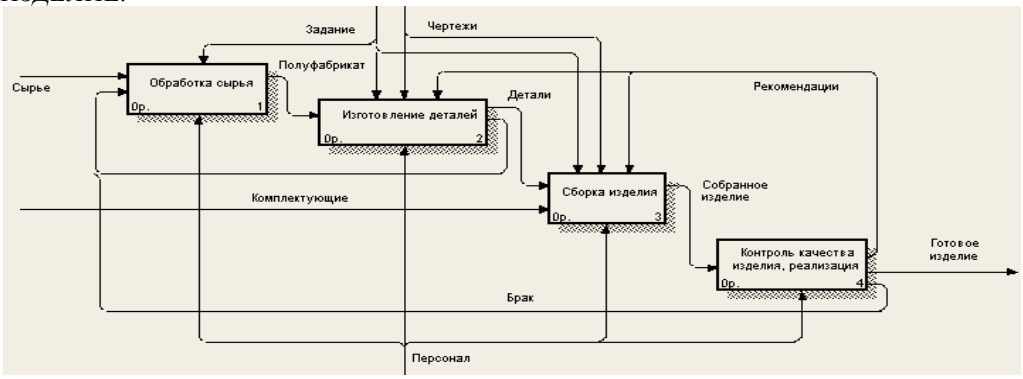


Рисунок 38 - Диаграмма декомпозиции

Работа КОНТРОЛЬ КАЧЕСТВА осуществляет контроль качества выпускаемого изделия и имеет на входе потоки *ГОТОВОЕ ИЗДЕЛИЕ, БРАК и РЕКОМЕНДАЦИИ*, необходимые для снижения брака на предыдущих этапах работы.

Стрелки (потоки) связывают работы между собой в соответствие с технологией и процессом изготовления изделий. Они реализуют определенные типы связей. Например, на рисунке поток *ДЕТАЛИ* связывает работы *ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ* и *СБОРКА ИЗДЕЛИЯ* - прямая связь по входу, когда стрелка выхода вышестоящей работы направляется на вход нижестоящей; поток *БРАК* - обратная связь по входу, когда выход нижестоящей работы направляется на вход вышестоящей; поток *РЕКОМЕНДАЦИИ* - обратная связь по управлению, когда выход нижестоящей работы направляется на управление вышестоящей.

Второй уровень декомпозиции

На этом уровне декомпозиции осуществляется детализация основных работ.

Функция ОБРАБОТКА СЫРЬЯ **(**рисунок 39) декомпозируется на две функции: *ИЗГОТОВЛЕНИЕ ПОЛУФАБРИКАТОВ* и *КОНТРОЛЬ КАЧЕСТВА ПОЛУФАБРИКАТОВ.*

Функция ИЗГОТОВЛЕНИЕ ПОЛУФАБРИКАТОВ имеет на входе потоки *СЫРЬЕ* и *БРАК,* управляющий поток *ЗАДАНИЕ,* на выходе - *ИЗГОТОВЛЕННЫЙ ПОЛУФАБРИКАТ.*

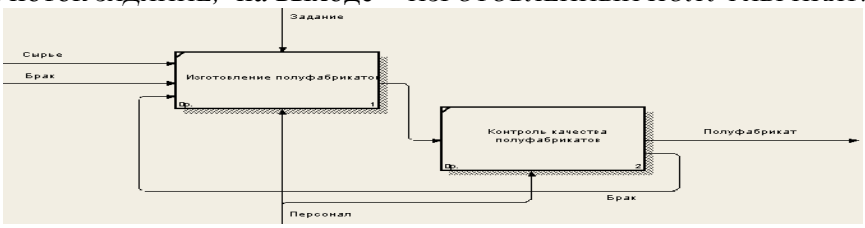


Рисунок 39 – Обработка сырья

Функция КОНТРОЛЬ КАЧЕСТВА ПОЛУФАБРИКАТОВ. В качестве управляющего потока – *ИЗГОТОВЛЕННЫЙ ПОЛУФАБРИКАТ,* на выходе - поток *ПОЛУФАБРИКАТ.*

Поток *ИЗГОТОВЛЕННЫЙ ПОЛУФАБРИКАТ* осуществляет прямую связь по управлению, когда выход вышестоящей работы направляется на управление нижестоящей.

Функция ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ (рисунок 40) детализируется на четыре функции: *ВЫБОР ВАРИАНТА ИЗГОТОВЛЕНИЯ, ВЫБОР ОБОРУДОВАНИЯ, ПРОИЗВОДСТВО ДЕТАЛЕЙ, КОНТРОЛЬ КАЧЕСТВА ДЕТАЛЕЙ.*

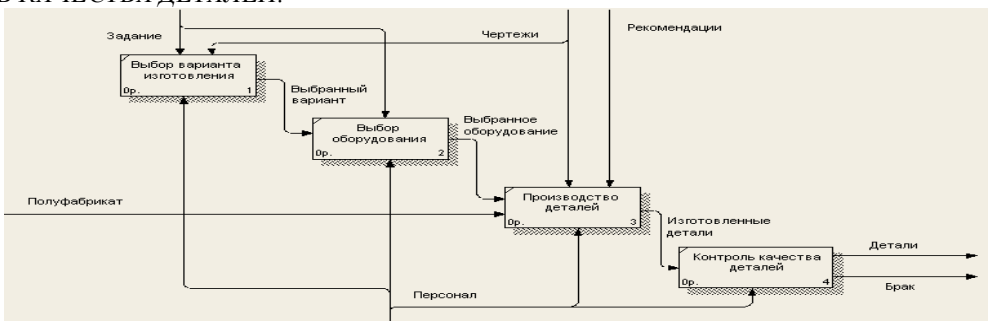


Рисунок 40 – Изготовление деталей

Функция СБОРКА ИЗДЕЛИЯ (рисунок 41) декомпозируется на три функции: *СБОРКА УЗЛОВ, ПРОВЕРКА УЗЛОВ, СБОРКА ВСЕГО ИЗДЕЛИЯ.*

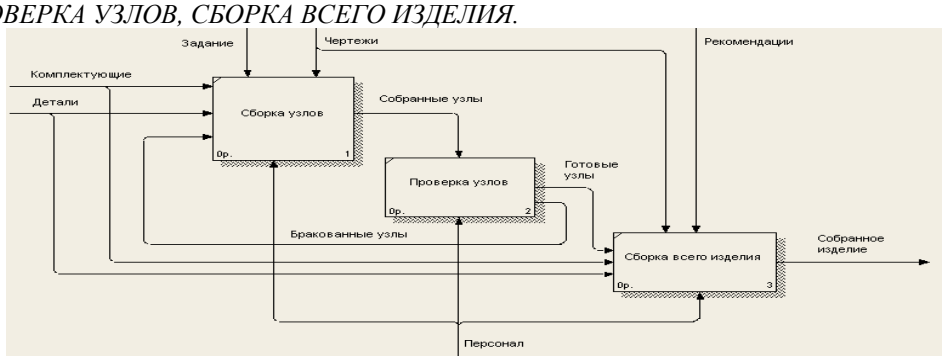


Рисунок 41 – Сборка изделия

Функция КОНТРОЛЬ КАЧЕСТВА (рисунок 42) декомпозируется на две функции: *ТЕСТИРОВАНИЕ ИЗДЕЛИЯ, ВЫРАБОТКА РЕКОМЕНДАЦИЙ.*

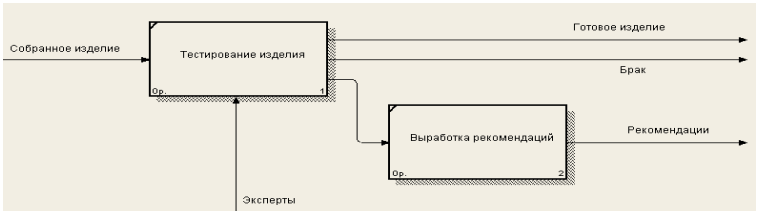


Рисунок 42 – Контроль качества

При необходимости декомпозицию работ (функций) продолжают до тех пор, пока не станет возможной их формализация. Для приведенного примера это касается функции *ПРОИЗВОДСТВО ДЕТАЛЕЙ,* которую можно декомпозировать далее.

ПОСТРОЕНИЕ ФУНКЦИОНАЛЬНОЙ МОДЕЛИ ТО-ВЕ ("Как должно быть")

Построение функциональной модели "Как должно быть" происходит на основе процессного подхода организации и управления деятельности предприятия как бизнес-процесса с управлением как всего процесса в целом, так и отдельными бизнес-процессами. Одна из основных функций деятельности бизнес-организаций, бизнес-процессов является функция маркетинг.

МАРКЕТИНГ

Маркетинг **-** вид человеческой деятельности, направленной на удовлетворение нужд и потребностей посредством обмена.

Основные функции маркетинга:

- ***аналитическая функция -*** изучения структуры, запросов потребителей, объемов закупок, реакции на появление новых товаров и изменение цен, существующего ассортимента, действующих на рынке стандартов, норм, требований к качеству товаров, изучаются компании-конкуренты, товарное предложение и спрос на их продукцию, система сбыта, прогноз на будущее в плане конкуренции продукции;

- ***организация материально - технического снабжения.*** Материально - техническое снабжение – это наиболее существенный элемент обеспечения производства. В условиях рыночной экономики предприятие обеспечивает свои потребности посредством закупки необходимых материально-технических ресурсов по прямым договорам купли-продажи, а также используя возможности оптового рынка;

- ***управление качеством и конкурентоспособностью товаров*** **-** соответствие стандартам. Товар должен удовлетворять явным ожиданиям потребителя - технически, эксплуатационно, эстетически, по цене;

- ***организация производства новых товаров*** **-** это так называемые пионерные товары, они поднимают на качественно новую ступень удовлетворение уже известной потребности; позволяют значительно более широкому кругу покупателей удовлетворять на определенном уровне известную потребность;

- ***распределительно-сбытовые функции -*** речь идет о продвижении товара на рынок. Воздействие на рынок, являющееся одним из основополагающих принципов маркетинга, преследует цель способствовать успешной реализации товаров;

- ***ценообразование*** является сложным процессом для любого предприятия. Выбор общего направления ценовой политики, т.е. определение цен на новые и уже выпускаемые товары (оказываемые услуги) и повышение рентабельности производства, является важной составляющей сбытовых функций маркетинга, и ее роль все больше возрастает;

- ***рекламная кампания*** предприятия имеет целью создать у потенциального потребителя полное представление о своих товарах (услугах), включая их полный спектр, качество, стоимость.

Управленческие функции маркетинга предполагают планирование хозяйственной деятельностью предприятия и управление производством. В процессе этой деятельности определяется общая стратегия предприятия, и формулируются оперативные задачи.

Управленческие функции маркетинга включают**:**

- ***стратегическое планирование.*** Цель стратегического планирования состоит в том, чтобы усиливать сильные стороны предприятия и уменьшать воздействие слабых

сторон. Это достигается с помощью создания и поддержания стратегического соответствия между целями фирмы, ее потенциальными возможностями и шансами в сфере маркетинга. ***Цели деятельности*** фирмы могут быть количественными и качественными. ***К количественным целям*** можно отнести объем прибыли, рост прибыли, долю рынка, производительность труда и т.д. ***Качественные цели,*** в основном, связаны с имиджем, престижем фирмы, издержками и спонсированием общественных организаций и мероприятий;

- ***маркетинговый контроль***, вступающий как один из важнейших управленческих функций, бывает трех видов: стратегический, ежегодный плановый (тактический) и прибыли. ***Стратегический*** — это периодическая, всесторонняя и объективная проверка маркетинговой деятельности предприятия с целью выявления соответствия выбранной стратегии реальным процессам, протекающим на рынке. ***Тактический*** — текущий контроль достижения намеченных целей (объемы производства, продаж, доля занимаемого рынка, отношение потребителей к товарам и т.д.);

Информационное обеспечение маркетинга представляет собой совокупность различных видов исходных данных, привлекаемых в ходе анализа рыночных процессов и возможностей предприятия для разработки и обоснования стратегии и тактики его маркетинговой деятельности.

Виды информации: ***внутренняя*** — основывается на бухгалтерской, статистической и оперативной отчетности предприятия, раскрывает внутреннее его состояние и содержит данные о движении товаров и их запасов, доходах и расходах; ***внешняя*** — дает возможность изучать состояние рынка и его инфраструктуру, поведение покупателей и поставщиков, действия конкурентов, меры государственного регулирования, в том числе законодательные; ***исследовательская*** — позволяет более глубоко раскрыть состояние отдельных элементов рынка и маркетинговой деятельности предприятия.

ПОСТРОЕНИЕ МОДЕЛИ

Рассматривается система ***ПРОИЗВОДСТВО ИЗДЕЛИЙ***. В качестве модели AS-IS используется рассмотренная ранее модель ***ИЗГОТОВЛЕНИЕ ИЗДЕЛИЙ***. На основе этой модели строится модель TO-BE (как должно быть).

Требования к модели

Если рассматривать систему ***ПРОИЗВОДСТВО ИЗДЕЛИЙ*** как бизнес-систему, то необходимо ввести в модель функцию ***МАРКЕТИНГ***. С введением функции ***МАРКЕТИНГ*** появятся потоки данных ***ЗАКАЗ, ИНФОРМАЦИЯ О РЫНКЕ, РЕКЛАМА, ЗАКАЗЫ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ***.

Кроме того, необходимо добавить к функции ***КОНТРОЛЬ КАЧЕСТВА*** функцию ***РЕАЛИЗАЦИЯ ИЗДЕЛИЯ***.

Добавим также функцию ***УПРАВЛЕНИЕ БИЗНЕС-ПРОЦЕССОМ.***

Контекстная диаграмма будет иметь вид (рисунок 43):

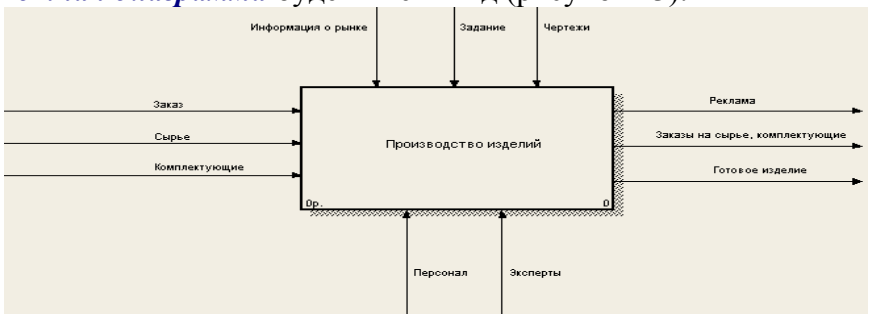


Рисунок 43 – Контекстная диаграмма

Первый уровень декомпозиции (рисунок 44)

Принятый заказ на изготовление изделия от заказчика обрабатывается с помощью функции ***МАРКЕТИНГ*** и передается по инстанции в виде потока данных ***ЗАКАЗ НА ИЗДЕЛИЕ*** для выполнения функциями***:*** *ОБРАБОТКА СЫРЬЯ, ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ, СБОРКА ИЗДЕЛИЯ.*

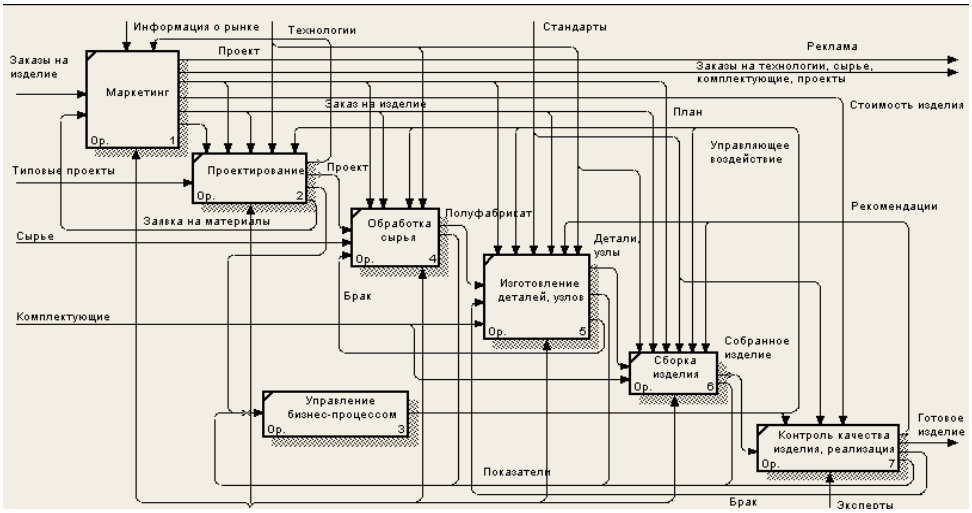


Рисунок 44 – Первый уровень декомпозиции

ДИАГРАММЫ DFD

Диаграммы потоков данных (Data flow diagramming, DFD) используются для описания документооборота и обработки информации. Их можно использовать как дополнение к модели IDEF0 для более наглядного отображения текущих операций документооборота в корпоративных системах обработки информации.

Второй уровень декомпозиции. Декомпозиция функции МАРКЕТИНГ

Функция МАРКЕТИНГ (рисунок 45):

- *ПРИЕМ ЗАКАЗОВ*, в результате чего появляется документ *ЗАКАЗ*;

- *ЗАКЛЮЧЕНИЕ ДОГОВОРА* – документ *ДОГОВОР*;

- *ФОРМИРОВАНИЕ ЗАКАЗОВ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ* – на основании данных *ДОГОВОРА* составляется заказ предприятиям-поставщикам – выходной поток *ЗАКАЗ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ;*

- *ОПРЕДЕЛЕНИЕ СТОИМОСТИ И ЦЕНЫ ИЗДЕЛИЯ* на основе данных *ЗАКАЗ* и *ИНФОРМАЦИЯ О РЫНКЕ* – выходной поток данных *ЦЕНА*;

- *ИЗГОТОВЛЕНИЕ РЕКЛАМЫ* на основе данных *ЗАКАЗ* и *ИНФОРМАЦИЯ О РЫНКЕ* – выходной поток *РЕКЛАМА;*

- *ПЛАНИРОВАНИЕ* на основе данных *ЗАКАЗ* и *ПОТРЕБНОСТИ РЫНКА* – поток *ПЛАН;*

Рисунок 45 – Маркетинг

Декомпозиция функции ПРОЕКТИРОВАНИЕ

Функция ПРОЕКТИРОВАНИЕ (рисунок 46) состоит из функций:

- *РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ* на основе данных *ДОГОВОРА, ТИПОВЫХ ПРОЕКТОВ, ПЛАНА И ЗАДАНИЯ,* в результате чего появляется документ *ТЕХНИЧЕСКОЕ ЗАДАНИЕ;*

- *РАЗРАБОТКА КОНСТРУКТОРСКОЙ* и *ТЕНОЛОГИЧЕСКОЙ ДОКУМЕНТАЦИИ* на основе данных *ТЕХНИЧЕСКОГО ЗАДАНИЯ* - выходной поток *КОНСТРУКТОРСКАЯ* и *ТЕНОЛОГИЧЕСКАЯ ДОКУМЕНТАЦИЯ;*

- *РАЗРАБОТКА ПРОЕКТА* на основе данных *КОНСТРУКТОРСКОЙ* и *ТЕНОЛОГИЧЕСКОЙ ДОКУМЕНТАЦИИ.*

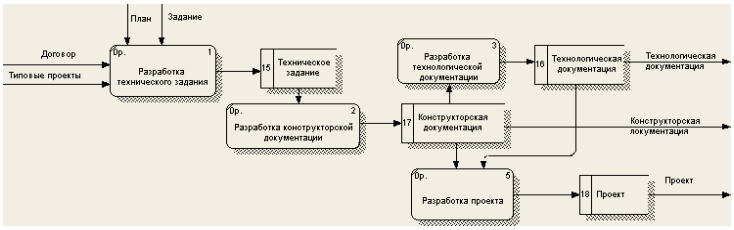


Рисунок 46 - Проектирование

Декомпозиция функции ОБРАБОТКА СЫРЬЯ

Функция ОБРАБОТКА СЫРЬЯ детализируется на функции (рисунок 47):

- *ИЗГОТОВЛЕНИЕ ПОЛУФАБРИКАТА*. Полуфабрикат изготавливается из сырья и брака при изготовлении деталей и сборки изделия (потоки *СЫРЬЕ* и *БРАК*) – выходной поток *ИЗГОТОВЛЕННЫЙ ПОЛУФАБРИКАТ.*

- *КОНТРОЛЬ КАЧЕСТВА ПОЛУФАБРИКАТА*. Некачественный полуфабрикат возвращается на переработку, а остальной используется для изготовления деталей – поток данных *ПОЛУФАБРИКАТ.*

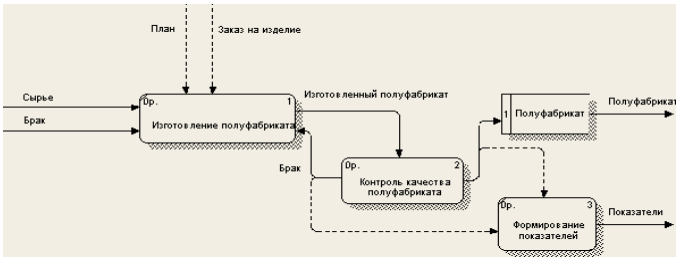


Рисунок 47 – Обработка сырья

Декомпозиция функции ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ

Функция ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ детализируется на функции (рисунок 48):

- *ВЫБОР ВАРИАНТА ИЗГОТОВЛЕНИЯ* на основании данных *ЗАДАНИЕ* и *ЧЕРТЕЖИ - ВЫБРАННЫЙ ВАРИАНТ;*

- *ВЫБОР ОБОРУДОВАНИЯ* – поток *ВЫБРАННОЕ ОБОРУДОВАНИЕ*;

- *ПРОИЗВОДСТВО ДЕТАЛЕЙ* на основе потоков данных *ПОЛУФАБРИКАТ*, *ОБОРУДОВАНИЕ*, *ЗАКАЗ НА ИЗДЕЛИЕ*, *ЧЕРТЕЖИ*, *РЕКОМЕНДАЦИИ* – поток *ИЗГОТОВЛЕННЫЕ ДЕТАЛИ*;

- *КОНТРОЛЬ КАЧЕСТВА ДЕТАЛЕЙ*. Некачественные детали возвращаются на переработку – поток *БРАК*, остальные – на сборку изделия – поток *ДЕТАЛИ*.

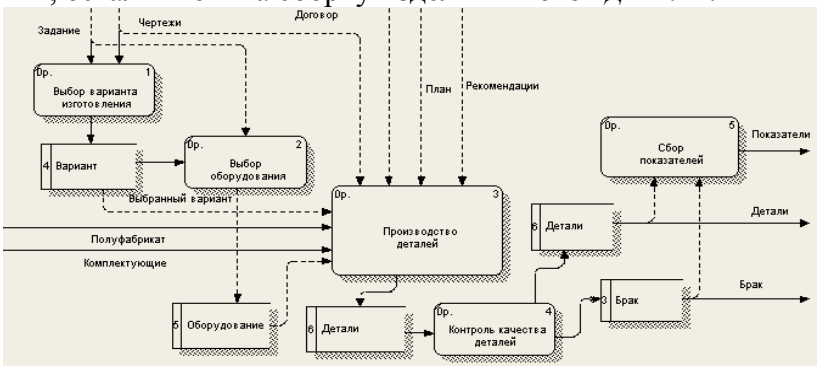


Рисунок 48 – Изготовление деталей

Декомпозиция функции СБОРКА ИЗДЕЛИЯ

Функция *СБОРКА ИЗДЕЛИЯ* детализируется на функции (рисунок 49):

- *СБОРКА УЗЛОВ* на основе потоков *КОМПЛЕКТУЮЩИЕ*, *ДЕТАЛИ*, *ЗАДАНИЕ*, *ЗАКАЗ НА ИЗДЕЛИЕ*, *ЧЕРТЕЖИ* – поток *СОБРАННЫЕ УЗЛЫ*;

- *ПРОВЕРКА УЗЛОВ*. Неверно собранные узы возвращаются на сборку (поток *БРАК*), остальные идут на сборку изделия – поток *УЗЛЫ*;

- *СБОРКА ВСЕГО ИЗДЕЛИЯ* на основе потоков *УЗЛЫ*, *КОМПЛЕКТУЮЩИЕ*, *ЗАДАНИЕ*, *ЗАКАЗ НА ИЗДЕЛИЕ*, *ЧЕРТЕЖИ*, *РЕКОМЕНДАЦИИ* – поток *СОБРАННОЕ ИЗДЕЛИЕ.*

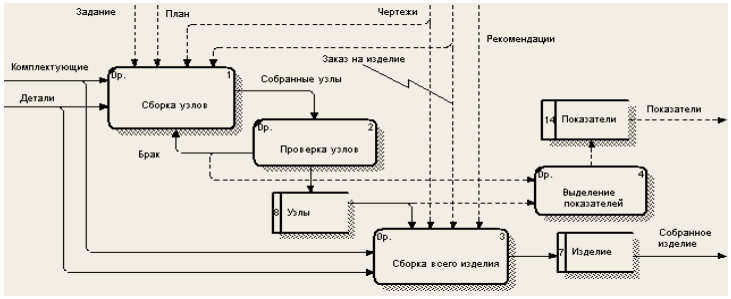


Рисунок 49 – Сборка изделия

Декомпозиция функции КОНТРОЛЬ КАЧЕСТВА ИЗДЕЛИЯ, РЕАЛИЗАЦИЯ

Функция *КОНТРОЛЬ КАЧЕСТВА ИЗДЕЛИЯ* детализируется на функции (рисунок 50):

- *ТЕСТИРОВАНИЕ ИЗДЕЛИЯ*. Собранное изделие тестируется экспертами (поток *ЭКСПЕРТЫ*). Изделие, не прошедшее тест (поток *БРАК*), возвращается на переработку, остальные изделия поступают на продажу (поток *ИЗДЕЛИЕ*);

- *ПРОДАЖА ИЗДЕЛИЯ* на основе потоков данных *ИЗДЕЛИЕ* и *ЦЕНА ИЗДЕЛИЯ* – поток *ГОТОВОЕ ИЗДЕЛИЕ*.

- *ВЫРАБОТКА РЕКОМЕНДАЦИЙ* на основе потока *ЗАКЛЮЧЕНИЕ ПО ТЕСТИРОВАНИЮ* – поток *РЕКОМЕНДАЦИИ*.

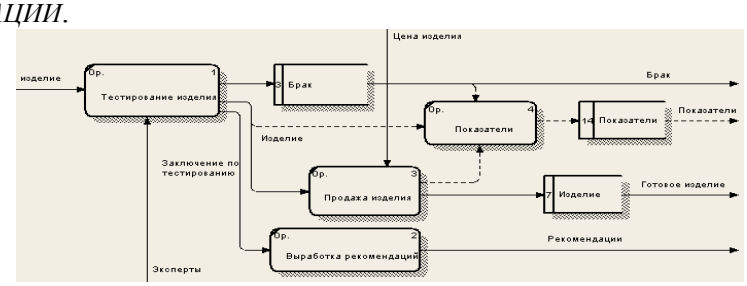


Рисунок 50 – Контроль качества изделия

ДИАГРАММЫ IDEF3

Наличие в диаграммах DFD элементов для описания источников, приемников и хранилищ данных позволяет более наглядно описать процесс документооборота. Однако для описания логики взаимодействия информационных потоков более подходит IDEF3. Завершим декомпозицию DFD диаграммой IDEF3 *СБОРКА ВСЕГО ИЗДЕЛИЯ* (рисунок 51).

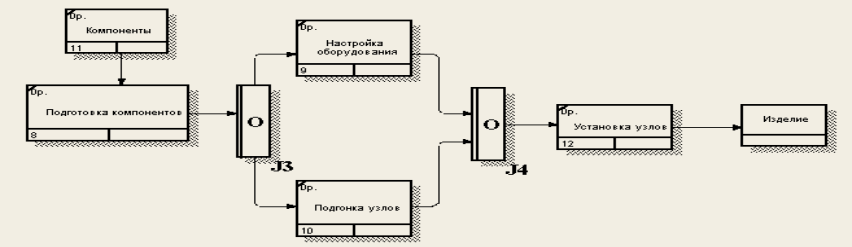


Рисунок 51 – Сборка изделия

СМЕШАННАЯ МОДЕЛЬ

Синтаксический анализ модели BPwin позволяет легко обнаружить "бесполезные" (не имеющие выхода), "неуправляемые" (не имеющие управления) и "простаивающие" работы. Более тонкий анализ позволяет выявить дублирующие, избыточные или неэффективные работы. Модель дает целостное представление о работе системы в целом и позволяет понять взаимосвязи всех составляющих системы. При этом часто выясняется, что обработка информации и использование ресурсов неэффективны, важная информация не доходит до соответствующего рабочего места и т.д. Признаком неэффективной организации работ является, например, отсутствие обратных связей по входу и управлению для многих, критически важных работ.

Невозможно построить эффективную ИС при неэффективной общей организации работы. Поэтому результатом анализа и критической оценки модели AS-IS должно быть перенаправление информационных потоков и усовершенствование бизнес-процессов в новой модели TO-BE, которая должна использоваться для реорганизации деятельности

предприятия. Однако при создании ИС модель процессов - это только первый шаг, за которым обычно следует построение модели данных.

СПЕЦИФИКАЦИИ ПРОЦЕССОВ

Спецификация функции Маркетинг

@ВХОД *= ЗАКАЗ*
@ВХОД *= ИНФОРМАЦИЯ О РЫНКЕ*
@ВЫХОД *= РЕКЛАМА*
@ВЫХОД *= ПЛАН*
@ВЫХОД *=ЗАКАЗ НА ИЗДЕЛИЕ*
@ВЫХОД *=ЗАКАЗ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ*
@ВЫХОД *=ЦЕНА ИЗДЕЛИЯ*
@СПЕЦПРОЦ 1 МАРКЕТИНГ

ВЫПОЛНИТЬ *ПРИЕМ ЗАКАЗОВ, в результате чего появляется документ ЗАКАЗ*;

ВЫПОЛНИТЬ *ЗАКЛЮЧЕНИЕ ДОГОВОРА – документ ДОГОВОР* – выходной поток *ЗАКАЗ НА ИЗДЕЛИЕ*;

ВЫПОЛНИТЬ *ФОРМИРОВАНИЕ ЗАКАЗОВ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ – на основании данных ДОГОВОРА составляется заказ предприятиям-поставщикам – выходной поток ЗАКАЗ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ;*

ВЫПОЛНИТЬ *ОПРЕДЕЛЕНИЕ СТОИМОСТИ И ЦЕНЫ ИЗДЕЛИЯ на основе данных ЗАКАЗ и ИНФОРМАЦИЯ О РЫНКЕ – выходной поток данных ЦЕНА*;

ВЫПОЛНИТЬ *ИЗГОТОВЛЕНИЕ РЕКЛАМЫ на основе данных ЗАКАЗ и ИНФОРМАЦИЯ О РЫНКЕ – выходной поток РЕКЛАМА;*

ВЫПОЛНИТЬ *ПЛАНИРОВАНИЕ на основе данных ЗАКАЗ* и *ПОТРЕБНОСТИ РЫНКА* – поток *ПЛАН;*

@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 1

Спецификация функции Проектирование

@ВХОД *= ДОГОВОР*
@ВХОД *= ТИПОВЫЕ ПРОЕКТЫ*
@ВХОД *= ЗАДАНИЕ*
@ВХОД *= ПЛАН*
@ВЫХОД *=КОНСТРУКТОРСКАЯ ДОКУМЕНТАЦИЯ*
@ВЫХОД *=ТЕХНОЛОГИЧЕСКАЯ ДОКУМЕНТАЦИЯ*
@ВЫХОД *=ПРОЕКТ*
@СПЕЦПРОЦ 2 ПРОЕКТИРОВАНИЕ

ВЫПОЛНИТЬ *разработку ТЕХНИЧЕСКОГО ЗАДАНИЯ на основе данных ДОГОВОРА, ТИПОВЫХ ПРОЕКТОВ, ПЛАНА И ЗАДАНИЯ, в результате чего появляется документ ТЕХНИЧЕСКОЕ ЗАДАНИЕ;*

ВЫПОЛНИТЬ *разработку КОНСТРУКТОРСКОЙ и ТЕНОЛОГИЧЕСКОЙ ДОКУМЕНТАЦИИ на основе данных ТЕХНИЧЕСКОГО ЗАДАНИЯ - выходной поток КОНСТРУКТОРСКАЯ и ТЕНОЛОГИЧЕСКАЯ ДОКУМЕНТАЦИЯ;*

ВЫПОЛНИТЬ *разработку ПРОЕКТА на основе данных КОНСТРУКТОРСКОЙ и ТЕНОЛОГИЧЕСКОЙ ДОКУМЕНТАЦИИ.*

@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 2

Спецификация функции Обработка сырья

@ВХОД *= СЫРЬЕ*
@ВХОД *= БРАК*
@ВХОД *= ЗАКАЗ НА ИЗДЕЛИЕ*
@ВХОД *= ПЛАН*
@ВЫХОД *=ПОЛУФАБРИКАТ*
@ВЫХОД *=ПОКАЗАТЕЛИ*
@СПЕЦПРОЦ 3 ОБРАБОТКА СЫРЬЯ

ВЫПОЛНИТЬ *ИЗГОТОВЛЕНИЕ ПОЛУФАБРИКАТА. Полуфабрикат изготавливается из сырья и брака при изготовлении деталей и сборки изделия (потоки СЫРЬЕ и БРАК) – выходной поток ИЗГОТОВЛЕННЫЙ ПОЛУФАБРИКАТ.*

ВЫПОЛНИТЬ *КОНТРОЛЬ КАЧЕСТВА ПОЛУФАБРИКАТА.*

ЕСЛИ *Некачественный полуфабрикат* ***ТО ВЫПОЛНИТЬ*** *возвращается на переработку,* ***ИНАЧЕ***
ВЫПОЛНИТЬ *для изготовления деталей – поток данных ПОЛУФАБРИКАТ.* ***КОНЕЦЕСЛИ***

@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 3

Спецификация функции Изготовление деталей

@ВХОД *= ЗАКАЗ НА ИЗДЕЛИЕ*
@ВХОД *= ПОЛУФАБРИКАТ*
@ВХОД *=ЗАДАНИЕ*

@ВХОД *=ЧЕРТЕЖИ*
@ВХОД *=ПЛАН*
@ВХОД *=РЕКОМЕНДАЦИИ*
@ВЫХОД *= ДЕТАЛИ*
@ВЫХОД *= БРАК*
@ВЫХОД *=ПОКАЗАТЕЛИ*
@СПЕЦПРОЦ 4 ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ
ВЫПОЛНИТЬ *ВЫБОР ВАРИАНТА ИЗГОТОВЛЕНИЯ на основании данных ЗАДАНИЕ* и *ЧЕРТЕЖИ - ВЫБРАННЫЙ ВАРИАНТ.*
ВЫПОЛНИТЬ *ВЫБОР ОБОРУДОВАНИЯ – поток ВЫБРАННОЕ ОБОРУДОВАНИЕ*;
ВЫПОЛНИТЬ *ПРОИЗВОДСТВО ДЕТАЛЕЙ на основе потоков данных ПОЛУФАБРИКАТ, ОБОРУДОВАНИЕ, ЗАКАЗ НА ИЗДЕЛИЕ, ЧЕРТЕЖИ, РЕКОМЕНДАЦИИ – поток ИЗГОТОВЛЕННЫЕ ДЕТАЛИ*;
ВЫПОЛНИТЬ *- КОНТРОЛЬ КАЧЕСТВА ДЕТАЛЕЙ.* ***ЕСЛИ*** *Некачественные детали* ***ТО ВЫПОЛНИТЬ*** *возвращается на переработку – поток БРАК,* ***ИНАЧЕ ВЫПОЛНИТЬ*** *на сборку изделия – поток данных ДЕТАЛИ.* ***КОНЕЦЕСЛИ***
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 4

Спецификация функции Сборка изделия

@ВХОД *= ЗАКАЗ НА ИЗДЕЛИЕ*
@ВХОД *= КОМПЛЕКТУЮЩИЕ*
@ВХОД *=ЗАДАНИЕ*
@ВХОД *=ЧЕРТЕЖИ*
@ВХОД *=ПЛАН*
@ВХОД *=РЕКОМЕНДАЦИИ*
@ВХОД *= ДЕТАЛИ*
@ВЫХОД *= СОБРАННОЕ ИЗДЕЛИЕ*
@ВЫХОД= *ПОКАЗАТЕЛИ*
@СПЕЦПРОЦ 5 СБОРКА ИЗДЕЛИЯ
ВЫПОЛНИТЬ *СБОРКА УЗЛОВ на основе потоков КОМПЛЕКТУЮЩИЕ, ДЕТАЛИ, ЗАДАНИЕ, ЗАКАЗ НА ИЗДЕЛИЕ, ЧЕРТЕЖИ – поток СОБРАННЫЕ УЗЛЫ*;
ВЫПОЛНИТЬ *ПРОВЕРКА УЗЛОВ.* ***ЕСЛИ*** *неверно собранные узлы,* ***ТО ВЫПОЛНИТЬ*** *возвращаются на сборку – поток БРАК,* ***ИНАЧЕ ВЫПОЛНИТЬ*** *на сборку изделия – поток данных УЗЛЫ*; ***КОНЕЦЕСЛИ***
ВЫПОЛНИТЬ *СБОРКА ВСЕГО ИЗДЕЛИЯ на основе потоков УЗЛЫ, КОМПЛЕКТУЮЩИЕ, ЗАДАНИЕ, ЗАКАЗ НА ИЗДЕЛИЕ, ЧЕРТЕЖИ, РЕКОМЕНДАЦИИ – поток СОБРАННОЕ ИЗДЕЛИЕ.*
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 5

Спецификация функции Контроль качества изделия

@ВХОД *= ИЗДЕЛИЕ*
@ВХОД *= ЦЕНА*
@ВЫХОД *= ГОТОВОЕ ИЗДЕЛИЕ*
@ВЫХОД *= РЕКОМЕНДАЦИИ*
@ВЫХОД= *БРАК*
@ВЫХОД= *ПОКАЗАТЕЛИ*
@СПЕЦПРОЦ 6 КОНТРОЛЬКАЧЕСТВА ИЗДЕЛИЯ
ВЫПОЛНИТЬ *ТЕСТИРОВАНИЕ ИЗДЕЛИЯ. Собранное изделие тестируется экспертами.* ***ЕСЛИ*** *Изделие, не прошедшее тест,* ***ТО ВЫПОЛНИТЬ*** *возвращается на переработку – поток БРАК,* ***ИНАЧЕ ВЫПОЛНИТЬ*** *поступает на продажу – поток данных ГОТОВОЕ ИЗДЕЛИЕ.* ***КОНЕЦЕСЛИ***
ВЫПОЛНИТЬ *ПРОДАЖА ИЗДЕЛИЯ на основе потоков данных ИЗДЕЛИЕ и ЦЕНА ИЗДЕЛИЯ – поток ГОТОВОЕ ИЗДЕЛИЕ.*
ВЫПОЛНИТЬ *ВЫРАБОТКА РЕКОМЕНДАЦИЙ на основе потока ЗАКЛЮЧЕНИЕ ПО ТЕСТИРОВАНИЮ – поток РЕКОМЕНДАЦИИ*
@ КОНЕЦ СПЕЦИФИКАЦИИ ПРОЦЕССА 6

МОДЕЛЬ ДАННЫХ

ЭКСПОРТ ДАННЫХ В AllFusion ERWin Data Modeler

Информацию из AllFusion Process Modeler (функциональная модель) можно экспортировать в AllFusion ERwin Data Modeler с возможностью создания баз данных. Таким же образом, данные из AllFusion ERwin Data Modeler могут быть импортированы в AllFusion Process Modeler и привязаны к потокам и операциям моделей процессов.

Хранилища данных функциональной модели в AllFusion ERwin Data Modeler (ER-Win) при экспорте становятся сущностями (рисунок 52).

AllFusion Process Modeler - [Entity Dictionary]

Dictionary Edit View Help

Name	Definition	Exchange with ERwin
Брак		Exchange with ERwin
Вариант		Exchange with ERwin
Данные кредитной карты		Exchange with ERwin
Детали		Exchange with ERwin
Договор		Exchange with ERwin
Заказ на сырье, комплектующие		Exchange with ERwin
Заказы		Exchange with ERwin
Изделие		Exchange with ERwin
Конструкторская документация		Exchange with ERwin
Оборудование		Exchange with ERwin
План		Exchange with ERwin
Показатели		Exchange with ERwin
Полуфабрикат		Exchange with ERwin
Проект		Exchange with ERwin
Техническое задание		Exchange with ERwin
Технологическая документация		Exchange with ERwin
Узлы		Exchange with ERwin
Цена		Exchange with ERwin

Рисунок 52 – Сущности модели

При построения модели данных AllFusion ERwin Data Modeler позволяет разрабатывать два вида моделей: ***логическую*** (Logical) и ***физическую*** (Physical). ***Логическая*** модель отображает объекты так, как они выглядят в реальном мире. Модель строится из сущностей, атрибутов и связей (рисунок 53).

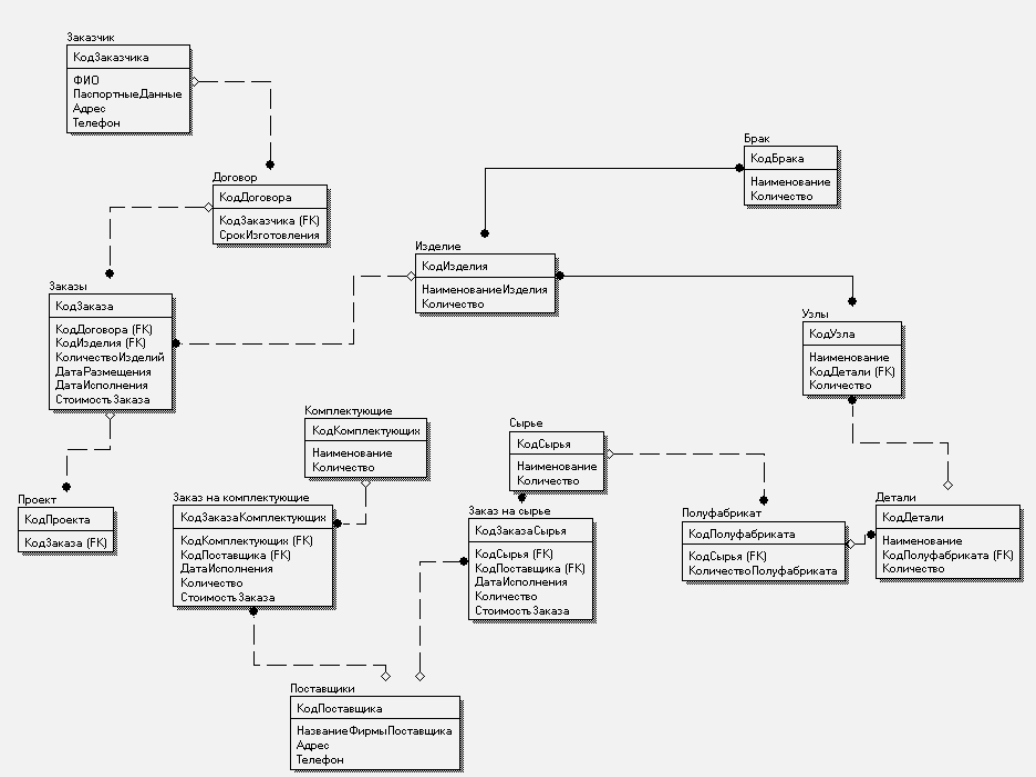


Рисунок 53 – Модель данных

Физическая модель предназначена для реализации базы данных под конкретную СУБД и обладает всеми теми ограничениями, которые накладывает конкретная СУБД. Для одной логической модели может быть несколько физических моделей. Физическая модель строится из таблиц, полей, связей и видов.

Конвертируем модель в СУБД Access, получаем схему БД (рисунок 54).

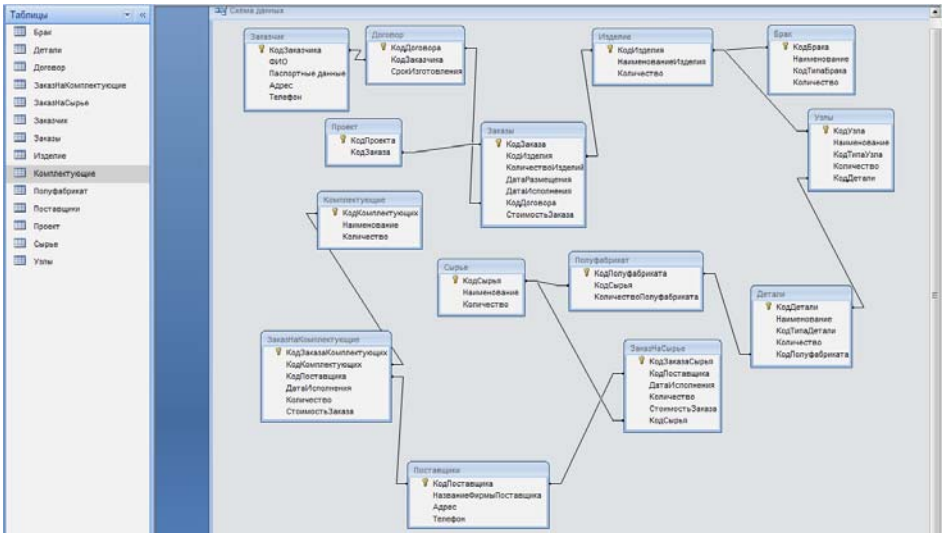


Рисунок 54 – Схема БД

В полученной базе данных можно создавать запросы. MS Access позволяет создавать запросы с помощью конструктора или вручную на языке SQL. Например:

```
INSERT INTO Изделие (Наименование, Количество)
VALUES (Forms!Форма!НаименованиеИзд, Forms!Форма!КоличествоИзд);
```

МОДУЛИ

Модуль МАРКЕТИНГ

***ВХОДНЫЕ ДАННЫЕ**: ЗАКАЗЫ, ИНФОРМАЦИЯ О РЫНКЕ.*

***ВЫХОДНЫЕ ДАННЫЕ**: РЕКЛАМА, ПЛАН, ЗАКАЗ НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ, ЦЕНА ИЗДЕЛИЯ, ЗАКАЗ НА ИЗДЕЛИЕ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** осуществить ПРИЕМ ЗАКАЗОВ на изготовление ИЗДЕЛИЙ, в результате чего появляется документ ЗАКАЗ; на основании данных ЗАКАЗА составить ДОГОВОР. Данные ДОГОВОРА использовать при составлении ЗАКАЗА НА СЫРЬЕ, КОМПЛЕКТУЮЩИЕ предприятиям - поставщикам. На основе данных ЗАКАЗОВ, РЕСУРСОВ ПРЕДПРИЯТИЯ и ПОТРЕБНОСТЕЙ РЫНКА создать ПЛАН. На основе данных ЗАКАЗ и ИНФОРМАЦИЯ О РЫНКЕ сформировать ЦЕНУ ИЗДЕЛИЯ и создать РЕКЛАМУ.*

Модуль ПРОЕКТИРОВАНИЕ

***ВХОДНЫЕ ДАННЫЕ**: ДОГОВОР, ПЛАН, ЗАДАНИЕ, ТИПОВЫЕ ПРОЕКТЫ.*

***ВЫХОДНЫЕ ДАННЫЕ**: КОНСТРУКТОРСКАЯ ДОКУМЕНТАЦИЯ, ПРОЕКТ, ТЕХНОЛОГИЧЕСКАЯ ДОКУМЕНТАЦИЯ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** на основе данных ДОГОВОРА, ТИПОВЫХ ПРОЕКТОВ, ПЛАНА И ЗАДАНИЯ разработать ТЕХНИЧЕСКОЕ ЗАДАНИЕ, КОНСТРУКТОРСКУЮ и ТЕНОЛОГИЧЕСКУЮ ДОКУМЕНТАЦИЮ. На основе документации разработать ПРОЕКТ изделия.*

Модуль ОБРАБОКА СЫРЬЯ

***ВХОДНЫЕ ДАННЫЕ**: ЗАКАЗ НА ИЗДЕЛИЕ, СЫРЬЕ, ПЛАН, БРАК.*

***ВЫХОДНЫЕ ДАННЫЕ**: ПОЛУФАБРИКАТ, ПОКАЗАТЕЛИ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** на основании данных потока СЫРЬЕ с заданными характеристиками в количестве, определяемым ПЛАНОМ выпуска изделий, вычислить количество выпущенного ПОЛУФАБРИКАТА в соответствии с технологией и нормами качества и количество БРАКА, полученного в этой технологической операции, а также во всех последующих операциях. Рассчитать процентное соотношение брака.*

Модуль ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ

***ВХОДНЫЕ ДАННЫЕ**: ЗАКАЗ НА ИЗДЕЛИЕ, ПОЛУФАБРИКАТ, РЕКОМЕНДАЦИИ, ЧЕРТЕЖИ, ЗАДАНИЕ, ПЛАН.*

***ВЫХОДНЫЕ ДАННЫЕ**: ДЕТАЛИ, БРАК, ПОКАЗАТЕЛИ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** на основании данных потоков ЗАДАНИЕ и ЧЕРТЕЖИ выбрать ВАРИАНТ ИЗГОТОВЛЕНИЯ деталей, выбрать ОБОРУДОВАНИЕ, на котором будут произведены ДЕТАЛИ из ПОЛУФАБРИКАТА в соответствии с технологией, ЗАКАЗОМ НА ИЗДЕЛИЕ в количестве, определяемым ПЛАНОМ с учетом РЕКОМЕНДАЦИЙ. Учесть количество ПОЛУФАБРИКАТА и количество произведенных из него ДЕТАЛЕЙ, определить количество БРАКА в соответствии с нормами качества. Определить загрузку оборудования.*

Модуль СБОРКА ИЗДЕЛИЯ

***ВХОДНЫЕ ДАННЫЕ**: ЗАКАЗ НА ИЗДЕЛИЕ, КОМПЛЕКТУЮЩИЕ, ЗАДАНИЕ, ЧЕРТЕЖИ, ПЛАН, РЕКОМЕНДАЦИИ, ДЕТАЛИ.*

***ВЫХОДНЫЕ ДАННЫЕ**: СОБРАННОЕ ИЗДЕЛИЕ, ПОКАЗАТЕЛИ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** на основании данных потоков КОМПЛЕКТУЮЩИЕ, ДЕТАЛИ с заданными характеристиками в количестве, определяемым ПЛАНОМ, ЗАДАНИЕМ, ЗАКАЗОМ НА ИЗДЕЛИЕ, вычислить количество собранных УЗЛОВ в соответствии с технологией и нормами качества, и количество БРАКА, полученного в этой технологической операции. На основании данных потоков КОМПЛЕКТУЮЩИЕ, ДЕТАЛИ с заданными характеристиками в количестве, определяемым ПЛАНОМ, ЗАДАНИЕМ, ЗАКАЗОМ НА ИЗДЕЛИЕ, вычислить количество собранных ИЗДЕЛИЙ в соответствии с РЕКОМЕНДАЦИЯМИ, технологией и нормами качества, и количество БРАКА, полученного в этой технологической операции. Оценить время выполнения СБОРКИ УЗЛОВ и СБОРКИ ИЗДЕЛИЯ.*

Модуль КОНТРОЛЬ КАЧЕСТВА ИЗДЕЛИЯ

***ВХОДНЫЕ ДАННЫЕ**: ИЗДЕЛИЕ, ЦЕНА.*

***ВЫХОДНЫЕ ДАННЫЕ**: ГОТОВОЕ ИЗДЕЛИЕ, РЕКОМЕНДАЦИИ, БРАК, ПОКАЗАТЕЛИ.*

***ВЫПОЛНЯЕМЫЕ ОПЕРАЦИИ:** определить количество ИЗДЕЛИЙ, прошедших тестирование и не прошедших тест и возвращенных на переработку. Найти процентное отношение. Реализовать ГОТОВЫЕ ИЗДЕЛИЯ по сформированной цене на продажу. На основе потока ЗАКЛЮЧЕНИЕ ПО ТЕСТИРОВАНИЮ разработать РЕКОМЕНДАЦИИ по изготовлению ДЕТАЛЕЙ, сборке УЗЛОВ и ИЗДЕЛИЙ.*

КОМПЛЕКСНАЯ ТЕХНОЛОГИЯ РАЗРАБОТКИ ИС SADT

На рынке средств разработки ИС достаточно много систем, в той или иной степени удовлетворяющих перечисленным требованиям. Рассматривается конкретная технология разработки, основывающаяся на решениях фирм Logic Works и Rational Software, которая является одной из лучших по критерию стоимость/эффективность.

Для проведения анализа и реорганизации бизнес-процессов Logic Works предлагает CASE - средства верхнего уровня - BPwin методологии IDEF0 (функциональная модель), IDEF3 (WorkFlow Diagram) и DFD (DataFlow Diagram). Функциональная модель предназначена для описания существующих бизнес-процессов на предприятии (модель AS-IS) и идеального положения вещей - того, к чему нужно стремиться (модель TO-BE).

Такая технология создания модели позволяет построить модель, адекватную предметной области на всех уровнях абстрагирования. Если в процессе моделирования нужно осветить специфические стороны технологии предприятия, ***BPwin позволяет переключиться на любой ветви модели на нотацию IDEF3 или DFD и создать смешанную модель***. Нотация DFD включает такие понятия как внешняя ссылка и хранилище данных, что делает ее более удобной (по сравнению с IDEF0) для моделирования документооборота. Методология IDEF3 включает элемент "перекресток", что позволяет описать логику взаимодействия компонентов системы.

На основе модели BPwin можно построить модель данных. ***Для построения модели данных Logic Works предлагает мощный и удобный инструмент - ERwin***. Хотя процесс преобразования модели BPwin в модель данных плохо формализуется и поэтому полностью не автоматизирован, ***Logic Works предлагает удобный инструмент для облегчения построения модели данных на основе функциональной модели - механизм двунаправленной связи BPwin - ERwin*** (рисунок 55 (1)).

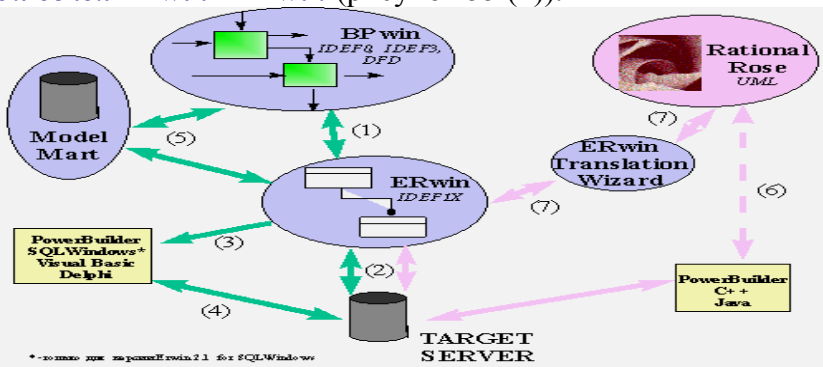


Рисунок 55 - Общая схема взаимодействия инструментальных средств

ERwin имеет два уровня представления модели - логический и физический. На логическом уровне данные представляются безотносительно конкретной СУБД, поэтому могут быть представлены наглядно. Физический уровень данных - это, по - существу, отображение системного каталога, который зависит от конкретной реализации СУБД. ERwin позволяет проводить процессы прямого и обратного проектирования БД (рисунок 55 (2)). Это означает, что по модели данных можно сгенерировать схему БД или автоматически создать модель данных на основе информации системного каталога. ERwin интегрируется с популярными средствами разработки клиентской части - PowerBuilder, SQLWindows, Visual Basic, Delphi (рисунок 55 (3)), что позволяет автоматически генерировать код приложения, который готов к компиляции и выполнению (рисунок 55 (4)).

Создание современных информационных систем, основанных на широком использовании распределенных вычислений, объединении традиционных и новейших информационных технологий, требует тесного взаимодействия всех участников проекта: менеджеров, бизнес и системных аналитиков, администраторов баз данных, разработчиков. Для этого средства моделирования и разработки должны быть объединены общей системой организации совместной работы. Фирма Logic Works разработала систему Model Mart - хранилище моделей, к которому открыт доступ для участников проекта создания информационной системы (рисунок 55 (5)). Model Mart удовлетворяет всем ***требованиям, предъявляемым к средствам разработки крупных информационных систем, а именно:***

- ***совместное моделирование***. Каждый участник проекта имеет инструмент поиска и доступа к интересующей его модели в любое время. При совместной работе используются три режима: ***незащищенный, защищенный и режим просмотра***. В режиме просмотра запрещается любое изменение моделей. В защищенном режиме модель, с которой работает один пользователь, не может быть изменена другими пользователями. В незащищенном режиме пользователи могут работать с общими моделями в реальном масштабе времени. Возникающие при этом конфликты разрешаются при помощи специального модуля - Intelligent Conflict Resolution (ICR). Model Mart позволяет сохранять множество версий, снабженных аннотациями, с последующим сравнением предыдущих и новых версий. При необходимости возможен возврат к предыдущим версиям;

- ***создание библиотек решений***. Model Mart позволяет формировать библиотеки стандартных решений, включающие наиболее удачные фрагменты реализованных проектов, накапливать и использовать типовые модели, объединяя их при необходимости "сборки" больших систем. На основе существующих баз данных с помощью ERwin возможно восстановление моделей (обратное проектирование), которые в процессе анализа могут объединяться с типовыми моделями из библиотек моделей;

- ***управление доступом***. Для каждого участника проекта определяются права доступа, в соответствии с которыми они получают возможность работать только с определенными моделями. Права доступа могут быть определены как для групп, так и для отдельных участников проекта.

Архитектура Model Mart. Model Mart реализована на архитектуре клиент - сервер. В качестве платформы реализации хранилища выбраны РСУБД Sybase, Microsoft SQL Server и Oracle. Клиентскими приложениями являются ERwin 3.x и BPwin 2.0x. В следующих версиях Model Mart предполагается открыть доступ к хранилищу моделей через API, что позволит включению новых инструментов моделирования и анализа.

Как было указано выше, ***при разработке крупных проектов критичным становится время реализации проекта.*** Одним из решений проблемы может стать автоматическая генерация кода приложения (клиентской части) CASE - средствами на основе модели предметной области. Хотя ERwin решает эту задачу, код генерируется на основе модели IDEF1X, то есть фактически на основе реляционной модели данных, которая непосредственно не содержит информацию о бизнес - процессах. Как следствие этого, сгенерированный код не может полностью обеспечить функциональность приложения со

сложной бизнес-логикой. ***Существует альтернативная технология кодогенерации, которая лишена этого недостатка - объектно-ориентированное проектирование, реализованное в Rational Rose*** (Rational Software). Rational Rose - позволяющее строить объектные модели в различных нотациях (OMT, UML, Буч) и генерировать на основе полученной модели приложения на языках программирования C++, Visual Basic, Power Builder, Java, Ada, Smalltalk и др. Поскольку генерация кода реализована на основе знаний предметной области, а не на основе реляционной структуры данных, полученный код более полно отражает бизнес-логику. Rational Rose поддерживает не только прямую генерацию кода, но и обратное проектирование, то есть создание объектной модели по исходному коду приложения (рисунок 55 (6)).

Rational Rose предназначен для генерации клиентской части приложения. Для генерации схемы БД объектную модель следует конвертировать в модель данных IDEF1X. Модуль ERwin Translation Wizard (Logic Works) позволяет перегружать объектную модель Rational Rose в модель данных ERwin (и обратно) и сгенерировать схему БД (рисунок 55 (7)). Таким образом, технологическая цепочка Rational Rose - ERwin Translation Wizard - ERwin позволяет реализовывать крупные проекты в технологии клиент - сервер.

СОЗДАНИЕ ФУНКЦИОНАЛЬНОЙ МОДЕЛИ В BPwin (IDEF0) И МОДЕЛИ ДАННЫХ В Erwin (IDEF1X)

На начальных этапах создания ИС необходимо понять, как работает организация, которую мы собираемся автоматизировать. Никто в организации не знает, как она работает в той мере подробности, которая необходима для создания ИС. Руководитель хорошо знает работу в целом, но не в состоянии вникнуть в детали работы каждого рядового сотрудника. Рядовой сотрудник хорошо знает, что творится на его рабочем месте, но плохо знает, как работают коллеги. Поэтому для описания работы предприятия необходимо построить функциональную модель и модель данных.

ГРУППОВАЯ РАЗРАБОТКА МОДЕЛЕЙ С ПОМОЩЬЮ Model Mart

Model Mart является системой групповой разработки крупных проектов, которая интегрирует инструментальные средства системных аналитиков и разработчиков БД. Одной из проблем, возникающей при многопользовательской работе с моделями, является разграничение прав доступа. Поскольку ModelMart является специализированным хранилищем моделей, помимо разграничения прав доступа на уровне модели возможно регулирование прав на уровне отдельных элементов модели. Для управления правами доступа в состав Model Mart включена утилита Model Mart Security Manager.

СОЗДАНИЕ ОБЪЕКТНОЙ МОДЕЛИ С ПОМОЩЬЮ Rational Rose

ДИАГРАММЫ

Применение CASE-технологий и CASE - средств ERWin и BPWin позволяет в несколько раз сократить время разработки информационных систем и значительно снизить вероятность появления ошибок за счет автоматизации начальных этапов разработки и автоматической генерации структуры сервера БД и кода клиентского приложения. Однако эта технология не лишена недостатков. Код клиентского приложения генерируется на основе информации о структуре БД. К структуре БД предъявляются определенные требования (нормализации), в результате чего данные хранятся в таблицах БД не всегда в той же форме, в которой они должны представляться на экранных формах. Другими словами, если код приложения генерируется не на основе описания предметной области, невозможно построить эффективное приложение со сложной бизнес-логикой.

Альтернативой структурному подходу стали лишенные перечисленных недостатков объектно-ориентированные методы разработки информационных систем. В начале девяностых годов был предложен разработанный на основе наиболее популярных объектных методов - OMT (Rumbaudh), Booch и OOSE (Jacobsom) универсальный язык объектного проектирования - Unified Modeling Language UML. Одним из CASE-средств, поддерживающих язык UML, является выпущенный фирмой Rational Software программный па-

кет Rational Rose, который позволяет генерировать код приложения, в полной мере отвечающий бизнес-правилам и с наименьшим риском.

Снижение риска в объектной технологии достигается за счет реализации технологии итерационной разработки (так называемая спиральная модель жизненного цикла разработки). Разработка состоит из ряда итераций, которые в дальнейшем приводят к созданию информационной системы. Поскольку тестирование проводится на каждой итерации, риск снижается уже на начальных этапах жизненного цикла разработки.

Модель представляет собой совокупность диаграмм, описывающих различные аспекты структуры и поведения информационной системы. Для просмотра модели в Rational Rose используется иерархический навигатор модели - Browser. Рассмотрим в общих чертах некоторые диаграммы UML.

Диаграммы использования системы (Use Cases) показывают основные функции, которые должны быть включены в систему (use case), их окружение (actors) и взаимодействие функций с окружением. Воздействующие объекты (actors) не являются частью системы - это конечные пользователи или другие программы, взаимодействующие с проектируемой информационной системой. Функциональность (use case) - последовательность действий, выполняемых системой, которые приводят к определенным результатам, необходимым для конкретного воздействующего объекта.

Диаграммы Use Cases включают отношения и ассоциации, показывающие взаимодействие между воздействующими объектами и функциями (изображаются в виде стрелок) и примечания (note), которые могут быть привязаны к любому объекту диаграммы Use Cases (рисунок 56).

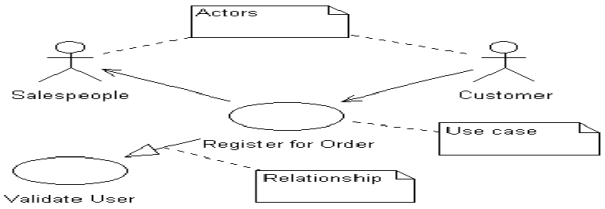


Рисунок 56 - Диаграмма Use Cases

Диаграммы классов. Под объектом в UML понимается некоторое абстрактное представление конкретного объекта предметной области. Каждый объект имеет состояние, поведение и индивидуальность. Например, объект "Проект" может иметь два состояния - "открыт" и "закрыт". Поведение объекта определяет, как объект взаимодействует с другими объектами. Индивидуальность означает, что каждый объект уникален и отличается от других объектов. Под классом понимается описание объектов, обладающих общими свойствами (атрибутами), поведением, общими взаимоотношениями с другими объектами и общей семантикой. Класс является шаблоном для создания новых объектов.

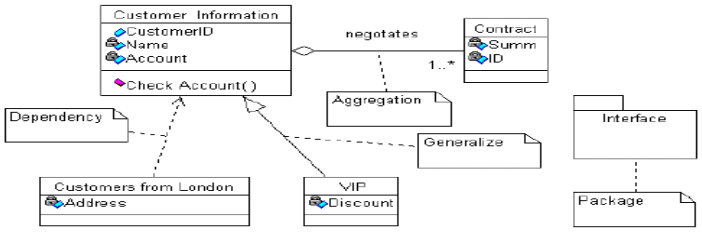


Рисунок 57 - Диаграмма классов

Каждый класс может иметь атрибуты (свойства). Так, на рисунке 57 класс Customer Information (информация о клиенте) имеет атрибуты CustomerID (идентификатор клиента), Name (имя) и Account (счет). Кроме того, каждый класс может иметь методы (operations) - некоторые действия, которые описывают поведение объектов класса. Класс Customer Information имеет метод Check Account.

Классы могут иметь взаимосвязи (relationship), называемые отношениями. В нотации UML имеется несколько типов отношений. Отношение использования (associations)

показывает, что объект одного класса связан с одним или несколькими объектами другого класса. Отношение включения (aggregation) является частным случаем отношения использования. Оно показывает, что один объект является частью другого. При воздействии на один объект, связанный отношением включения, некоторые операции автоматически могут затронуть другой объект. Например, класс Customer Information связан отношением включения с классом Contract. При удалении объекта класса Customer Information (информация о клиенте) должны удаляться все объекты класса Contract (относящиеся к данному клиенту контракты). Каждая связь может быть охарактеризована определенной фразой, называемой именем роли. Связь между классами Customer Information и Contract имеет имя negotiates. Каждая связь может иметь индикатор множественности, который показывает, сколько объектов одного класса соответствует объекту другого класса. Связь negotiates имеет индикатор 1..* (один или много).

Наследование (inher называемым стратегическим itance) описывает взаимосвязь между классами, когда один класс (называется подклассом, subclass) наследует структуру и/или поведение одного или нескольких классов. Подкласс VIP наследует свойства и поведение класса Customer Information. Связь классов в иерархии наследования называется отношением наследования (generalization).

Временная диаграмма (Sequences) демонстрирует поведение объектов во времени. Она показывает объекты и последовательность сообщений, посылаемых объектами. Сообщения на диаграмме сценариев изображаются в виде стрелок (рисунок 58).

Архитектура приложения описывается в диаграммах компонент (Component Diagram), которые описывают вхождение классов и объектов в программные компоненты системы (модули, библиотеки) и диаграммы развертывания (Deployment Diagram), при помощи которых документируется размещение программных модулей на узлах (физических и логических устройствах) системы.

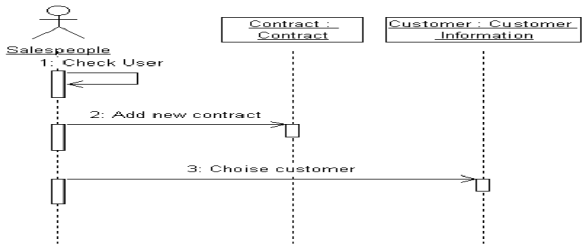


Рисунок 58 - Временная (*Sequence*)диаграмма

Генерация кода осуществляется на основе диаграмм классов.

При генерации кода Rational Rose включает строки комментария, начинающиеся последовательностью символов "//##". Сгенерированный код (в отличие от кода, сгенерированного ERWin) не является готовым приложением. Здесь генерируются лишь заголовки методов (Check_Account), сами методы необходимо дописывать вручную.

МОДЕЛЬ ДАННЫХ

Rational Rose позволяет строить объектную модель, но не может построить модель данных или сгенерировать системный каталог сервера БД. Для решения этой задачи фирмой Logic Works выпущена утилита ERWin Translation Wizard, позволяющая перегрузить объектную модель в ERWin и автоматически получить на ее основе модель данных. После инсталляции ERWin Translation Wizard вызывается из среды Rational Rose. Для того, чтобы классы могли быть конвертированы в сущности модели данных, они должны быть определены как Persistent. ERWin Translation Wizard позволяет как сгенерировать диаграмму классов на основе модели данных, так и модель данных на основе диаграммы классов. На рисунке 59 показана физическая модель данных, полученная на основе диаграммы классов, представленной на рисунке 57. Модель данных может быть использована для генерирования системного каталога сервера БД.

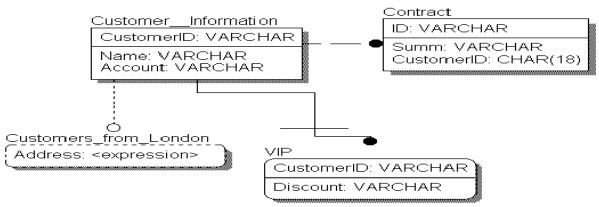


Рисунок 59 - Модель данных

В таблице 11 показано соответствие между объектами диаграммы классов и объектами модели данных при перегрузке моделей из Rational Rose в ERWin и обратно.

Таблица 11

Объекты диаграммы классов.	**Объекты модели данных.**
Класс (Class)	Сущность, таблица (Entity, Table)
Атрибут класса (Attribute)	Атрибут сущности, колонка (Attribute, Column)
Отношение использования (association) Отношение включения (aggregation)	Неидентифицирующая связь (Non-identifying relationship)
Отношение наследования (generalization)	Иерархия подкатегорий, полная подкатегория (Complete sub-category)
Имя роли (Role name)	Наименование связи (Verb phrases)
Индикатор множественности (multiplicity indicators)	Мощность связи (Cardinality)
Класс - клиент в отношении зависимости (Dependency relationship - Client)	Временная таблица (View)
Отношение зависимости (Dependency)	View relationship

КОМПЛЕКСНАЯ ТЕХНОЛОГИЯ РАЗРАБОТКИ ИС DATARUN

Одной из наиболее распространенных в мире методологий является методология DATARUN. В соответствии с методологией DATARUN ЖЦ ПО 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.

МЕТОДОЛОГИЯ DATARUN

Стадия формирования требований и планирования включает в себя действия по определению начальных оценок объема и стоимости проекта. Должны быть сформулированы требования и экономическое обоснование для разработки ИС, функциональные модели (модели бизнес-процессов организации) и исходная концептуальная модель данных, которые дают основу для оценки технической реализуемости проекта. Основными результатами этой стадии должны быть модели деятельности организации (исходные модели процессов и данных организации), требования к системе, включая требования по сопряжению с существующими ИС, исходный бизнес-план.

Стадия концептуального проектирования начинается с детального анализа первичных данных и уточнения концептуальной модели данных, после чего проектируется архитектура системы. Архитектура включает в себя разделение концептуальной модели на обозримые подмодели. Оценивается возможность использования существующих ИС. Выходными компонентами этой стадии являются концептуальная модель данных, модель архитектуры системы и уточненный бизнес-план.

На стадии спецификации приложений 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. В конце этой стадии принимается окончательное решение о способе реализации приложений. По результатам стадии должен быть построен проект ИС, включающий модели архитектуры ИС, данных, функций, интерфейсов (с внешними системами и пользователями), требований к разрабаты-

ваемым приложениям, требований к доработкам существующих ИС, требований к интеграции приложений, а также сформирован окончательный план создания ИС.

На стадии разработки, интеграции и тестирования должна быть создана 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 и приложений. Приложения интегрируются в систему, проводится тестирование приложений в составе системы и испытания системы. Основными результатами стадии являются готовые приложения, проверенные в составе системы на комплексных тестах, текущее описание конфигурации ПО, скорректированная по результатам испытаний версия системы и эксплуатационная документация на систему.

Стадия внедрения включает действия по установке и внедрению баз данных и приложений. Основными результатами стадии должны быть готовая к эксплуатации и перенесенная на программно-аппаратную платформу заказчика версия системы, документация сопровождения и акт приемочных испытаний по результатам опытной эксплуатации.

Стадии сопровождения и развития включают процессы и операции, связанные с регистрацией, диагностикой и локализацией ошибок, внесением изменений и тестированием, проведением доработок, тиражированием и распространением новых версий ПО, переносом приложений на новую платформу и масштабированием системы.

Методология DATARUN опирается на две модели или на два представления: модель организации; модель ИС.

Методология DATARUN базируется на системном подходе к описанию деятельности организации. 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 становятся основой для проектирования архитектуры ИС. Архитектура ИС будет более стабильной, если она основана на первичных данных, тесно связанных с основными деловыми операциями, определяющими природу бизнеса.

ИС (рисунок 60) представляет собой набор модулей, взаимодействующих с базами данных. 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.

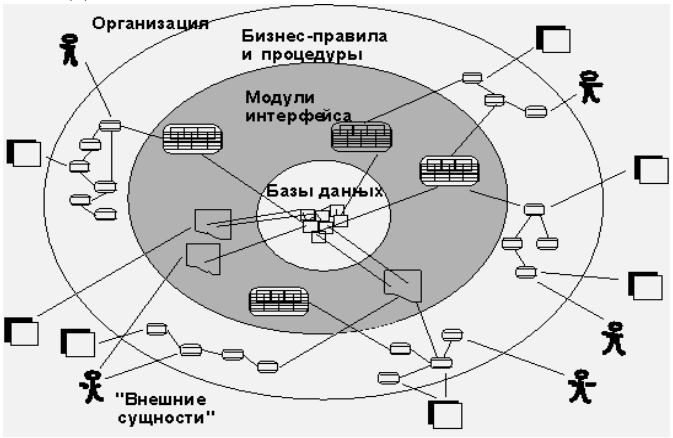


Рисунок 60 - Модель ИС

Подход DATARUN преследует две цели: определить стабильную структуру, на основе которой будет строиться ИС. Такой структурой является модель данных, полученная из первичных данных, представляющих фундаментальные процессы организации; спроектировать ИС на основе модели данных.

Объекты, формируемые на основе 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. На рисунке 61 представлена последовательность шагов проектирования ИС.

На рисунке 62 определены модели, создаваемые в процессе разработки ИС. 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. 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.

Информационная система создается последовательным построением ряда моделей, начиная с модели бизнес-процессов и заканчивая моделью программы, автоматизирующей эти процессы.

Рисунок 61 - Последовательность шагов проектирования системы

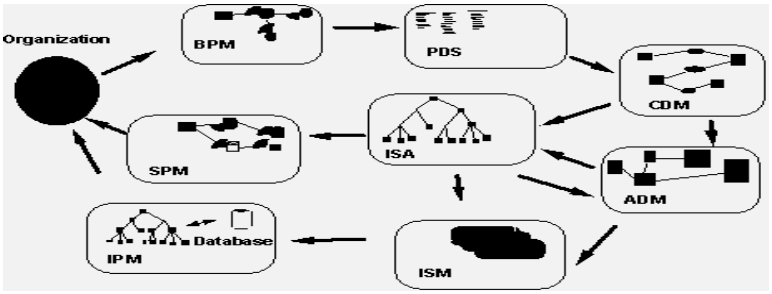


Рисунок 62 - Модели, создаваемые с помощью подхода DATARUN

BPM (Business Process Model) - модель бизнес-процессов;
PDS (Primary Data Structure) - структура первичных данных;
CDM (Conceptual Data Model) - концептуальная модель данных;
SPM (System Process Model) - модель процессов системы;
ISA (Information System Architecture) - архитектура информационной системы;
ADM (Application Data Model) - модель данных приложения;
IPM (Interface Presentation Model) - модель представления интерфейса;
ISM (Interface Specification Model) - модель спецификации интерфейса.

Создаваемая ИС должна основываться на функциях, выполняемых организацией. ***Поэтому первая создаваемая модель - это модель бизнес-процессов***, построение которой осуществляется в модуле Silverrun BPM. Для этой модели используется специальная нотация BPM. В процессе анализа и спецификации бизнес-функций выявляются ос-

новные 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.

В процессе обследования работы организации выявляются и документируются структуры первичных данных. Эти структуры заносятся в репозиторий модуля BPM при описании циркулирующих в организации документов, данных. В модели бизнес-процессов первичные структуры данных связаны с потоками и хранилищами информации.

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 в хорошо структурированном нормализованном виде.

На основе модели бизнес-процессов и концептуальной модели данных проектируется архитектура ИС. Определяются входящие в систему приложения, для каждого приложения специфицируются используемые данные и реализуемые функции. Архитектура ИС 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.

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.

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.

Приложение состоит из интерфейсных объектов (экранных форм, отчетов, процедур обработки данных). Каждый интерфейс системы (экранная форма, отчет, процедура обработки данных) 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.

Модель представления интерфейса - это описание внешнего вида интерфейса, как его видит конечный пользователь системы. Это может быть 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.

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.

Далее, с помощью средств разработки приложений происходит физическое создание системы: приложения интегрируются в информационную систему.

СРЕДА РЕАЛИЗАЦИИ

Инструментальное средство SE Companion является средой, в которой реализован электронный вариант методологии DATARUN. Оно позволяет: создать гипертекстовое описание методологии в виде иерархии описания стадий, этапов и операций разработки; создать гипертекстовое описание методик реализации процессов ЖЦ ПО; выделять из гипертекстового описания иерархию процессов ЖЦ ПО для планирования и управления процессом создания ПО (иерархию работ); изменять гипертекстовые описания ЖЦ и методов как это необходимо разработчику, производить авторизацию методологии и отслеживать эти изменения в иерархии работ, предназначенных для управления проектом; привязать к процессам ЖЦ инструментальные средства поддержки этих процессов и обеспечить вызов инструментальных средств из соответствующих экранов гипертекстового справочника; обеспечить просмотр гипертекстовых экранов описания используемых методов из инструментальных средств; обеспечить поддержку процесса управления разработкой за счет взаимодействия со средством планирования работ MS Project, оценивания трудоемкости проекта, отслеживания выполнения работ, создания графиков работ и др.

Особенно важными являются возможность авторизации методологии и интерактивный доступ любого разработчика к описанию любого метода или процесса в нужный ему момент времени. На современном этапе развития технологии, в условиях быстрого изменения как программных и аппаратных средств, так и задач бизнеса, методология создания, сопровождения и развития ПО должна иметь возможность изменяться и настраиваться на новые технологии, методы и инструментальные средства.

В SE Companion исходным документом, описывающим методологию (как процессы ЖЦ, так и все сопутствующие методы и методики), является файл в формате MS Word. Это обеспечивает возможности для описания методологии с любой степенью детализации, проведения разметки для создания гипертекста и авторизации методологии в принятом стандартном формате.

Гипертекстовое описание методологии и технологии создания ПО строится из описания процессов жизненного цикла, методов и методик и представляет собой единый гипертекстовый документ в формате MS Help. Все изменения методологии производятся посредством корректировки и дополнительной разметки исходного документа.

Описание методологии создания системы обычно состоит из раздела описания процессов ЖЦ и разделов описания методов и методик. В свою очередь, раздел описаний процессов состоит из иерархии описаний стадий, этапов и операций жизненного цикла с обязательным описанием выходных компонентов каждого процесса. Компоненты ПО создаются с применением методик и методов, описываемых в соответствующих разделах.

ПРОЕКТИРОВАНИЕ ИС С ПОМОЩЬЮ Silverrun+JAM

CASE - средство Silverrun фирмы Computer Systems Advisers используется для анализа и проектирования ИС бизнес - класса и ориентировано на спиральную модель ЖЦ. Оно применимо для поддержки любой методологии, основанной на раздельном построе-

нии функциональной и информационной моделей (диаграмм потоков данных и диаграмм "сущность-связь"). Настройка на конкретную методологию обеспечивается выбором требуемой графической нотации моделей и набора правил проверки проектных спецификаций. В системе имеются готовые настройки для наиболее распространенных методологий: DATARUN Gane/Sarson, Yourdon/DeMarco, Merise, Ward/Mellor, Information Engineering. Архитектура Silverrun позволяет наращивать среду разработки по мере необходимости.

Структура и функции

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.

Модуль построения моделей бизнес-процессов в форме диаграмм потоков данных (BPM-Business Process Modeler) позволяет создавать модель организации или создаваемой ИС. 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 нотациях, включая Yourdon/DeMarco и Gane/Sarson. Имеется возможность создавать собственные нотации, добавлять в число изображаемых на схеме дескрипторов определенные пользователем поля.

Модуль концептуального моделирования данных (ERX-Entity-Relationship eXpert) 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.

Модуль реляционного моделирования (RDM-Relational Data Modeler) 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. 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.

Менеджер репозитория рабочей группы (WRM-Workgroup Repository Manager) применяется как словарь данных для хранения общей для всех моделей информации, а также обеспечивает интеграцию модулей Silverrun в единую среду проектирования.

Платой за высокую гибкость и разнообразие изобразительных средств построения моделей является такой недостаток Silverrun, как отсутствие жесткого взаимного контроля между компонентами различных моделей (например, возможности автоматического распространения изменений между DFD различных уровней декомпозиции). Следует, однако, отметить, что этот недостаток может иметь существенное значение только в случае использования каскадной модели ЖЦ ПО.

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.

При использовании моста Silverrun расширяет свой внутренний репозиторий специфичными для целевой системы атрибутами. После определения значений этих атрибутов генератор приложений переносит их во внутренний каталог среды разработки или использует при генерации кода на языке SQL. Таким образом, можно полностью определить ядро базы данных с использованием всех возможностей конкретной СУБД: триггеров, хранимых процедур, ограничений ссылочной целостности. При создании приложения на языке 4GL данные, перенесенные из репозитория Silverrun, используются либо для автоматической генерации интерфейсных объектов либо для быстрого их создания вручную.

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***: с***истема отчетов***. Можно, определив содержимое отчета по репозиторию, выдать отчет в текстовый файл; с***истема 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 другими CASE-средствами, СУБД, текстовыми редакторами и электронными таблицами; х***ранение репозитория во внешних файлах*** через ODBC-драйверы. Для доступа к данным репозитория из наиболее распространенных СУБД 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; сетевая версия Silverrun позволяет осуществлять одновременную групповую работу с моделями, хранящимися в сетевом репозитории на базе СУБД Oracle, Sybase или Informix, несколько разработчиков могут работать с одной и той же моделью.

Среда функционирования

Имеются реализации Silverrun 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.

РАЗРАБОТКА ПРИЛОЖЕНИЙ С JAM

Средство разработки приложений JAM 7(JYACC's Application Manager) - продукт фирмы JYACC (США). В настоящее время готовится к выходу JAM 8.

Основной чертой JAM является его соответствие методологии RAD, 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.

Структура и функции

JAM имеет модульную структуру и состоит из следующих компонент:

- ядро системы; 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.

Ядро системы 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 основные

компоненты: редактор экранов (в состав редактора экранов входит среда разработки экранов, визуальный репозиторий объектов, собственная СУБД JAM - JDB, менеджер транзакций, отладчик, редактор стилей); редактор меню; набор вспомогательных утилит; средства изготовления промышленной версии приложения.

При использовании JAM разработка внешнего интерфейса приложения представляет собой визуальное проектирование и сводится к созданию экранных форм путем размещения на них интерфейсных конструкций и определению экранных полей ввода/вывода информации. Проектирование интерфейса в JAM осуществляется с помощью редактора экранов. Приложения, разработанные в JAM, имеют многооконный интерфейс. Разработка отдельного экрана заключается в размещении на нем интерфейсных элементов, возможной (но не обязательной) их группировке и конкретизации различных их свойств, включающих визуальные характеристики (позиция, размер, цвет, шрифт), поведенческие характеристики (многообразные фильтры, форматы, защита от ввода) и ряд свойств, ориентированных на работу с БД.

Редактор меню позволяет разрабатывать и отлаживать системы меню. Реализована возможность построения пиктографических меню (toolbar). Назначение каждого конкретного меню тому или иному объекту приложения осуществляется в редакторе экранов.

В ядро JAM встроена однопользовательская реляционная СУБД JDB. Основным назначением JDB является прототипирование приложений в тех случаях, когда работа со штатной СУБД невозможна или нецелесообразна. В JDB реализован необходимый минимум возможностей реляционных СУБД за исключением индексов, хранимых процедур, триггеров и представлений (view). С помощью JDB можно построить БД, идентичную целевой БД и разработать значительную часть приложения. Отладчик позволяет проводить комплексную отладку разрабатываемого приложения. Осуществляется трассировка всех событий, возникающих в процессе исполнения приложения.

Утилиты JAM включают три группы: конверторы файлов экранов JAM в текстовые. JAM сохраняет экраны в виде двоичных файлов собственного формата. В ряде случаев (для изготовления программной документации проекта) необходимо текстовое описание экранов; конфигурирование устройств ввода/вывода. JAM и приложения, построенные с его помощью, не работают непосредственно с устройствами ввода/вывода. Вместо этого JAM обращается к логическим устройствам ввода/вывода (клавиатура, терминал, отчет). Отображение логических устройств в физические осуществляется с помощью средств конфигурирования; обслуживание библиотек экранов (традиционные операции с библиотеками).

Одним из дополнительных модулей JAM является генератор отчетов. Компоновка отчета осуществляется в редакторе экранов JAM. Описание работы отчета осуществляется с помощью специального языка. Генератор отчетов позволяет определить данные, выводимые в отчет, группировку выводимой информации, форматирование вывода и др.

Приложения, разработанные с использованием JAM, не требуют так называемых исполнительных (run-time) систем и могут быть изготовлены в виде исполняемых модулей. Для этого разработчик должен иметь компилятор С и редактор связей. Для изготовления промышленной версии в состав JAM входит файл сборки (makefile), исходные тексты (на языке С) ряда модулей приложения и необходимые библиотеки.

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, совместимым по вызовам функций с языком С.

С точки зрения реализации логики приложения JAM является событийно-ориентированной системой. В JAM определен набор событий, включающий открытие и

закрытие окон, нажатие клавиши клавиатуры, срабатывание системного таймера, получение и передача управления каждым элементом экрана. Разработчик реализует логику приложения путем определения обработчика каждого события. Например, обработчик события "нажатие кнопки на экране" (мышью или с помощью клавиатуры) может открыть следующее экранное окно. Обработчиками событий в JAM могут быть как встроенные функции JAM, так и функции, написанные разработчиком на C или JPL. Набор встроенных функций включает в себя более 200 функций различного назначения. Встроенные функции доступны для вызовов из функций, написанных как на JPL, так и на C.

Промышленная версия приложения, разработанного с помощью JAM, включает в себя следующие компоненты: исполняемый модуль интерпретатора приложения. В этот модуль могут быть встроены функции, написанные разработчиками на языках 3-го поколения; экраны, составляющие само приложение (могут поставляться в виде отдельных файлов, в составе библиотек экранов или же быть встроены в тело интерпретатора); внешние JPL-модули. Могут поставляться в виде текстовых файлов или в прекомпилированном виде, причем прекомпилированные внешние JPL-модули могут быть как в виде отдельных файлов, так и в составе библиотек экранов; файлы конфигурации приложения - файлы конфигурации клавиатуры и терминала, файл системных сообщений, файл общей конфигурации.

Взаимодействие 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, так называемых QBE (Query By Example - запросы по образцу), 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.). Может потребоваться лишь "перерисовать" статические текстовые поля на экранах с русским текстом при переносе между средами DOS-Windows-UNIX. Кроме того, переносимость облегчается тем, что в JAM приложения разрабатываются для виртуальных устройств ввода/вывода, а не для физических. Таким образом, 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 условии переноса структуры самой БД в ряде случаев приложения могут не требовать никакой модификации, за исключением инициализации сеанса работы. Такая ситуация может сложиться в том случае, если в приложении не использовались специфические для той или иной СУБД расширения SQL.

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. Компоненты JAM/TPi-Client и JAM/TPi-Server позволяют достаточно просто перейти на трехзвенную модель. При этом ключевую роль играет модуль JAM/TPi-Server, так как основная трудность внедрения трехзвенной модели заключается в реализации логики приложения в сервисах менеджеров транзакций.

Интерфейс JAM/CASE подобен интерфейсу к СУБД и позволяет осуществить обмен информацией между репозиторием объектов JAM и репозиторием CASE-средства аналогично тому, как структура БД импортируется в репозиторий JAM непосредственно из БД. Отличие заключается в том, что в случае интерфейса к CASE этот обмен является двунаправленным. Кроме модулей JAM/CASEi, существует также модуль JAM/CASEi Developer's Kit. С помощью этого модуля можно самостоятельно разработать интерфейс (т.е. специализированный модуль JAM/CASEi) для конкретного CASE-средства, если готового модуля JAM/CASEi для него не существует.

Мост (интерфейс) Silverrun-RDM <-> JAM реализует взаимодействие между CASE-средством Silverrun и JAM (перенос схемы базы данных и экранных форм приложения между CASE-средством Silverrun-RDM и JAM версии 7.0). ***Данный программный продукт имеет 2 режима работы:***

- ***прямой режим*** (Silverrun-RDM->JAM) предназначен для создания объектов CASE-словаря и элементов репозитория JAM на основе представления схем в Silverrun-RDM. В этом режиме мост позволяет, исходя из представления моделей данных интерфейса в Silverrun-RDM, производить генерацию экранов и элементов репозитория JAM. Мост преобразует таблицы и отношения реляционных схем RDM в последовательность объектов JAM соответствующих типов. Методика построения моделей данных интерфейса в Silverrun-RDM предполагает применение механизма подсхем для прототипирования экранов приложения. По описанию каждой из подсхем RDM мост генерирует экранную форму JAM;

- ***обратный режим*** (JAM->Silverrun-RDM) предназначен для переноса модификаций объектов CASE-словаря в реляционную модель Silverrun-RDM.

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.

Использование PVCS является более предпочтительным по сравнению с SCCS, так как позволяет организовать единый архив модулей проекта для всех платформ. Так как JAM на платформе UNIX не имеет прямого интерфейса к архивам PVCS, то выборка модулей из архива и возврат их в архив производятся с использованием PVCS Version Manager. На платформе MS-Windows JAM имеет встроенный интерфейс к PVCS и действия по выборке/возврату производятся непосредственно из среды JAM.

Среда функционирования

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 операционной системой.

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОЕКТИРОВАНИЕ

Разработанная универсальная нотация для моделирования объектов (UML - Unified Modeling Language) претендует на роль стандарта в области объектно-ориентированного анализа и проектирования. Конкретный вариант Rational Rose определяется языком, на

котором генерируются коды программ (C++, Smalltalk, PowerBuilder, Ada, SQLWindows и ObjectPro). Основной вариант - Rational Rose/C++ - позволяет разрабатывать проектную документацию в виде диаграмм и спецификаций, а также генерировать программные коды на C++. Кроме того, Rational Rose содержит средства реинжиниринга программ, обеспечивающие повторное использование программных компонент.

Структура и функции

В основе работы Rational Rose лежит построение различного рода диаграмм и спецификаций, определяющих логическую и физическую структуры модели, ее статические и динамические аспекты. В их число входят диаграммы классов, состояний, сценариев, модулей, процессов.

В составе Rational Rose можно выделить 6 основных структурных компонент: репозиторий, графический интерфейс пользователя, средства просмотра проекта (browser), средства контроля проекта, средства сбора статистики и генератор документов. К ним добавляются генератор кодов (для каждого языка) и анализатор для C++, обеспечивающий реинжиниринг - восстановление модели проекта по исходным текстам программ.

Репозиторий 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.

Средства автоматической генерации кодов программ на языке C++, 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 C++. Анализатор кодов C++ 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 C++. 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. Можно определить типы исходных файлов, базовый компилятор, задать информацию для включения в формируемую модель, какие элементы выходной модели следует выводить на экран. Таким образом, Rational Rose/C++ обеспечивает возможность повторного использования программных компонент.

В результате разработки проекта с помощью CASE-средства Rational Rose ***формируются следующие документы***: диаграммы классов, состояний, сценариев, модулей, 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.

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.

Взаимодействие с другими средствами и организация групповой работы

Rational Rose интегрируется со средством PVCS для организации групповой работы и управления проектом и со средством SoDA - для документирования проектов. Интеграция Rational Rose и SoDA обеспечивается средствами SoDA.

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.

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 и случайные воздействия окажутся невозможными.

Среда функционирования

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 80386SX или выше, память 8Мб (рекомендуется 12Мб), пространство на диске 8Мб + 1-3Мб для одной модели;
- платформа UNIX - память 32+(16*число пользователей) Мб, пространство на диске 30Мб + 20 при инсталляции + 1-3Мб для одной модели.

БИЗНЕС – ПРОЦЕССЫ И ЛОГИСТИКА

ОПРЕДЕЛЕНИЕ

“***Логистика - уникальная область человеческой деятельности: здесь никогда не бывает остановок. Логистикой занимаются повсюду в мире по 24 часа в сутки. Лишь немногие сферы деловых операций могут похвастаться той же сложностью внутренних взаимосвязей и такой же широтой географического охвата, какие характерны для логистики. Предназначение логистики — обеспечить получение продуктов и услуг там, где они необходимы, тогда, когда они требуются”***

Логистика - это наука о рациональной организации производства и распределения, которая комплексно с системных позиций охватывает вопросы снабжения предприятия сырьем, топливом, материалами, полуфабрикатами, об организации сбыта, распределении и транспортировке готовой продукции. Логистика как наука устанавливает связь между запасами, вместимостью, производительностью и гибкостью системы; позволяет преодолеть инерциальные процессы при переходе от частично оптимальных к полностью оптимальным системам.

"Логистика - наука об управлении информационными и материальными потоками в процессе движения товаров".

“***Логистика — это процесс*** планирования и обеспечения (включая контроль) эффективного и непрерывного поступления товаров, услуг и сопутствующей информации оттуда, где они создаются, к потребителю, направленный на всемерное удовлетворение потребительских запросов”.

С позиции менеджмента организации логистику можно рассматривать как стратегическое управление материальными потоками в процессе закупки, снабжения, перевозки и хранения материалов, деталей и готового продукта. Понятие включает в себя также управление соответствующими потоками информации, а также финансовыми потоками.

Логистика направлена на оптимизацию издержек и рационализацию процесса производства, сбыта и сопутствующего сервиса как в рамках одного предприятия, так и для группы предприятий.

Логистика — это взгляд (мировоззрение) на все бизнес-процессы предприятия через призму издержек, с целью их оптимизации, контроля и управления ими.

Содержанием логистики как науки является установление причинно-следственных связей и закономерностей, присущих процессу товародвижения, в целях определения и реализации на практике эффективных организационных форм и методов управления материальными и информационными потоками.

Хотя логистика рассматривает проблему управления экономической деятельностью как единое целое, вследствие различного физического характера управляемых материальных и нематериальных потоков выделяют функциональные разделы или ***области логистического управления:*** логистика грузовых перевозок; банковская логистика; заготовительная логистика; логистика распределения (распределительная или сбытовая логистика); логистика поставок; логистика сервиса; морская логистика; промышленная логистика; логистика товародвижения; логистика сбыта; логистика снабжения; глобальная логистика; логистика запасов; транспортная логистика; коммерческая логистика; маркетинговая логистика; закупочная логистика (логистика закупок); логистика производственных процессов (производственная логистика); логистика складирования (складская логистика); информационная логистика. Большинство из них взаимно пересекаются.

Пример. Основными задачами транспортной логистики являются: обеспечение грузоперевозки товаров; выбор вида транспортного средства; минимизация затрат на грузоперевозку; нахождение оптимальных маршрутов по грузовой перевозке; поиск и выбор экспедиторов, транспортной компании – перевозчика; грамотное оформление и заключение договоров и контрактов на грузоперевозку, куплю - продажу товаров; прохождение таможни; обеспечение оптимальной схемы доставки грузов при перевозках; консолидация (объединение) грузов в процессе доставки; хранение на складах; согласование прочих вопросов с грузоотправителем и грузополучателем и т.д.

Эти задачи представляются задачами исследования операций: задача о ранце, задача коммивояжёра, транспортная задача, задача об упаковке в контейнеры, задачи составления расписания, диспетчеризации и другие.

Пример. Грузоподъемность автомобиля - 10 тонн.
Статистический коэффициент использования грузоподъемности - 0,8.
Коэффициент использования пробега - 0,75.
Продолжительность поездки - 2,5 часа.
Продолжительность простоя под погрузкой и разгрузкой за поездку - 30 минут (0,5 часа).
Техническая скорость - 60 км/час.
Время работы автомобиля за смену - 15 часов.

Определить необходимое число автомобилей для перевозки 480 тонн.

Для того, чтобы перевести 10 тонн груза автомобилю потребуется 3 часа (2,5 часа поездка и 0,5 часа разгрузка – погрузка).

За смену машина сможет сделать 5 ходок (15/3).

Но с учетом статистического коэффициента использования грузоподъемности 0,8 машина перевезет только 40 тонн. (50*0,8).

С учетом же коэффициента использования пробега 0,75 машина реально перевезет только 30 тонн. (40*0,75).

Общее количество автомобилей рассчитаем как соотношение имеющегося груза на объем груза, который может перевезти одна машина за смену.

Получим 16 автомобилей (480/30).

Принципиальная новизна логистического подхода - органичная взаимная связь, интеграция вышеперечисленных областей в единую систему.

Цель логистического подхода - сквозное управление материальными потоками. Управление материальными потоками всегда является существенной стороной бизнес-процессов. Однако лишь сравнительно недавно оно приобрело положение одной из наиболее важных функций экономической жизни. Основная причина - переход от рынка про-

давца к рынку покупателя, вызвавший необходимость гибкого реагирования производственных и торговых систем на быстро изменяющиеся приоритеты потребителей.

Построение и функционирование логистической системы основывается на реализации принципа системного подхода, который проявляется в первую очередь в интеграции и четком взаимодействии всех элементов логистической системы. Данный принцип находит свое отражение в разработке и осуществлении единого технологического процесса производственно - транспортной системы, в переходе от конструирования отдельных видов оборудования, к созданию комплексных производственно – складских и производственно – транспортных систем. Системный подход открывает новые возможности для сокращения продолжительности и оптимизации производственного цикла, повышения производительности во всех звеньях логистической системы;

В зависимости от специфики деятельности компании применяются различные логистические системы. ***Логистическая система — совокупность действий участников логистической цепи (предприятий-производителей, транспортных, торговых организаций), построенных так, чтобы выполнялись основные задачи логистики***.

Логистические системы очень разнообразны по охвату деятельности предприятия. По своему назначению (уменьшение затрат при выполнении плановых заданий и повышение эффективности производственной деятельности) логистические системы должны охватывать практически все направления деятельности.

Для устойчивости функционирования системы первостепенное значение имеет достоверное планирование производства сбыта и распределения, причем предпочтение отдается стратегическому планированию по отношению к оперативному. С целью достижения высокой надежности такого планирования необходимо изучение поведения внешней среды и, прежде всего, рынка, идентификация возможных ситуаций и получение стратегических ответов на возникшие в связи с этим вопросы.

Основная концепция построения логистической системы основывается на принципе четкого взаимодействия и согласованности функциональных элементов. К ним можно отнести объекты производства и потребления продукции, объемы ее поставок (транзитом и через склады), наличие и потребность в складских мощностях для ее хранения, объемы требуемых капитальных вложений и т.д.

При таком подходе в рамках логистической системы интегрируются функции производства, снабжения и сбыта. Такая система решает в глобальном плане следующие основные вопросы: производство продукции, выработка общей концепции товародвижения, выбор рациональных материальных потоков, определение объема запасов, объема требуемых складских мощностей для их хранения, необходимость их расширения или нового строительства, необходимый объем капитальных вложений для увеличения производства продукции.

ВИДЫ ЛОГИСТИЧЕСКИХ СИСТЕМ

Комплексная логистика — системный подход к организации всего цикла жизни товара и связанных с ним мероприятий в период от момента производства его комплектующих до момента потребления. Это эффективная система управления материальными, информационными и финансовыми потоками, связанными с жизненным циклом товара. Комплексный подход к логистическому процессу позволяет сократить риски неопределенности, под влиянием которых находится функциональный цикл жизни товара.

Закупочная логистика. Основной целью закупочной логистики является удовлетворение производства материалами с максимальной экономической эффективностью, качеством и кратчайшими сроками. Закупочная логистика проходит по поиску и выбору альтернативных поставщиков-изготовителей.

Сбытовая логистика. Целью сбытовой логистики (логистики распределения) является обеспечение доставки товаров в нужное место в нужное время с определенными

затратами. С этим понятием тесно связано понятие канал распределения — совокупность различных организаций, которые осуществляют доставку товара до потребителя.

Транспортная логистика. Транспортная логистика - это система по организации доставки каких-либо материальных предметов из одной точки в другую по оптимальному маршруту.

Логистика запасов. Политика управления запасами состоит из решений — что закупать или производить, когда и в каких объемах. Она также включает в себя решения о размещении запасов на производственных предприятиях и в распределительных центрах.

Складская логистика. Основной задачей складской логистики является оптимизация бизнес-процессов приемки, обработки, хранения и отгрузки товаров на складах. Для информационно-технической поддержки таких процессов используются специализированные системы управления складом WMS.

Информационная логистика - совокупность действий по эффективному распределению информационных потоков между цифровыми и традиционными носителями.

Неологистика — логистика второго поколения, для которой ***на первом этапе*** характерно преобладание комплексного подхода к развитию систем логистики на основе всего предприятия, исходя из общей цели: достижения максимальной эффективности работы всего предприятия.

Второй этап неологистики получил новое развитие в виде концепции “общей ответственности”. Критерием концепции “общей ответственности” является максимальное соотношение выхода и затрат. При этом подходе логистические системы вышли за пределы экономической сферы и стали учитывать социальные, экологические и политические аспекты. По новой точке зрения логистика с уровня фирм может перейти на отрасль и даже на уровень народного хозяйства. Категории глобальной логистики - логистика “стройного” (бережливого) производства и сетевая логистика (оперативная связь с партнерами, клиентами и покупателями всех уровней) также относятся ко второму этапу неологистики.

МАРКЕТИНГ И ЛОГИСТИКА

Оптимизация бизнес-процессов, как и организация компании, это и маркетинг, и логистика.

Маркетинг – вид экономической деятельности человека, направленный на удовлетворение нужд и потребностей путем обмена. Это определение верно для любой рыночно ориентированной компании. Таким образом, мнение, что маркетинг занимается “сбытом” и “исследованиями рынка» - это только одно направление деятельности маркетинговых подразделений.

Логистика - “общая стратегия хозяйственной деятельности или стратегия оптимизации материальных и информационных потоков, необходимая для обеспечения конкурентоспособности компании”.

Логистика, как наука, занимает ведущую роль в рационализации и автоматизации производства. Таким образом, логистика, являясь по сути одним из средств конкурентной борьбы и реализующаяся в зависимости от ситуации на рынке, организационно отделенная от маркетинговой деятельности по определению остается с маркетингом в теснейшей связке функционально (рисунок 63).

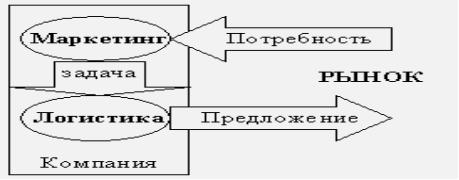


Рисунок 63 – Логистика маркетинга

Очевидно, что маркетинг планирует процессы организационно, а службы логистики занимаются оптимизацией (для обеспечения равновесного состояния между всеми подразделениями компании: снабжение, транспорт, производство, склад, сбыт). При этом

важнейшей задачей логистики является разделение функций организации, оптимизации и управления процессами, то есть логистика в связке с маркетингом становится логистикой маркетинга, становится одним из ключевых факторов успеха на рынке. Отдача от логистики выступает в виде показателя всей хозяйственной деятельности компании – достижение уровня обслуживания потребителей, заданного маркетингом.

Такая система взаимодействия маркетинга и логистики (маркетинговые подразделения планируют потоковые процессы организационно, а службы логистики занимаются их оптимизацией), позволяет полнее раскрыть отдачу от логистики как основной результат деятельности всей системы движения товаров в виде показателя уровня обслуживания клиентов.

Таким образом, реальные аспекты функционирования логистической системы управления потоковыми процессами включают в себя ***три случая оптимизации управления: при равновесном состоянии производства, потребления и спроса; при изменении интенсивности производства, потребления и спроса; при постоянно действующем запаздывании реакции отдельных подразделений на изменение внешней среды***.

Из этого вытекает, что основная функция логистики как инструмента маркетинга состоит в обеспечении своими средствами и методами условий покупок и продаж, определенных маркетинговыми службами. Это выражается в поддержании высокого уровня обслуживания потребителей при изменении внешней и внутренней среды.

Маркетинг диктует, какой следует быть логистике. Важнейший стратегический вопрос заключается в том, чтобы найти такие комбинации услуг к уровню сервиса, которые содействовали бы заключению прибыльных сделок.

ПРОЕКТ ОПТИМИЗАЦИИ БИЗНЕС- ПРОЦЕССОВ

Речь идет об оптимизации бизнес-процессов складского комплекса в одной из российских компаний, занимающейся продажей строительных материалов, и демонстрации логики построения эффективной модели взаимодействия маркетинга и логистики. При проведении маркетингового исследования и анализа рынка оказалось, что одним из ключевых факторов, который может оказать существенное влияние на конкурентоспособность предложения компании, является отсутствие очередей на складе и быстрота обслуживания. Причем, среди причин, отмечаемых клиентами, были отсутствие собственного грузового транспорта (приходится арендовать машины с почасовой оплатой), то есть клиент платит деньги за простой в очереди, что сказывается на несвоевременной доставке материалов.

Таким образом, реализуется модель взаимодействия маркетинга и логистики путем постановки задачи для логистики: провести оптимизацию процессов обслуживания клиентов на складском комплексе, при этом критерием оптимизации является минимизация очереди и времени обслуживания (клиент должен обслуживаться не более 20 минут).

Отдел логистики (совместно с технической службой) занялся поиском оптимальных путей для решения проблемы равновесного состояния системы товарных потоков в рамках задач определенных отделом маркетинга.

Поэтому проект был разбит на следующие этапы.

Описание процессов “AS-IS”

Составление “фотографии рабочего дня”. На этом этапе проходит измерение затрачиваемого времени на каждую функцию.

Анализ проблем и выявление их причин. Результатом этого является составление схем бизнес-процессов, имеющих место на складском комплексе, которые связаны с обслуживанием клиентов. В итоге были отмечены следующие результаты: при отгрузке товара клиентам (рисунок 64), что занимает в среднем 55 минут, 25% времени процесса занимает ожидание в очереди и 28% - поиск товара при погрузке.

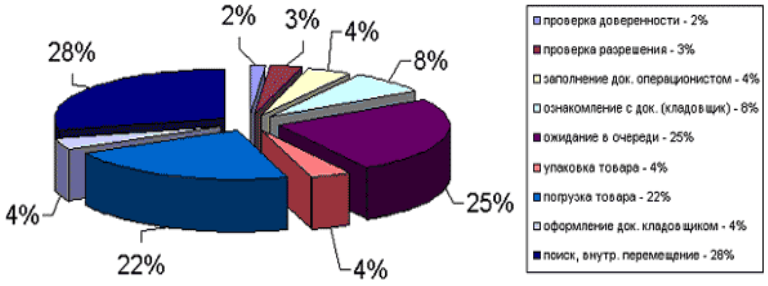


Рисунок 64 – Анализ маркетинга

В тоже время, если посмотреть на график загрузки склада (рисунок 65), то станет очевидным крайняя неравномерность нагрузки. Это является с одной стороны следствием отсутствия регламента процессов в течение рабочего дня, а с другой стороны отсутствием связи между складскими процессами и процессами поставок и продаж (между подразделениями склада, поставок и сбыта).

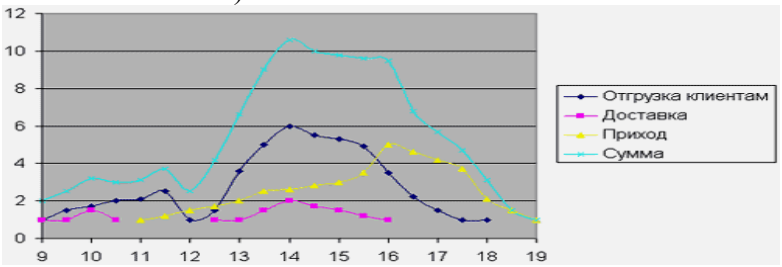


Рисунок 65 – График загрузки склада

Из приведенной диаграммы видно, что максимальная загрузка склада приходится на время с 13-30 до 16-30 часов, в остальное время объемы значительно ниже.

Основные проблемы, выявленные в процессе анализа - отсутствие необходимой организации, персонала и техники, способной обслужить клиентов на складе.

Причины: отсутствие регламента процессов по доставкам и перевозкам со стороны компании, что вызывает “пиковые” нагрузки; отсутствие деления склада на зоны разгрузки, приемки, основного хранения, комплектации заказа и отгрузки, поэтому, с точки зрения клиента функции погрузки сложны и выполняются медленно.

Наиболее важным фактором, определившим всю концепцию оптимизации, стала проблема “выравнивания” товарных потоков во времени.

Моделирование потоков и построение схем процессов «ТО-ВЕ»

Исходя из описанных факторов, реорганизация складского хозяйства ориентирована в нескольких направлениях:

Регламентирование бизнес-процессов. Так как к данному этапу уже доступен список бизнес-процессов и известны среднестатистические нагрузки, приходящиеся на каждый процесс, то путем варьирования допустимых интервалов времени каждого процесса можно получать теоретический график общей загрузки. В итоге численного моделирования были найдены оптимальные допустимые интервалы времени для каждого процесса, при которых минимизировался “пик” графика общей загрузки.

Изменение процедур поставки и отгрузки. Проведено выделение зон разгрузки, приемки, основного хранения, комплектации заказа и отгрузки. Проведены изменения в информационной системе (склад - офис), а для сбытовых подразделений введена обязательная функция – указание планируемой даты и времени отгрузки заказа клиента. Теперь персонал склада, получив информацию о планируемых расходах, комплектует заказ в зоне комплектации задолго до приезда клиента (во время низкой загруженности), исключая время на поиск товара во время самой отгрузки.

Осуществлено информирование отдела снабжения о ежедневно планируемых сроках и объемах закупок и корректирование данной информации. Персонал склада, получив информацию о планируемых поставках, готовит складские помещения под приемку груза, исключая встречные потоки на одном складе.

Реализация изменений

После введения вышеуказанных изменений в процессах, удалось сократить среднее время обслуживания клиентов до 17 минут при практически полном отсутствии очередей даже в “час пик”, причем такого результата удалось достичь без привлечения дополнительной техники и персонала. График загрузки складского комплекса после изменений отображен на рисунке 66.

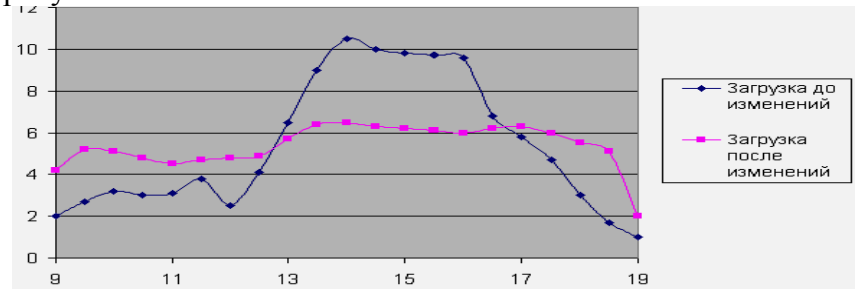


Рисунок 66 – Сравнение загрузки до и после изменений

Таким образом, проблема связана с двумя разными, но взаимосвязанными процессами: организация товародвижения и распределение товаров; изучение спроса на базе распределительного и информационного подходов к маркетингу может быть решена при внедрении и использовании эффективных вариантов размещения, хранения и транспортировки товаров с учетом требований конкретных рынков.

Модель воздействует на все основные области деятельности компании как инструмент маркетинга, обеспечивающая рыночный успех, а отсюда возникает ее маркетинговая функция, которая причинно связана со всеми сферами деятельности компании (прогнозирование рынка, планирование производства, организация сбыта и т. д.). Рыночное значение модели обусловлено также принятием многих распределительных функций торговли на производителя и возрастающей конкуренцией в области сервиса (расходы составляют 10–30 %). Все эти факторы показывают необходимость включения логистики в общую концепцию маркетинга.

ПРОЦЕССНЫЕ ПОТОКОВЫЕ МОДЕЛИ

ПОНЯТИЙНАЯ СТРУКТУРА ПРОЦЕССНЫХ ПОТОКОВЫХ МОДЕЛЕЙ

На нижнем уровне выделенные ("ключевые") процессы могут быть описаны как технологическая последовательность операций (для получения требуемых результатов). ***Для этого применяются потоковые модели бизнес-процессов, назначение которых - описание горизонтальных отношений в организации, связывающих между собой описанные ранее объекты посредством информационных потоков.***

ОПРЕДЕЛЕНИЯ

Процессная потоковая модель организации приведенf на рисунке 67. Такое описание дает представление о процессе последовательного преобразования ресурсов в продукты усилиями различных исполнителей на основании соответствующих регламентов.

Рисунок 67 - Потоковая процессная модель

Разработка требований к проектируемой ИС строится на основе статического и динамичного описания компании. Статическое описание компании проводится на

уровне функциональных моделей и включает описание бизнес-потенциала, функционала и соответствующих матриц ответственности.

Дальнейшее развитие (детализация) бизнес-модели происходит на этапе динамичного описания компании на уровне ***процессных потоковых моделей***.

Процессные потоковые модели — это модели, описывающие процесс последовательного во времени преобразования материальных и информационных потоков компании в ходе реализации какой-либо бизнес-функции или функции менеджмента.

На верхнем уровне описывается логика взаимодействия участников процесса, на нижнем - технология работы отдельных специалистов на своих рабочих местах.

Процессные потоковые модели отвечают на вопросы **кто—что—как—кому**.

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***.

Главными ***недостатками функционального подхода*** являются: ***разбиение*** технологий выполнения работы на отдельные фрагменты, выполняемые различными структурными подразделениями; ***отсутствие*** целостного описания технологий выполнения работы; ***сложность*** увязывания задач в технологию, производящую реальный товар или услугу; ***отсутствие*** ответственности за конечный результат; ***высокие*** затраты на согласование, налаживание взаимодействия, контроль; отсутствие ориентации на клиента.

Процессный подход предполагает смещение акцентов от управления отдельными структурными элементами на управление сквозными бизнес-процессами, связывающими деятельность всех структурных элементов. Каждый деловой процесс проходит через ряд подразделений, в его выполнении участвуют специалисты различных отделов компании. ***Процессный подход*** позволяет устранить фрагментарность в работе, организационные и информационные разрывы, дублирование, нерациональное использование финансовых, материальных и кадровых ресурсов.

Процессный подход к организации деятельности предприятия ***предполагает***: широкое ***делегирование полномочий и ответственности*** исполнителям; ***сокращение уровней принятия решений***; ***сочетание принципа целевого управления*** с групповой организацией труда; ***повышенное внимание к вопросам обеспечения качества***; ***автоматизация технологий*** выполнения бизнес-процессов.

Согласно стандарту “Основные Положения и Словарь — ИСО/ОПМС 9000:2000” понятие “***Процессный подход”*** определяется как “***любая деятельность или комплекс деятельности, в которой используются ресурсы для преобразования входов в выходы, может рассматриваться как процесс. Чтобы результативно функционировать, организации должны определять и управлять многочисленными взаимосвязанными и взаимодействующими процессами. Часто выход одного процесса образует непосредственно вход следующего. Систематическая идентификация и менеджмент приме-***

няемых организацией процессов, и особенно взаимодействия таких процессов, могут считаться «процессным подходом”.

Основной принцип процессного подхода определяет структурирование ***бизнес–системы*** в соответствии с ***бизнес-процессами*** предприятия. Именно ***бизнес-процессы***, обеспечивающие значимый для потребителя результат, представляют ценность для специалистов, проектирующих ИС.

Процессная модель компании должна строиться с учетом следующих положений:

- верхний уровень модели должен отражать ***контекст диаграммы*** – взаимодействие моделируемого контекстным процессом предприятия с внешним миром;
- на втором уровне должны быть отражены тематически сгруппированные ***бизнес-процессы*** предприятия и их взаимосвязи;
- каждая из деятельностей должна быть детализирована на ***бизнес-процессы***;
- детализация ***бизнес-процессов*** осуществляется посредством бизнес – функций;
- описание бизнес–операций осуществляется с помощью мини-спецификаций.

Процессный подход требует комплексного изучения различных сторон жизни организации — правовых основ и правил деятельности, организационной структуры, функций и показателей результатов их исполнения, интерфейсов, ресурсного обеспечения, организационной культуры. В результате анализа ***создается модель деятельности*** “***как есть”.*** Обработка этой модели с помощью различных аналитических методов позволяет проверить, насколько деловые процессы рациональны, а также определить, является ли та или иная операция ориентированной на общественно значимый конечный результат.

В ходе анализа деловых процессов детально исследуются сферы ответственности подразделений ведомства, его руководителей и сотрудников. Это позволяет установить адреса ***владельцев*** деловых процессов, в результате чего процессы перестают быть бесхозными, создаются условия для разработки и внедрения систем стимулирования и ответственности за конечные результаты, определяются моменты и процедуры передачи ответственности. 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.

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. Поэтому ищется компромисс на основе процессно-матричных структур.

Кроме того, новая парадигма деятельности предприятия вызывает появление большого числа процессов управления, распределенных по всему предприятию, а не сосредоточенных в специализированных организационных единицах: это системы качества, бюджетирования, маркетинга и т.п. Поэтому постановка бюджетирования как организационной, а не только финансовой задачи предполагает делегирование полномочий. На более низкие уровни делегируется ответственность за принятие финансовых решений: о заключении сделки-договора, об оплате, о закупке, о скидках и отпуске в кредит и т.п. Это позволяет упростить связи между подразделениями и снизить количество уровней вертикального прохождения документов, что является необходимым условием реализации классической схемы реинжиниринга.

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.

ОСНОВНЫЕ ЭЛЕМЕНТЫ ПРОЦЕССНОГО ПОДХОДА

В рамках ***процессного подхода*** любое предприятие рассматривается как бизнес-система – система, которая представляет собой связанное множество бизнес-процессов, конечными целями которых является выпуск продукции или услуг.

Каждый бизнес-процесс имеет свои границы и роли.

В *процессном подходе* используются следующие ***ключевые роли***: ***владелец процесса*** – человек, отвечающий за ход и результаты процесса в целом; ***лидер команды*** — работник, обладающий знаниями о бизнес-процессе и имеющий позитивные личные качества; ***коммуникатор*** – работник, обучающий команду различным методам работы; ***координатор процесса*** – работник, отвечающий за согласованную работу всех частей бизнеса и обеспечивающий связь с другими бизнес-процессами; ***участники команды*** – специалисты различных уровней иерархии.

Достижение определенной совокупности целей за счет выполнения бизнес-процессов называется ***деревом целей***. ***Дерево целей*** имеет иерархический вид. Каждая цель имеет свой вес и критерий (количественный или качественный) достижимости.

Бизнес-процессы реализуют бизнес-функции предприятия. Под бизнес-функцией понимают вид деятельности предприятия (рисунок 68). Множество бизнес-функций представляет иерархическую декомпозицию функциональной деятельности (***дерево функций).***

Рисунок 68 – Бизнес-функции

Бизнес-функции связаны с показателями деятельности предприятия, образующими ***дерево показателей***. На основании показателей 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.

ВЫДЕЛЕНИЕ И КЛАССИФИКАЦИЯ БИЗНЕС-ПРОЦЕССОВ

При процессном описании должны решаться, как минимум, ***две задачи***:

- ***идентификация всей системы*** "функциональных областей" и процессов компании и их взаимосвязей;

- ***выделение "ключевых" интегрированных процессов*** и их описание на потоковом уровне.

Каждая деятельность компании реализуется как процесс, который имеет своего потребителя: внешнего — клиента или внутреннего — сотрудников или подразделения компании, реализующих другие процессы. На этом этапе выбираются ***ключевые процессы*** для потокового описания, которое необходимо, например, для создания информационной системы предприятия.

Наиболее распространены следующие ***четыре вида бизнес-процессов***:

- ***процессы, создающие наибольшую добавленную стоимость*** (экономическую стоимость, которая определяется издержками компании, относимыми на продукцию);

- ***процессы, создающие наибольшую ценность для клиентов*** (маркетинговую стоимость за счет дифференциации продукции);

- ***процессы с наиболее интенсивным межзвенным взаимодействием***, создающие транзакционные издержки;

- ***процессы, определенные стандартами ИСО 9000, как обязательные*** к описанию при постановке системы менеджмента качества.

Важнейшим шагом при структуризации любой компании является выделение и классификация бизнес-процессов. Выделяют ***следующие классы процессов***: ***основные; управления; обеспечения; сопутствующие; вспомогательные; развития.***

ОСНОВНЫЕ БИЗНЕС-ПРОЦЕССЫ

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

Основные процессы образуют "жизненный цикл" продукции компании. Критериями эффективности таких процессов являются обычно качество, точность и своевременность выполнения каждого заказа. Все они описываются по производственно-коммерческим цепочкам: "***первичное взаимодействие с клиентом и определение его потребностей → реализация запроса (заявки, заказа, контракта и т.п.) → послепродажное сопровождение и мониторинг удовлетворения потребностей".***

Основные бизнес-процессы:

- ***разработка*** (проектирование) продукции;
- ***закупка*** (товаров, материалов, комплектующих изделий);
- ***транспортировка*** (закупленного);
- ***разгрузка, приемка на склад и хранение*** (закупленного);
- ***производство*** (со своим технологическим циклом и внутренней логистикой);
- ***приемка на склад и хранение*** (готовой продукции);
- ***отгрузка*** (консервация и упаковка, погрузка, доставка);
- ***пуско-наладка***;
- ***оказание услуг*** (предусмотренных контрактом на поставку) и т.п.

ПРОЦЕССЫ УПРАВЛЕНИЯ

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 управленческих решений:

- ***стратегическое*** управление;
- ***организационное проектирование*** (структуризация);
- ***маркетинг***;
- ***финансово-экономическое*** управление;
- ***логистика*** и организация процессов;
- ***менеджмент*** качества.

Другая возможная ***систематизация функций управления связана с понятием управленческого цикла*** и базируется на пяти исходных функциях управления: планирование, организация, распорядительство, координация, контроль.

Любая управленческая деятельность развертывается по так называемому "***управленческому циклу"*** (рисунок 69), который включает ***сбор*** информации; ***выработку*** решения; ***реализацию***; ***учет***; ***контроль***; ***анализ***; ***регулирование***.

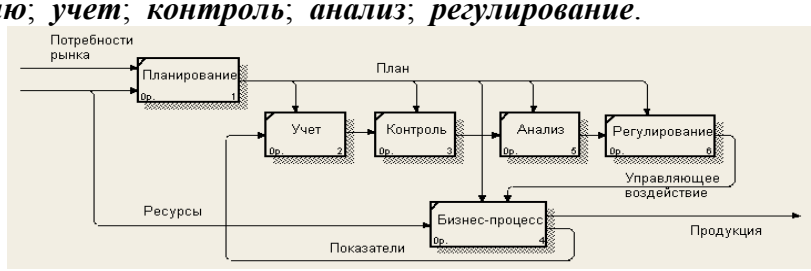


Рисунок 69 – Управленческий цикл

Сбор информации: определение состава собираемой информации, определение форм отчетности; ***выработка решения***: анализ альтернатив, подготовка вариантов решения, принятие решения; ***реализация***: выработка критериев оценки, реализация; ***учет***: планирование, организация, мотивация, координация; ***контроль***: учет результатов, сравнение по принятым критериям; ***анализ***: анализ дополнительной информации, диагностика возможных причин отклонений; ***регулирование:*** регулирование на уровне реализации, регулирование на уровне выработки решения.

Согласно некоторым подходам, в ***процессах управления*** выделяются два типа процессов, относящихся, соответственно, к двум типам менеджмента, условно обозначаемым как "***менеджмент ресурсов"*** и "***менеджмент организации"***, которые отличаются по объекту управления, базовым моделям и, что важно для описания процессов, — своими управленческими циклами.

В основе ***цикла управления ресурсами*** лежит расчет или имитационное моделирование и контроль результатов:

- ***выбор*** (или получение от системы верхнего уровня) целевого критерия оценки качества решения;
- ***сбор информации*** о ресурсах предприятия или возможностях внешней среды;
- ***просчет вариантов*** (с предположениями о возможных значениях параметров);
- ***выбор*** оптимального варианта — принятие решения (ресурсного плана);
- ***учет*** результатов (и отчетность);
- ***сравнение*** с принятым критерием оценки (контроль результатов);
- ***анализ*** причин отклонений и регулирование.

В основе ***цикла организационного менеджмента*** лежит структурное или процессное моделирование и процедурный контроль:

- ***определение состава задач*** (обособленных функций, операций);
- ***выбор*** исполнителей (распределение зон и степени ответственности);
- ***проектирование*** процедур (последовательности и порядка исполнения);
- ***согласование*** и утверждение регламента исполнения (процесса, плана мероприятий);
- ***отчетность*** об исполнении;
- ***контроль*** исполнения (процедурный контроль);
- ***анализ*** причин отклонений и регулирование.

Таким образом, на определенных шагах декомпозиции предприятию надо определить, какие стадии управленческого цикла реализуются по каждой из ранее выделенных задач управления.

ПРОЦЕССЫ ОБЕСПЕЧЕНИЯ

Процессы обеспечения – это процессы, предназначенные для жизнеобеспечения основных 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.

На верхнем уровне детализации можно выделить некоторые стандартные ***процессы обеспечения***: обеспечение ***производства***; ***техобслуживание*** и ремонт оборудования; ***технологическое*** обеспечение; ***экологический*** контроль; ***информационное*** обеспечение; обеспечение ***безопасности***; ***материально-техническое*** обеспечение управления; ***транспортное*** обслуживание и т.п.

ПРОЕКТИРОВАНИЕ БИЗНЕС – ПРОЦЕССОВ

Одной из важных и в то же время сложных задач современного менеджмента является проектирование оптимальных бизнес-процессов и организационной структуры, которые способны реализовать стратегические цели организации.

Совершенствование бизнес процессов и построение на их основе эффективной организации начинается с планирования целей, структуризации и описания организации.

Цель проекта - желаемый результат деятельности, достигаемый в пределах установленного интервала времени.

Сформулированные цели должны соответствовать принципу SMART, согласно которому они должны быть: ясными и точными (S - Specific); измеримыми (M - Measurable); достижимыми (A - Achievable); непротиворечивыми между собой и с целями организации (R - Related); определены по срокам их достижения (T - Times-bound).

ПОДХОДЫ К ОПИСАНИЮ БИЗНЕС-ПРОЦЕССОВ

ЦЕЛИ

Процессы данного уровня формулируют проблемную ситуацию, которую решает организация, и ставят глобальную цель для компании, решающую данную проблемную ситуацию (в долгосрочной и краткосрочной перспективе). То есть следует ответ на вопрос "зачем" создается организация и "что" она должна делать. В результате формируется: миссия, видение и философия организации, направление развития бизнеса, ключевые показатели, релевантные для оценки достижения глобальной цели (рост, прибыльность, эффективность и пр.).

Пример. Руководитель определяет, что его предприятие должно удовлетворять потребности людей товарами такого-то назначения по доступным ценам, высокого качества и предоставить лучший сервис, при этом стремиться стать лидирующей компанией на данном рынке, обеспечивая высокую эффективность деятельности.

Результат формализуется в виде учредительных документов, регламентирующих систему управления.

ОПИСАНИЕ ОКРУЖЕНИЯ БИЗНЕС-ПРОЦЕССА

Первым шагом описания бизнес-процесса является описание его окружения, которое представляет совокупность входов и выходов бизнес-процесса с указанием поставщиков и клиентов. Поставщики и клиенты процесса могут быть как внутренними, так и внешними. Внутренними поставщиками и клиентами являются подразделения и сотрудники компании, с которыми рассматриваемый бизнес-процесс взаимодействует.

При описании окружения бизнес-процесса рекомендуется построить его графическую схему, приведенную на рисунке 70.

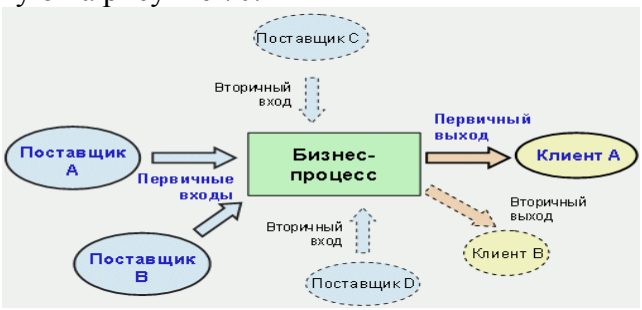


Рисунок 70 - Схема окружения бизнес-процесса

При описании окружения бизнес-процесса приходится его входы и выходы делить на два типа: ***первичные и вторичные***. В результате такого деления получаются первичные и вторичные входы, а также первичные и вторичные выходы. Это делается для того, чтобы не нарушать принцип Парето 20 на 80. Дело в том, что, когда описывается окружение бизнес-процесса, количество различных входов и выходов оказывается очень большим. Описанное окружение также получается чрезвычайно большим и насыщенным. Для того, что отделить существенное от несущественного используется деление входов и выходов бизнес-процесса на первичные и вторичные.

Первичный вход - поток объектов, инициирующий "запуск" бизнес-процесса - заказ клиента, план закупок и т.д. ***Первичный выход*** - основной результат, ради которого существует бизнес-процесс.

Вторичный вход - потоки объектов, обеспечивающие нормальное протекание бизнес-процесса – стандарты, правила, механизмы выполнения действий, оборудование и пр. ***Вторичный выход*** - побочный продукт бизнес-процесса, который может быть востребован вторичными клиентами. Не является основной целью бизнес-процесса.

Процессы окружения координируются результатами процессов первого уровня и связывают текущую деятельность подразделений и исполнителей с формализованными целями организации. ***Таким образом, суть данных процессов составляет:*** SWOT-анализ, то есть выявление угроз, возможностей, слабых и сильных сторон, и стратегический анализ и выработка стратегических альтернатив деятельности компании; выбор способов достижения поставленной глобальной цели для формирования конкурентоспособного предложения на рынке; определение точек контроля, нормативных значений параметров, характеризующих качество выполнения последующих процессов текущей операционной деятельности, исходя из сопоставления целей организации и анализа окружающей среды (клиенты, поставщики, государственные службы и пр.). Результат процессов данного уровня также формализуется в виде нормативных документов.

Пример. Цели и показатели интерпретируются на данный уровень. "Занять лидирующее положение" означает рост объема продаж и рост доли рынка. Чтобы обеспечить достижение этой цели становится очевидным, что необходимы функции процесса "Продажа". Также очевидно, что предложение компании должно быть конкурентоспособным, чтобы обеспечить поток новых клиентов, рост постоянных покупателей и рост частоты продаж. При этом эффективность должна определяться рентабельностью, но при оптимальном соотношении между затратами, наценкой, качеством товаров и сервиса.

Определяется оптимальное соотношение между ценами поставщика и наценкой (удовлетворенность клиента по ценам и рентабельность), качеством товара/сырья (удовлетворенность клиентов по товару, сокращение претензионной работы), имеющегося сервиса и дисциплины поставщика (удовлетворенность клиентов по сервису).

ВНУТРЕННЯЯ ОРГАНИЗАЦИЯ

Данный уровень является логическим продолжением предыдущего. После того как определен набор функций и вместе с тем набор требований, которые должны выполнять эти функции, выполняется следующее: выявляются и формируются элементы организации; определяются отношения между элементами, реализующие целенаправленное функционирование организации; выбираются способы реализации связей между элементами; множество образованных связей и отношений между элементами упорядочивается в структуру организации; проектируются процессы текущей операционной деятельности, отчетность, документооборот.

СПОСОБЫ ОПИСАНИЯ БИЗНЕС-ПРОЦЕССОВ

ГОРИЗОНТАЛЬНОЕ И ВЕРТИКАЛЬНОЕ ОПИСАНИЕ

При вертикальном описании показывают только работы и их иерархический порядок в дереве бизнес-процесса. При горизонтальном описании бизнес-процесса также показывается, как эти работы между собой взаимосвязаны, в какой последовательности они выполняются, какие информационные и материальные потоки между ними движутся. В этом случае в модели бизнес-процесса появляются горизонтальные связи между различными работами, которые процесс составляют (рисунок 71).

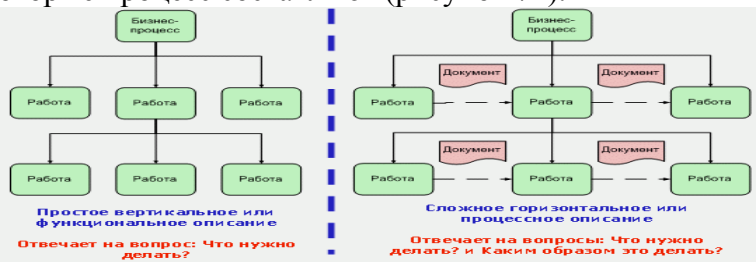


Рисунок 71 - Горизонтальное и вертикальное описание бизнес-процессов

Вертикальное описание бизнес-процессов называют функциональным описанием деятельности, а горизонтальное описание – процессным описанием или просто описанием бизнес-процессов.

При горизонтальном описании бизнес-процессов в настоящее время существуют три основных способа описания (рисунок 72).

Рисунок 72 - Способы описания бизнес-процессов

Первый способ – есть ***текстовое последовательное описание бизнес-процесса***. Для целей анализа и оптимизации деятельности компании данный способ не подходит, так как описание бизнес-процесса в текстовом виде системно рассмотреть и проанализировать невозможно.

Табличная форма описания бизнес-процессов более эффективна по сравнению с текстовой и в настоящее время активно применяется специалистами по информационным технологиям для описания бизнес-процессов в приложении к задачам автоматизации.

Признано, что ***графические методы*** обладают наибольшей эффективностью при решении задач связанных с описанием, анализом и оптимизацией деятельности компании.

ПОСТРОЕНИЕ СЕТИ БИЗНЕС-ПРОЦЕССОВ

Другим наглядным представлением бизнес-процессов компании является сеть процессов, которая представляет DFD-схему, построенную на основе бизнес-процессов, составляющих дерево. Взаимодействия между бизнес-процессами, составляющими дерево, показываются с помощью сети процессов (рисунок 73).

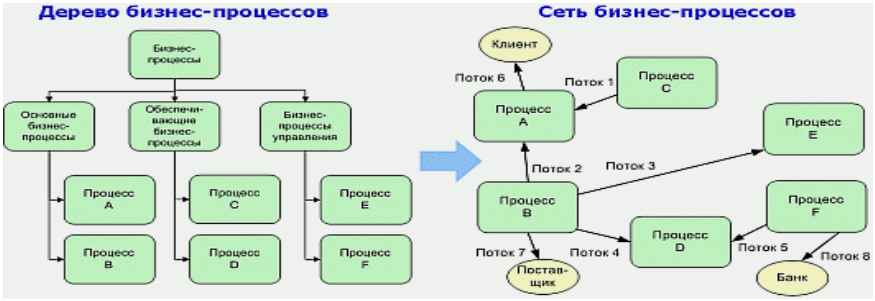


Рисунок 73 - Разработка сети бизнес-процессов

В отличие от дерева бизнес-процессов сеть процесса дает более полное системное представление о деятельности организации, так как позволяет показать не только элементы организации, но и взаимодействия между ними. Помимо этого сеть процессов обеспечивает проверку разработанной модели деятельности организации на целостность, правильность выделения бизнес-процессов и описания их окружения. На практике сеть процессов часто называют сетью или схемой взаимодействия бизнес-процессов. Отличие сети процессов от классической схемы DFD состоит в том, что на сети нужно показать внешние субъекты, с которыми взаимодействуют бизнес-процессы компании – клиенты, поставщики, банки и др. На рисунке 74 приведен пример сети бизнес-процессов для производственной компании.

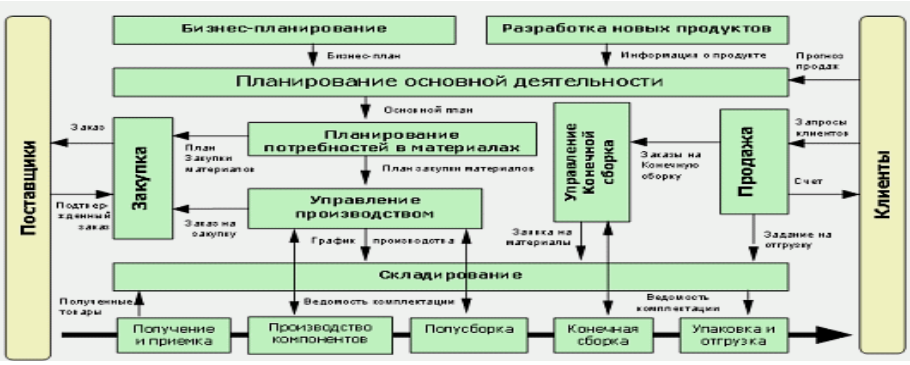


Рисунок 74 - Пример сети бизнес-процессов

ОПИСАНИЕ БИЗНЕС-ПРОЦЕССОВ НИЖНЕГО УРОВНЯ. ДЕКОМПОЗИЦИЯ

ПРИНЦИПЫ ДЕКОМПОЗИЦИИ

Проектирование бизнес-процессов включает их декомпозицию на функции и работы. Строится иерархическая структура работ, которая представляет последовательное многоуровневое расщепление бизнес-процессов высшего уровня на работы.

На рисунке 75 представлен пример иерархии работ.

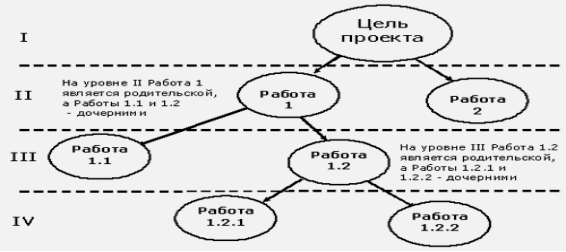


Рисунок 75 - Пример иерархической структуры работ

При построении иерархии работ необходимо соблюдать следующие принципы.

- Работы нижнего уровня являются способом достижения работ верхнего уровня.
- У каждой родительской работы может иметься несколько дочерних работ, достижение которых автоматически обеспечивает достижение родительской работы.
- У каждой дочерней работы может быть только одна родительская работа.
- Декомпозиция родительской работы производится по одному критерию.
- На одном уровне дочерние работы должны быть равнозначны.
- При построении иерархической структуры работ на различных уровнях можно и следует применять различные критерии декомпозиции.
- Последовательность критериев декомпозиции работ следует выбирать таким образом, чтобы большая часть зависимостей и взаимодействий между работами оказалась на самых нижних уровнях иерархии. На верхних уровнях работы должны быть автономны.
- Декомпозиция работ прекращается тогда, когда работы нижнего уровня удовлетворяют следующим условиям: работы ясны и понятны менеджеру и участникам проекта, ответственность за выполнение работ может быть однозначно определена.

Пример. Имеются шары двух цветов - белые и черные, при этом часть шаров сделана из дерева, а часть из железа. Задача: необходимо построить классификатор (дерево) шаров. На рисунке 76 показаны правильные и неправильные способы построения такого классификатора.

Классификатор 1 - неправильный. Одновременная декомпозиция по двум критериям: "Цвет" и "Материал". Блоки нижнего уровня "пересекаются". Любой шар принадлежит одновременно двум блокам нижнего уровня.

Классификатор 2 - правильный. Последовательная декомпозиция по двум критериям: "Цвет" и "Материал". Блоки нижнего уровня не "пересекаются". Любой шар принадлежит только одному блоку нижнего уровня.

Классификатор 3 - правильный. Последовательная декомпозиция по двум критериям: "Материал" и "Цвет". Блоки нижнего уровня не "пересекаются". Любой шар принадлежит только одному блоку нижнего уровня.

Рисунок 76 - Пример неправильной и правильной декомпозиции

Правильный подход к классификации основан на том, что на каждом уровне применяется только один критерий декомпозиции (классификации) - цвет или материал. Количество правильных вариантов больше одного и это означает, что процесс построения иерархии неоднозначен, при планировании проекта можно построить несколько вариантов иерархии работ. ***При разработке проекта в качестве критериев декомпозиции выступают следующие характеристики:*** компоненты результатов и продуктов проекта, этапы жизненного цикла проекта, функциональные виды деятельности и используемые ресурсы, элементы организационной структуры.

Пример. Рассматривается проект строительства гостиничного комплекса (рисунок 77) с вариантами построения иерархии работ: а) продуктовый подход, б) подход по жизненному циклу, в) функциональный подход, г) организационный подход, д) смешанный подход (первый уровень - продуктовый подход, второй уровень - подход по жизненному циклу).

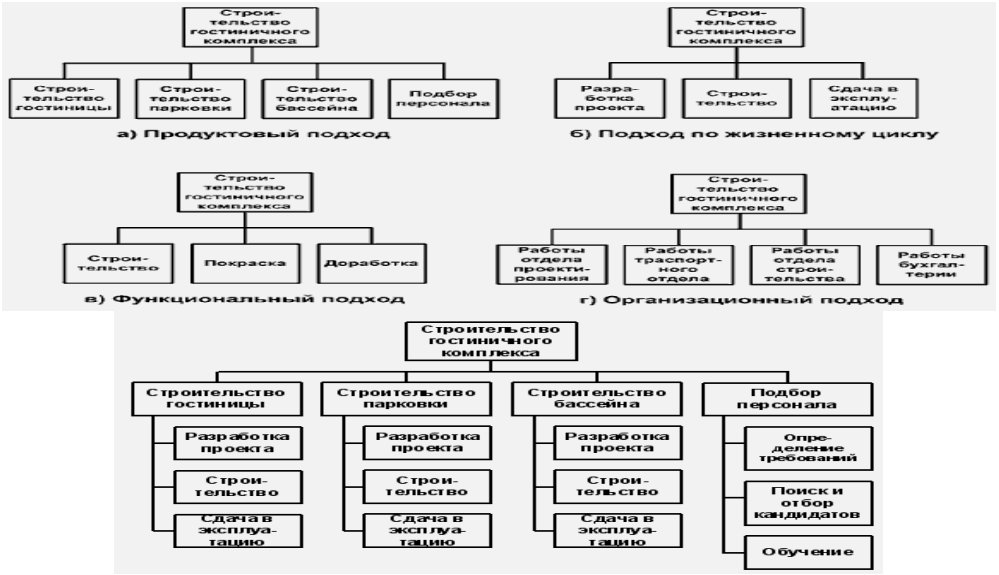


Рисунок 77 - Строительство гостиничного комплекса

Для различных задач управления проектом и различных субъектов управления удобен свой вариант иерархии, и современные программные продукты позволяют строить несколько вариантов иерархии и автоматически трансформировать их друг в друга.

На практике рекомендуется использовать смешанный вариант, при котором на верхнем уровне применяется продуктовый подход, а на втором уровне подход по жизненному циклу.

ПРАВИЛА ДЕКОМПОЗИЦИИ

При построении DFD-схемы бизнес-процесса необходимо использовать правило "7", согласно которому нужно выбрать такой уровень абстрагирования и детализации, при котором схема бизнес-процесса будет состоять в среднем из семи работ. В случае если для достижения целей оптимизации бизнес-процесса требуется большая его детализация, то для этого каждую или некоторые работы процесса рассматривают как подпроцесс и описывают в виде отдельной схемы бизнес-процесса второго уровня (рисунок 78).

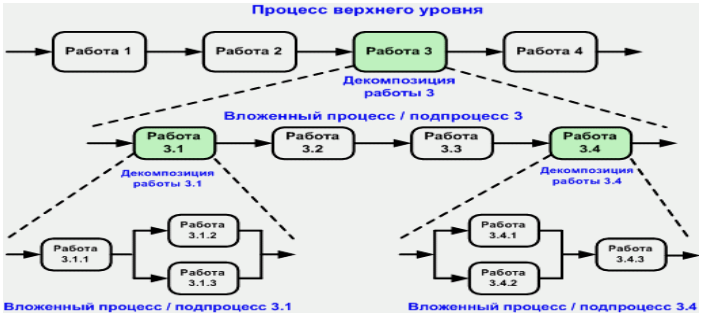


Рисунок 78 - Декомпозиция бизнес-процесса

При классическом подходе описания бизнес-процессов для разработанной схемы второго уровня может использоваться как DFD, так и IDEF3 - формат описания в зависимости от уровня и глобальности работы.

ДИНАМИЧЕСКИЕ ПРОЦЕССНЫЕ МОДЕЛИ

Для создания динамических процессных потоковых моделей используется имитационное моделирование систем. К системам имитационного моделирования относятся системы, созданные на основе сетей Петри (CPN (Color Petri Nets), INCOME Mobile, CPN-AMI), система GPSS, система Arena и другие.

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ с Arena

ОПИСАНИЕ

Перед предприятиями часто встает задача оптимизации технологических процессов. Построение функциональной модели осуществляется от общего к частному - сначала описывается общая схема деятельности предприятия, затем шаг за шагом все более и более подробно описываются конкретные технологические процессы. Такой подход весьма эффективен, однако на уровне наибольшей детализации целесообразно использовать имитационное моделирование.

Имитационное моделирование - это метод, позволяющий строить модели, учитывающие время выполнения функций. Одним из наиболее эффективных инструментов имитационного моделирования является система Arena компании System Modeling. Arena позволяет строить имитационные модели, проигрывать их и анализировать результаты такого проигрывания.

Имитационная модель включает следующие основные элементы: источники и стоки (Create и Dispose), процессы (Process) и очереди (Queue). Источники - это элементы, от которых в модель поступает информация или объекты. Скорость поступления данных или объектов от источника обычно задается статистической функцией. Сток - это устройство для приема информации или объектов. Понятие очереди близко к понятию хранилища данных - это место, где объекты ожидают обработки. Времена обработки объектов (производительность) в разных процессах могут быть разными. В результате перед некоторыми процессами могут накапливаться объекты, ожидающие своей очереди. Часто целью имитационного моделирования является минимизация количества объектов в очередях. Процессы - это аналог работ в функциональной модели. Простейшая имитационная модель, созданная в Arena, показана на рисунке 79.

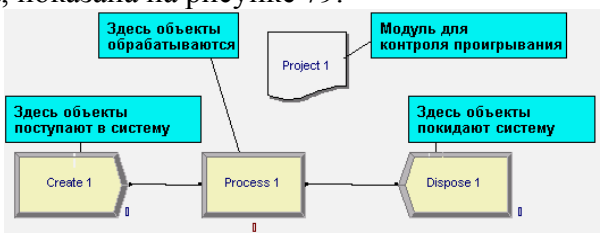


Рисунок 79 - Простейшая имитационная модель

Для построения моделей Arena имеет набор средств, палитру инструментов. Связи между модулями Create, Dispose и Process устанавливаются автоматически (или вручную). Модуль Create является источником сущностей в системе. Так, например, если описывает-

ся изготовление изделий, то модуль Create может описывать поступление заготовок на конвейер. Модуль Process отвечает за обработку сущностей. Например, он может имитировать станок, обрабатывающий заготовки. Модуль Dispose является стоком сущностей из системы. Он может моделировать снятие готовых изделий с конвейера. После проигрывания модели автоматически генерируются отчеты в формате Crystal Reports.

Модули, обрабатывающие сущности, могут иметь различные состояния, например, “ожидание” или “работа”. Каждому состоянию можно поставить в соответствие определенное изображение и, тем самым, анимировать имитационную модель. Один из примеров приведен на рисунке 80.

Модель показывает систему обработки документа (закладной). Сначала документ регистрирует секретарша (иконка слева в нижней части рисунка), затем просматривает клерк (иконка справа). Затем клерк либо принимает документ, либо возвращает. Очередь документов показывается в виде набора иконок сверху от процесса Review Application и в виде графика в правой нижней части рисунка. Иконки, отображающие секретаря и клерка, могут быть разными в зависимости от состояния (занят – ожидает).

Создавать имитационные модели без предварительного анализа бизнес-процессов не всегда представляется возможным. Поэтому функциональные модели и имитационные модели не заменяют, а дополняют друг друга и могут быть тесно взаимосвязаны.

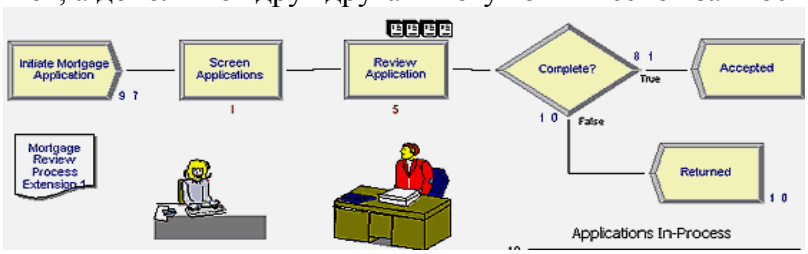


Рисунок 80 - Модель обработки документа

Имитационная модель дает больше информации для анализа системы, результаты такого анализа могут стать причиной модификации модели процессов. Наиболее целесообразно сначала создать функциональную модель, а затем на ее основе строить модель имитационную. Для поддержки такой технологии инструментальное средство функционального моделирования BPwin 4.0 имеет возможность преобразования диаграмм IDEF3 в имитационную модель Arena (версии 3.6 и выше). Для преобразования диаграммы IDEF3 в модель Arena необходимо, чтобы BPwin 4.0 и Arena одновременно были запущены. В BPwin 4.0 следует открыть диаграмму IDEF3 и затем выбрать меню File/Export/Arena. Далее экспорт производится автоматически.

Имитационная модель имеет больше параметров, чем IDEF3, в BPwin 4.0 имеется возможность задать эти параметры с помощью свойств, определяемых пользователем (UDP). В поставку BPwin 4.0 входят примеры моделей с предварительно внесенными UDP для экспорта в Arena (Program Files/Computer Associates/BPwin 4.0/Samples/Arena/) и модель ArenaBEUDPs.bp1, в которой определены все необходимые для экспорта UDP и которую можно использовать в качестве шаблона для создания новых моделей. На рисунке 81 показан пример модели и результат экспорта этой модели в Arena (рисунок 82).

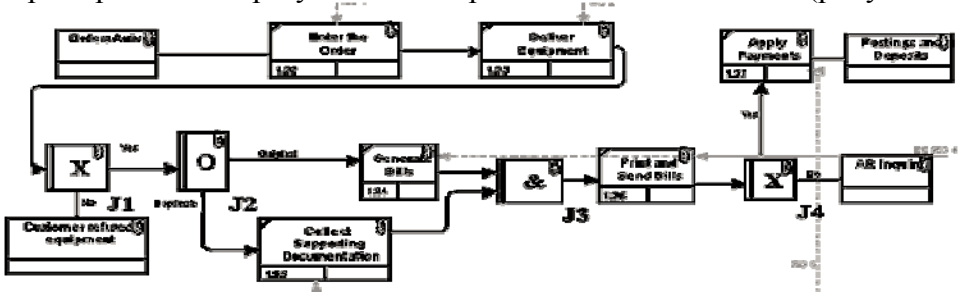


Рисунок 81 - Диаграмма IDEF3 – пример для иллюстрации экспорта в Arena

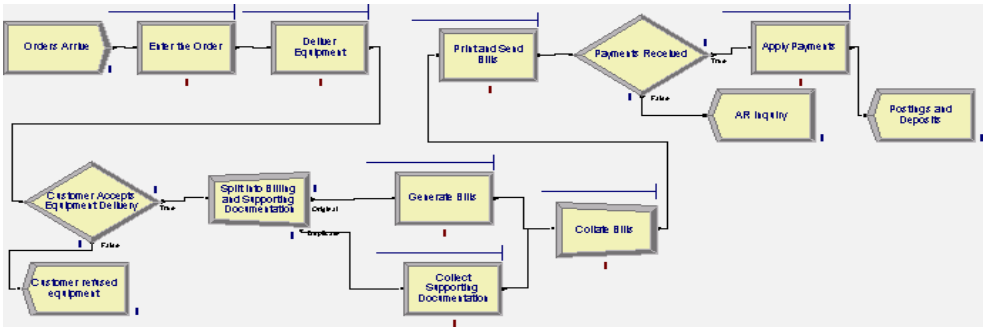


Рисунок 82 - Имитационная модель Arena – результат импорта из BPwin

ПОСТРОЕНИЕ МОДЕЛИ

Рассматривается преобразование модели ***ОБРАБОТКА СЫРЬЯ*** (рисунок 83) из функциональной модели ***ПРОИЗВОДСТВО ИЗДЕЛИЙ*** для экспорта в Arena.

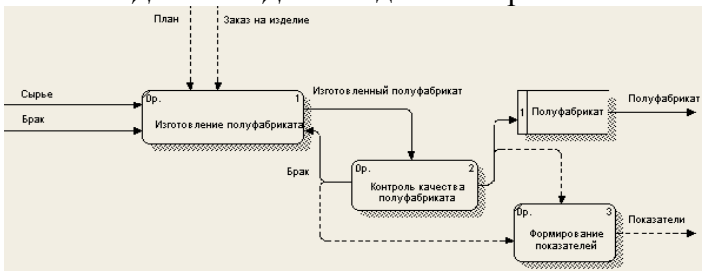


Рисунок 83 – Обработка сырья

При построении имитационной модели учитываются только обрабатываемые при помощи функций потоки данных. Используя методологию IDEF3, построим модель нижнего уровня с объектами – логическими операторами, с помощью которых показываются альтернативы и места принятия решений в бизнес-процессе, а также стрелки временной последовательности работ (рисунок 84).

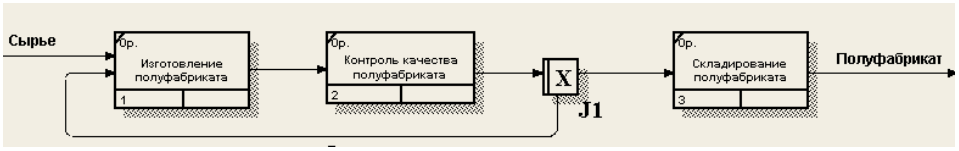


Рисунок 84 - Схема бизнес-процесса *ОБРАБОТКА СЫРЬЯ* в стандарте IDEF3

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ НА GPSS

ПОСТАНОВКА ЗАДАЧИ

Рассматривается построение имитационной модели ***СБОРКА ИЗДЕЛИЯ*** из функциональной модели ***ПРОИЗВОДСТВО ИЗДЕЛИЙ***. Исходная модель имеет вид (рисунок 85).

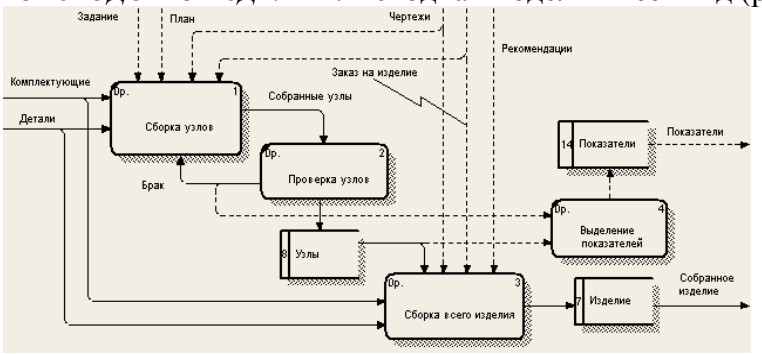


Рисунок 85 – Сборка изделий

Учитываются обрабатываемые при помощи функций потоки данных. Кроме того, потоки *КОМПЛЕКТУЮЩИЕ*, *ДЕТАЛИ* – потоки одного ансамбля заменяются одним потоком *КОМПЛЕКТУЮЩИЕ*, а также добавляются в модель накопители или очереди. Доля брака при проверке узлов составляет в среднем 0.05 (5%). Имитационная модель имеет вид (рисунок 86).

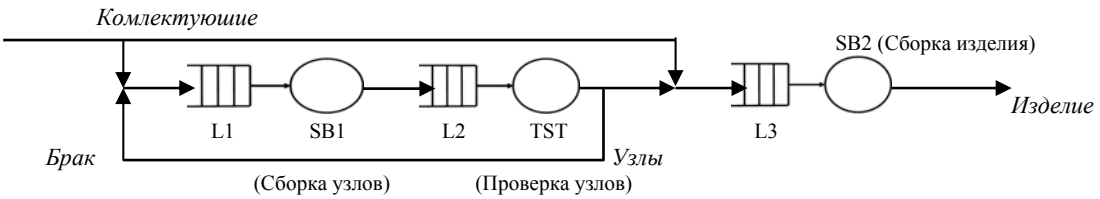


Рисунок 86 – Имитационная модель

ПРОГРАММА МОДЕЛИ

```
SIMULATE
        GENERATE    12
CHAN1   QUEUE       L1
        SEIZE       SB1
        DEPART      L1
        ADVANZE     5
        RELEASE     SB1
        QUEUE       L2
        SEIZE       TST
        DEPART      L2
        ADVANCE     5
        RELEASE     TST
        TRANSFER    .05,,CHAN1
        TRANSFER    ,CHAN2
        GENERATE    8
CHAN2   ASSEMBLE    3
        QUEUE       L3
        SEIZE       SB2
        DEPART      L3
        ADVANCE     5
        RELEASE     SB2
        TERMINATE
        GENERATE    480
        TERMINATE 1
        START 1
        END
```

С помощью блока **GENERATE** генерируется два одинаковых случайных потоков транзактов с равномерным распределением ***Комлектующие*** со средним интервалом временем 2. Для сборки узлов требуется 6 комплектов комплектующих (**GENERATE 12**), для сборки изделия – 4 (**GENERATE 8**). Блок **SB1** - моделирование процесса сборки узлов. Блок **TST** – проверка узлов. Блок **TRANSFER .05,,CHAN1** реализует статистический режим работы; из общего числа транзактов, входящих в блок, в среднем 0.05 по вероятности будут пытаться войти в блок **CHAN1,** остальные 0.95 – в следующий блок (моделирование потока ***Брак)***. Блок **ASSEMBLE 3** объединяет потоки ***Комлектующие*** и ***Узлы*** для сборки ***Изделия*** (**SB2**).

РЕЗУЛЬТАТ

Выходной отчет при заданных параметрах:

```
Facility         Average          Number          Average
                 utilization      entries         time/tran
   SB1               .42             41              4.88
   TST               .42             40              5.00
   SB2               .33             33              4.85



Queue            Maximum      Average      Total        Zero       Percent
(AD set)         contents     contents     entries      entries    zeros

L1                     1          .00          41           40       97.56
L2                     1          .00          40           40      100.00
L3                     1          .00          33           33      100.00



Queue            Average      $Average     Current
(AD set)         time/trans   time/trans   contents

L1                    .07         3.00           0
L2                    .00          .00           0
L3                    .00          .00           0
```

Загрузка приборов: **SB1**– 0.42; **TST** – 0.42; **SB2** – 0.33. Среднее время обслуживания: **SB1** – 4.88; **TST** – 5.00; **SB2** – 4.85. Количество обслуженных транзактов: **SB1** – 41; **TST** – 40; **SB2** – 33. Максимальные очереди: L_1 – 1; L_2 – 1; L_3 – 1.

На основании полученных данных можно сделать вывод о том, что за 480 ед. времени можно собрать 33 изделия.

Рассматривается построение функции реакции системы. В качестве факторов выступают три параметра - время сборки узлов $\mathbf{T_{SB1}}$, время проверка узлов $\mathbf{T_{TST}}$, время сборки

изделия T_{SB2}. Факторное пространство определяется нормативами в пределах (2 - 30) ед. времени. В качестве отклика выбирается количество собранных изделий.

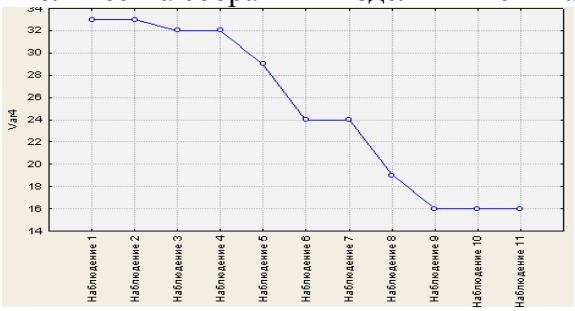
Считается, что модель линейна и, следовательно, включает два уровня факторов, т.е. реализуется план типа 2^K, где k- число факторов.

Факторы: **Var1** - время сборки узлов T_{SB1}, **Var2** - время проверки узлов T_{TST}, **Var3** - время сборки изделия T_{SB2}. Отклик системы **Var4** – количество собранных изделий.

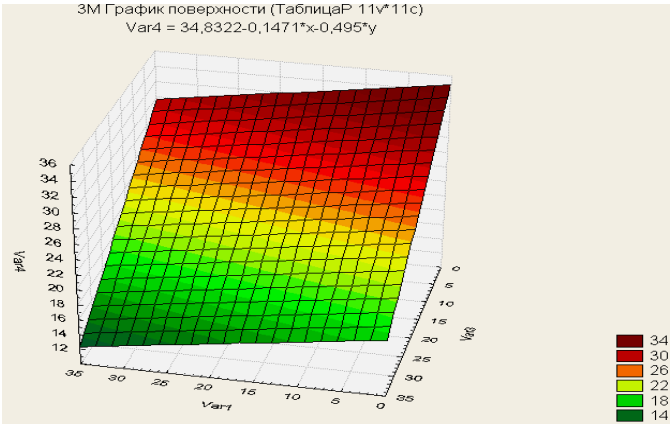
План проведения эксперимента и его результаты:

	1 Var1	2 Var2	3 Var3	4 Var4
1	2	2	2	33
2	6	6	6	33
3	8	8	8	32
4	10	10	10	32
5	15	15	15	29
6	30	7	7	24
7	7	30	7	24
8	25	25	25	19
9	30	7	30	16
10	7	7	30	16
11	30	30	30	16

График изменения количества собранных изделий в испытаниях:



Поверхность отклика **Var4** – количество собранных изделий в зависимости от основных параметров **Var1** - время сборки узлов, **Var3** - время сборки изделия:



Максимальное значение **Var4** достигается при минимальных значениях **Var1** и **Var3**.

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ СЕТЯМИ ПЕТРИ

ПОСТАНОВКА ЗАДАЧИ

Рассматривается построение сети Петри ***ИЗГОТОВЛЕНИЕ ДЕТАЛЕЙ*** из функциональной модели ***ПРОИЗВОДСТВО ИЗДЕЛИЙ***. Исходная модель имеет вид (рисунок 87).

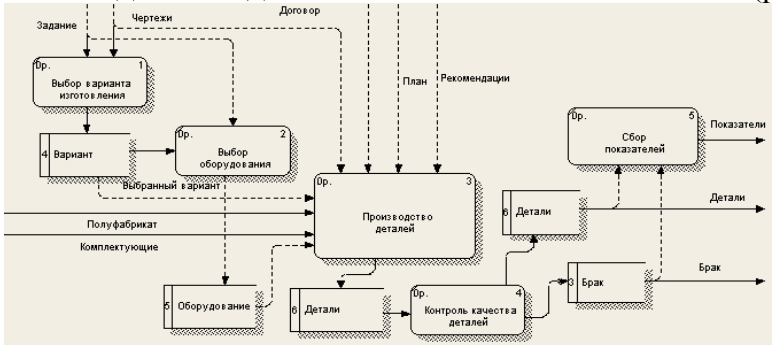


Рисунок 87 - Изготовление деталей

Имитационная модель имеет вид (рисунок 88):

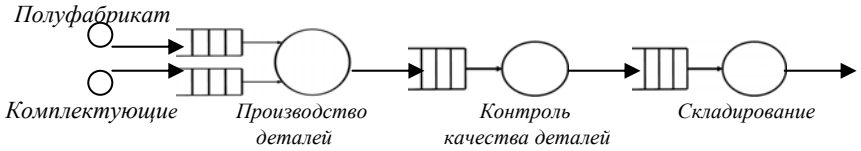
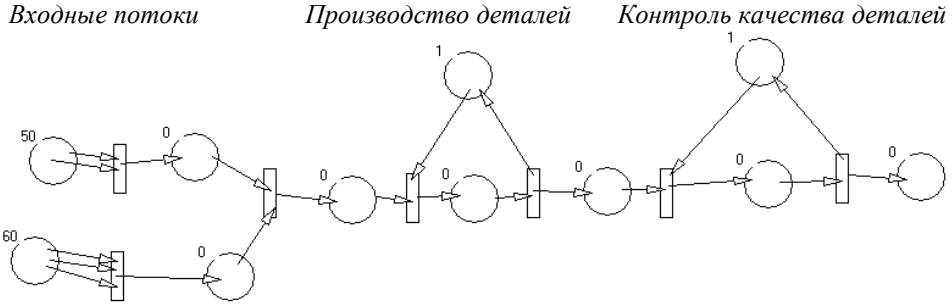


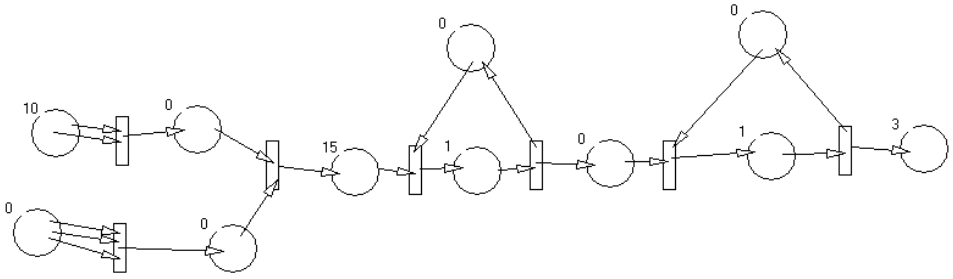
Рисунок 88 – Имитационная модель

Ниже на рисунке 89 показана сеть Петри в различных фазах функционирования. Первый рисунок а) показывает начальное состояние сети: в каждую ед. времени с равномерным распределением поступает одна единица полуфабриката и одна единица комплектующих; количество полуфабриката равно 50 ед., количество комплектующих – 60 ед.; для производства деталей необходимо 2 ед. полуфабриката и 3 ед. комплектующих; далее производится изготовление деталей (задержка времени 5 ед.) и контроль качества (задержка времени 4 ед.).



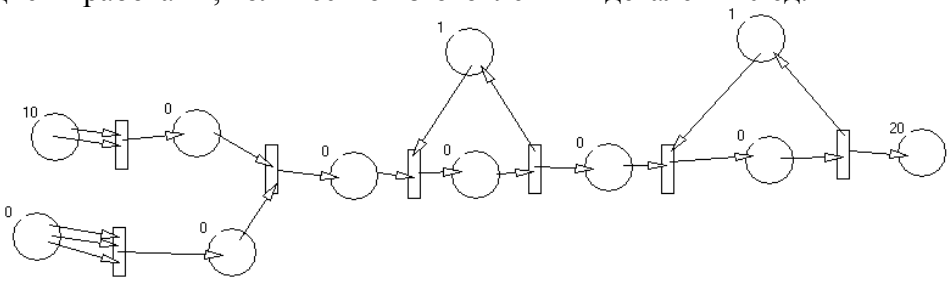
а)

Вторая фаза б) показывает текущий процесс функционирования; максимальная очередь изготовления деталей составляет 15 ед.; максимальная очередь контроля качества деталей составляет 1ед.; количество изготовленных деталей -3 ед.



б)

Заключительная фаза изготовления деталей в): необходимые полуфабрикат и комплектующие выработаны, количество изготовленных деталей -20 ед.



в)

Рисунок 89 – Сеть Петри

СПИСОК ЛИТЕРАТУРЫ

1 Диалоговые системы. Современное состояние и перспективы развития. Под общей редакцией А.М. Довгялло: [Текст]. – Киев. Наукова думка, 1987.
2 Грекул, В.И. Проектирование информационных систем. Интернет-университет информационных технологий [Электронный ресурс]/ В.И. Грекул, Г.Н. Денищенко, Н.Л. Коровкина //- ИНТУИТ.ru.
3 Данилин, А. Архитектура и стратегия. "Инь" и "янь" информационных технологий Интернет-университет информационных технологий [Электронный ресурс]/ А. Данилин, А. Слюсаренко// - ИНТУИТ.ru.
4 Вендров, А.М. Проектирование программного обеспечения экономических информационных систем [Текст]/ А.М. Вендров.- М.: «Финансы и статистика», 2000.
5 Проектирование информационных систем//КомпьютерПресс, 2001, №9.
6 Колтунова, Е. Требования к информационной системе и модели жизненного цикла. Автоматизированные Системы. Стадии создания. ГОСТ 34.601-90. Комплекс стандартов на автоматизированные системы [Текст]/Е. Колтунова.- М.: ИПК издательство стандартов, 1997.
7 ISO/IEC 12207:1995.
8 Буч, Г. Язык UML. Руководство пользователя [Текст]/ Г. Буч, Д. Рамбо, А. Джекобсон.- М.: ДМК, 2000.
9 Смирнова, Г.Н. Проектирование экономических информационных систем [Текст]/ Г.Н. Смирнова, А.А. Сорокин, Ю.Ф. Тельнов. - М.: Финансы и статистика, 2002.
10 Елиферов, В.Г. Бизнес-процессы: регламентация и управление [Текст]/ В.Г. Елиферов, В.В. Репин. - М.: ИНФРА-М, 2004.
11 Калянов, Г.Н. Теория и практика реорганизации бизнес-процессов [Текст]/ Г.Н. Калянов. - М.: СИНТЕГ, 2000.
12 Калянов, Г.Н. Структурный системный анализ [Текст]/ Г.Н. Калянов. - М.: Лори, 1997.
13 Марка, Д.А. SADT — методология структурного анализа и проектирования [Текст]/ Д.А. Марка, К. МакГоуэн. - М.: Метатехнология, 1993.
14 Маклаков, С.В. Создание информационных систем с AllFusion Modelling Suite [Текст]/ С.В. Маклаков.- М.: Диалог-МИФИ, 2003.
15 Черемных, С.В. Структурный анализ систем. IDEF-технологии [Текст]/ С.В. Черемных, В.С. Ручкин, И.О. Семенов. - М.: Финансы и статистика, 2001.
16 ГОСТ 6.01.1-87 Единая система классификации и кодирования технико-экономической информации [Текст].- М.: Изд-во стандартов, 1987.
17 Буч, Г. Объектно-ориентированное проектирование с примерами применения [Текст]/ Г. Буч. - М.: Конкорд, 1992.
18 Нейбург, Э. Д. Проектирование баз данных с помощью UML [Текст]/ Э. Д. Нейбург, Р.А. Максимчук. - М.: Издательский дом «Вильямс», 2002.
19 Робсон, М. Практическое руководство по реинжинирингу бизнес-процессов [Текст]/ М. Робсон, Ф. Уллах. – М.: Аудит, “Юнити”. – 224 с.
20 ГОСТ Р ИСО 9000:2001 Системы менеджмента качества [Текст].– М.: Госстандарт России. – 2001.
21 Щепетова, С.Е. Динамическое моделирование функционирования предприятия и формирование стратегии его поведения в конкурентной среде [Текст]: автореф. дис….канд. экон. Наук/ С.Е. Щепетова. –М.:Финансовая академия при Правительстве РФ, 2001.–24 с.
22 Кокин, А.Г. Построение модели данных [Текст]: методические указания/ А.Г. Кокин. – Курган:Изд-во, КГУ, 2003.
22. Кокин, А.Г. Создание функциональной модели с BPwin [Текст]: методические указания / А.Г. Кокин. – Курган: Изд-во КГУ, 2005.
23 Кокин, А.Г. Создание модели данных с помощью Erwin [Текст]: методические указания/ А.Г. Кокин. – Курган: Изд-во КГУ, 2006.

УЧЕБНОЕ ИЗДАНИЕ

Кокин Александр Георгиевич

ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

УЧЕБНОЕ ПОСОБИЕ

Редактор Н.М. Кокина

Подписано в печать	Формат 60x84 1/8	Бумага тип. №1
Печать трафаретная	Усл. печ. л. 12,5	Уч.-изд. л. 12,5
Заказ	Тираж: электр. вар.	Цена свободная

Редакционно-издательский центр КГУ
640669, г. Курган, ул. Гоголя, 25
Курганский государственный университет.