RAD модель на основе моделирования предметной области.

Функциональное моделирование (IDEF0, DFD) – моделирование данных (IDEF1X) – моделирование поведения (IDEF3)– автоматическая кодогенерация – сборка и квалификационные испытания

Если процесс длится дольше 60 дней, RAD не применяется.

RAD модель также может использоваться для параллельной разработки приложений.

Использование RAD в параллельной разработке ПО.

КАРТИНКА

RAD модель эффективно используется в следующих случаях:

1. При разработке систем и продуктов, для которых характерно хотя бы одно из следующих свойств:
   1. Поддается моделированию
   2. Предназначены для концептуальной проверки
   3. Являются некритическими
   4. Имеют небольшой размер
   5. Имеют низкую производительность
   6. Относятся к известной разработчикам ПрО.
   7. Являются информационными системами
   8. Требования для них хорошо известны
   9. Имеются пригодный к повторному использованию компоненты
2. Если пользователь может принимать постоянное участие в процессе разработки.
3. Если в процессе заняты разработчики, обладающие достаточными навыками в использовании инструментальных средств в разработке.
4. При выполнении проектов в сокращенные сроки (не более 60 дней).
5. При разработке ПО, для которого требуется быстрое наращивание функциональных возможностей на последовательной основе.
6. При невысокой степени технических рисков.
7. В составе других моделей ЖЦ.

Сравнительная оценка трудозатрат по этапам процесса разработки ПО.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер подхода | Анализ  % | Проектирование  % | Кодирование  % | Тестирование  % |
| 1 | 20 | 15 | 20 | 45 |
| 2 | 30 | 30 | 15 | 25 |
| 3 | 40 | 40 | 5 | 15 |

1: Традиционная разработка с использованием классических технологий.

2: Разработка с использованием современных структур методологий проектирования.

3: Разработка с использованием CASE технологий

Базовые принципы построения CASE средств.

Большинство CASE средств основывается на парадигме, методе, нотации средств.

Парадигма – система изменяющихся форм некоторого понятия. Например, метод реализуется с помощью нотации. Метод и нотации поддерживаются инструментальными средствами.

Метод – систематическая процедура или техника генерации описания компонентов ПО.

Например, метод JSP Джексона методологии SADT.

Нотация – система обозначений, предназначенная для описания структуры системы элементов данных, этапов обработки. Может включать графы, диаграммы, таблицы, схемы, алгоритмы, формальные и естественные языки.

Например, современной нотацией методологии SADT является IDEF0.

Средства – инструментарий для поддержки методов, позволяющий пользователям при создании или редактировании графического проекта в интерактивном режиме выполнять проверку соответствующих компонентов.

Например, IDEF0 поддерживается ERWin или Visio.

CASE средства базируются на следующих принципах:

1. Графическая ориентация

Используется мощная графика для описания и документирования программного обеспечения и для улучшения взаимодействия пользователя с интерфейсом.

1. Интеграция

Обеспечивает легкость передачи данных между своими компонентами и другими средствами, входящими в состав CASE средств.

1. Локализация всей проектной информации в репозитории.

В основе концептуального построения CASE средств лежат следующие принципы:

1. Человеческий фактор
2. Использование базовых программных средств, применяющихся в других приложениях
3. Автоматизация или автоматическая кодогенерация

При автоматической кодогенерации выполняется полная генерация кодов программным обеспечением.

При автоматизированной кодогенерации выполняется частичная генерация кодов, остальные участки кодов выполняются вручную.

1. Ограничение сложности

Такое ограничение позволяет поддерживать сложность компонентов разрабатываемого ПО на уровне, допустимого для понимания, использования и модификации.

1. Доступность для различных категорий пользователей.

В том числе заказчиков, специалистов ПрО, системных аналитиков и т. д.

1. Рентабельность

Обеспечивает быструю окупаемость денежных средств.

1. Сопровождаемость

CASE средства обладают способностью адаптации к изменяющимся требованиям и целям проекта.

Основные функциональные возможности CASE средств

В состав CASE средств входят 4 основных компонента:

1. Средства централизованного хранения всей информации о проекте (репозиторий)
2. Средства ввода

Служат для ввода данных в репозиторий, должны поддерживать методологии анализа, проектирования, тестирования и контроля. Предназначены для обеспечения ввода на протяжении всего ЖЦ ПО.

1. Средства анализа и разработки

Предназначены для анализа всех видов графических и текстовых описаний, их преобразования в процессе разработки.

1. Средства вывода

Служат для кодогенерации, создания различных видов документов, управления проектом.

Все компоненты обладают следующими функциями:

1. Поддержка графических моделей
2. Контроль ошибок
3. Поддержка репозитория
4. Поддержка основных, вспомогательных и организационных процессов ЖЦ ПО.

Поддержка графических моделей.

В CASE средствах разрабатываемое ПО представляется схематически. На разных уровнях проектирования могут использоваться различные виды и нотации графического представления. Обычно применяются диаграммы различных видов. На стадии диаграммы функционального моделирования применяется IDEF0 и DFD. На стадии диаграммы информационного моделирования применяется IDEF1X, JSP, Варнье – Орра, UML. Разработка программ осуществляется с помощью специальных графических редакторов, основная функция которых – создание и редактирование иерархически связанных диаграмм, объектов и связей между объектами, а также автоматический контроль.

Основной принцип методологии Варнье-Орра – зависимость структуры проектируемой программы от структур данных. Структура данных и структура программы могут быть представлены единым набором основных конструкций. В данной методологии структура программы зависит от структуры выходных данных. Декомпозиция структуры данных выполняется слева направо и использует 4 базовые конструкции:

1. Иерархия данных

Отражает вложенность данных.

1. Конструкция последовательности данных

Эта конструкция возникает, когда 2 или более компонента данных помещаются вместе строго последовательным образом и образует единый компонент данных.

1. Конструкция выбора данных

Конструкция позволяет выбрать компонент данных из родительского компонента к одному из двух или более выбираемых подкомпонентов.

1. Конструкция повторения данных

Когда компонент может повторяться.

1. Конструкции параллелизма

Используется, когда подкомпоненты некоторого компонента могут выполняться в любом порядке. В том числе и параллельно.

1. Конструкция рекурсии

Используется, если в состав некоторого компонента в качестве подкомпонента входит сам компонент.

Контроль ошибок.

В CASE средствах реализуется следующие тип контроля:

1. Контроль синтаксиса диаграмм и типа их элементов
2. Контроль полноты и корректности диаграмм
3. Контроль декомпозиции функций

Оценивается на основе различных метрик

1. Сквозной контроль

Диаграмм одного или различных типов

Поддержка репозитория.

В репозитории хранится диаграмма, определение экранов и меню, проектов отчетов, описание данных, моделей обработки, исходные коды, элементы данных.

Основные типы отчетов, выгружаемых из репозитория:

1. Отчеты по содержимому

Сюда входит информация по потокам данных и их компонентов, списки функциональных блоков диаграмм и их входных и выходных потоков, списки всех информационных объектов и их атрибутов, история изменения объектов, описание модулей и интерфейсов между ними, план тестирования модулей

1. Отчеты по перекрестным ссылкам
   1. Содержит информацию по связям всех вызывающих и вызываемых модулей.
   2. Список объектов репозитория, в который имеет доступ конкретный исполнитель проекта.
   3. Информация по связям между диаграммами и конкретными данными.
   4. Маршруты движения данных от входа к выходу
2. Отчеты по результатам анализа
3. Отчет по декомпозиции объектов включает совокупность объектов, входящих в каждый объект

Поддержка процессов ЖЦ ПО.

1. Покрытие всего ЖЦ системы или ПО.
2. Поддержка прототипирования.
3. Поддержка современных методологий разработки систем и программных средств.
4. Автоматическая кодогенерация.

Средства кодогенерации можно разделить на два вида:

1. Средства генерации управляющей структуры продукта

Данное средство выполняет автоматическое построение логической структуры ПО, кодов для баз данных, файлов, экранов и отчетов. Остальные фрагменты ПО кодируются вручную.

1. Средства генерации полного продукта.

Данные средства позволяют на основе разрабатываемых спецификаций или моделей генерировать полный код ПО, а также пользовательскую и программную документацию к нему.

Построение нотации DFD на примере автоматизации продаж

1. Клиент предоставляет свои данные и заявку.
2. Менеджер проверяет и вносит полученные данные в систему.
3. Работник склада формирует документы.
4. Клиент получает заказ.

Эти шаги мы рассматриваем с точки зрения хранения данных и работы с ними в IT системе.

С точки зрения DFD модели у нас есть:

1. Покупатель – внешняя сущность

Он является источником данных и получателем результата.

1. Процесс обработки заказа

Здесь происходит подтверждение и проверка данных в системе менеджера.

1. Сбор заказа на складе после получения заказа.
2. Оформление отгрузки и создание необходимых документов.

Правила, которые необходимо соблюдать:

1. Каждый процесс должен иметь хотя бы один вход и выход
2. Процесс обработки должен иметь внешнюю входящую стрелку - данные от внешней сущности.
3. Стрелки не могут связывать напрямую хранилища данных. Все связи идут через процессы.
4. Все процессы должны быть связаны либо с другими процессами, либо с другими хранилищами данных.
5. Декомпозиция

Документы по заказу

Заявка клиента

Система обработки заказа

Покупатель

Оформление заказа

Инф о заказе

Инф о клиенте

(БД) Остаток товара

Собрать заказ клиента

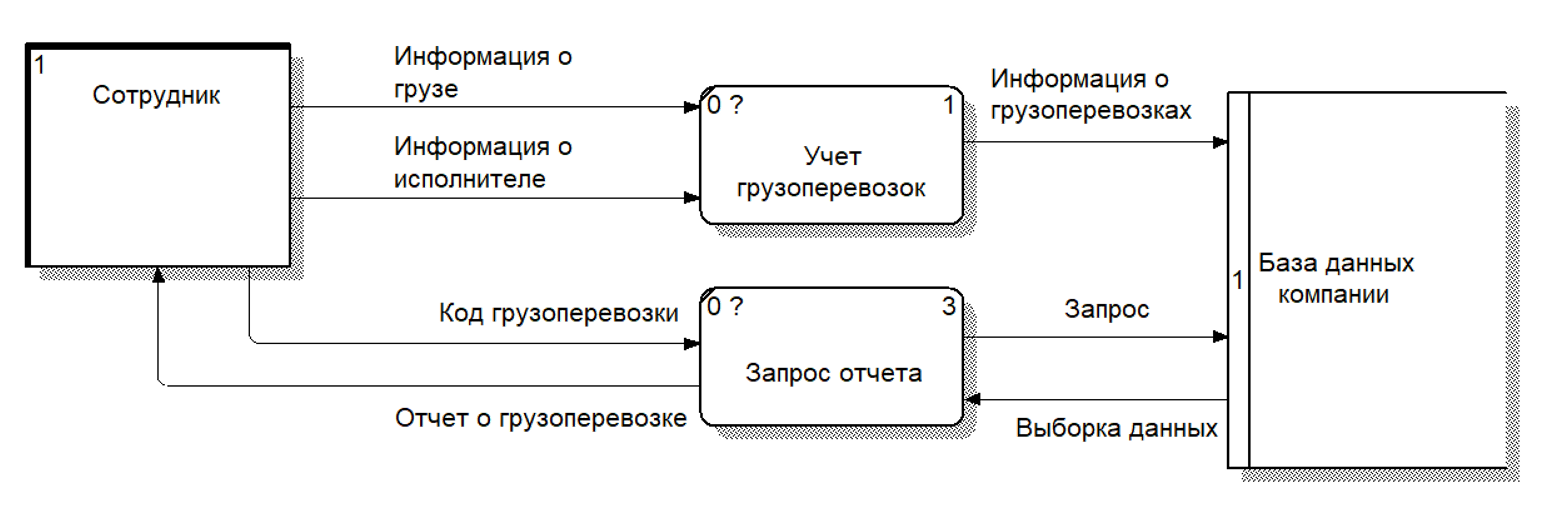
(БД) Заказы

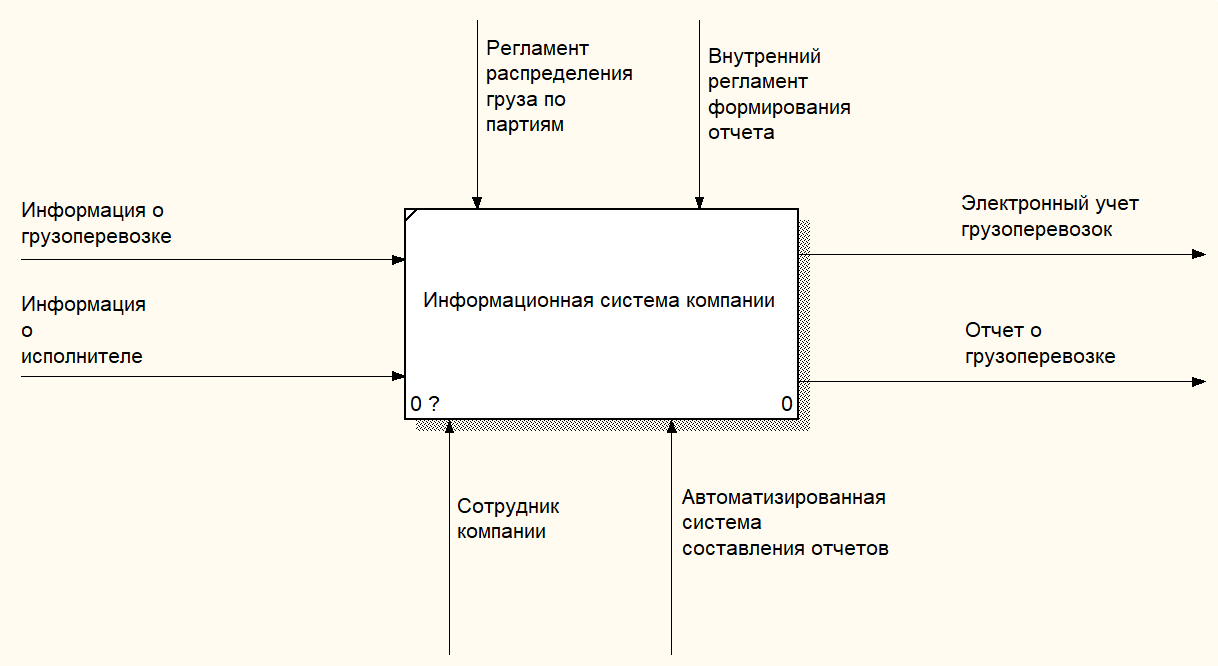
клиентов

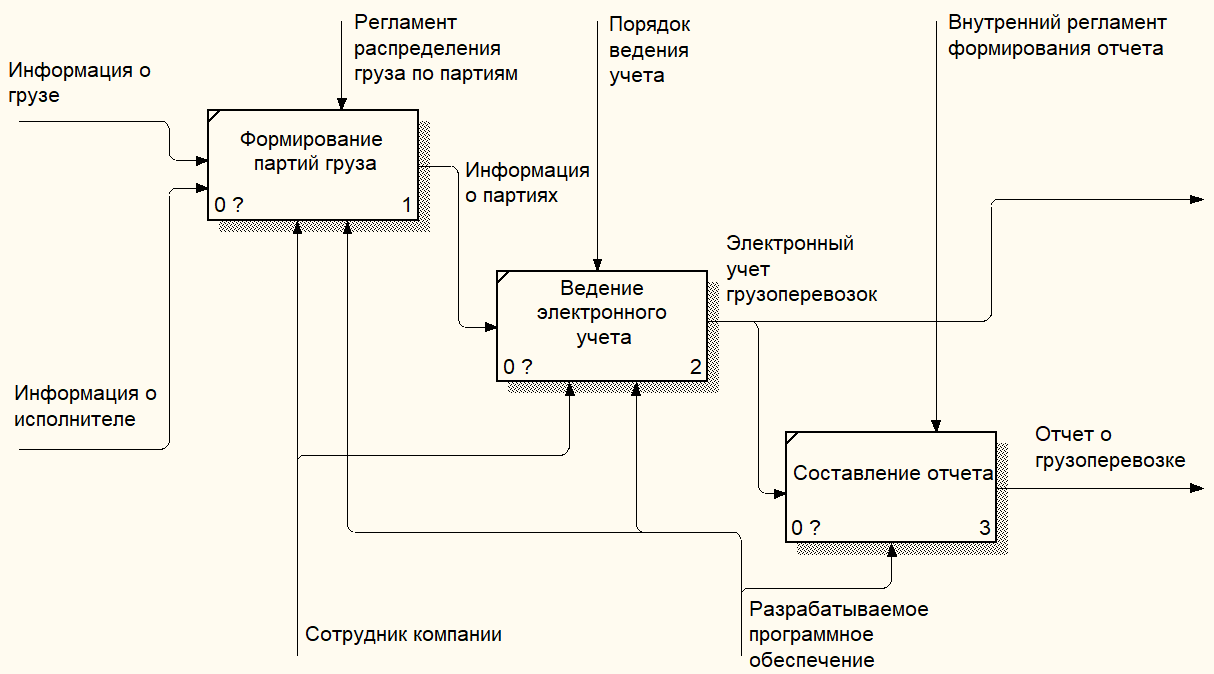
(БД) Клиенты

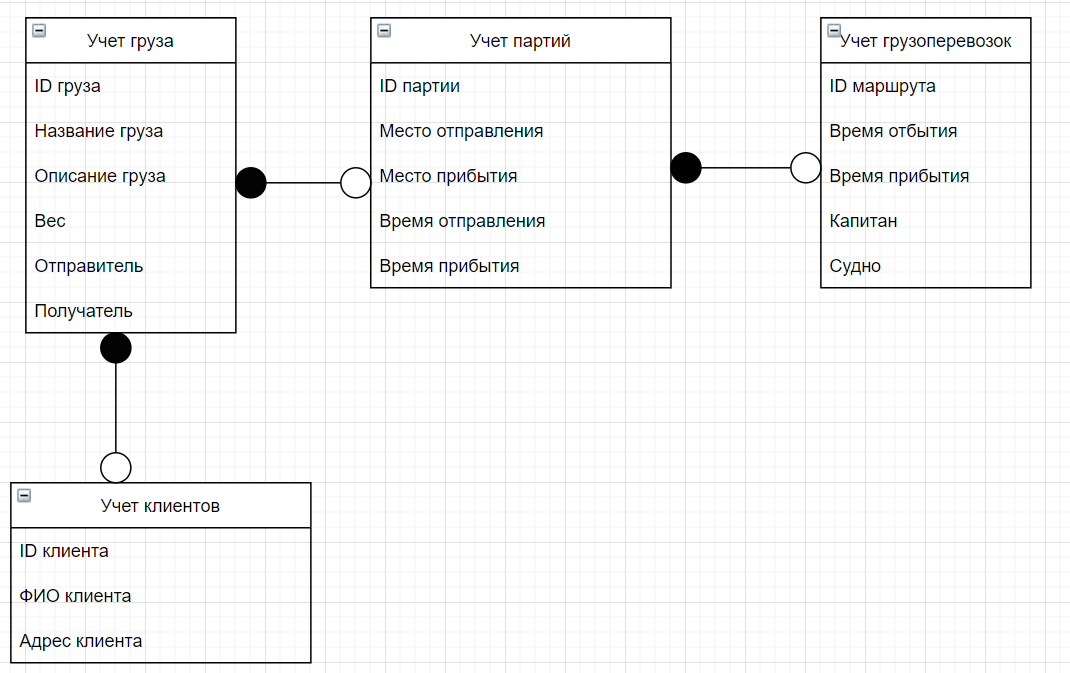
Обработка заявки

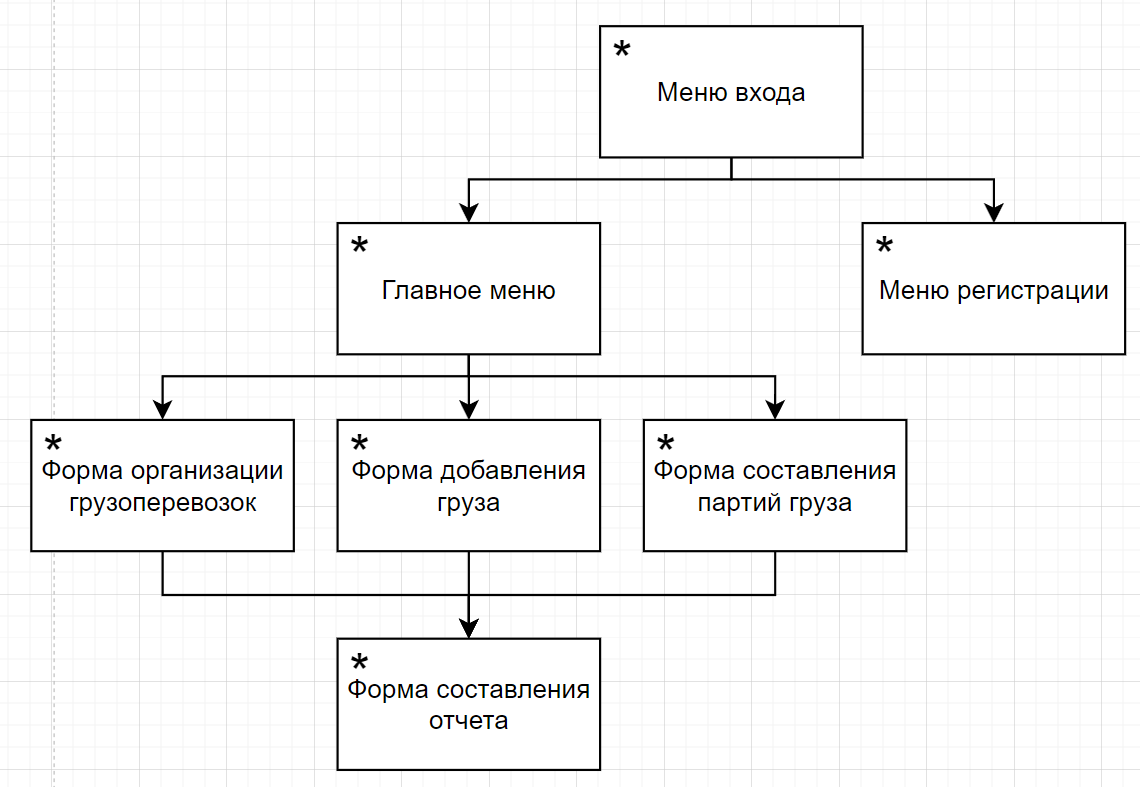
Покупатель

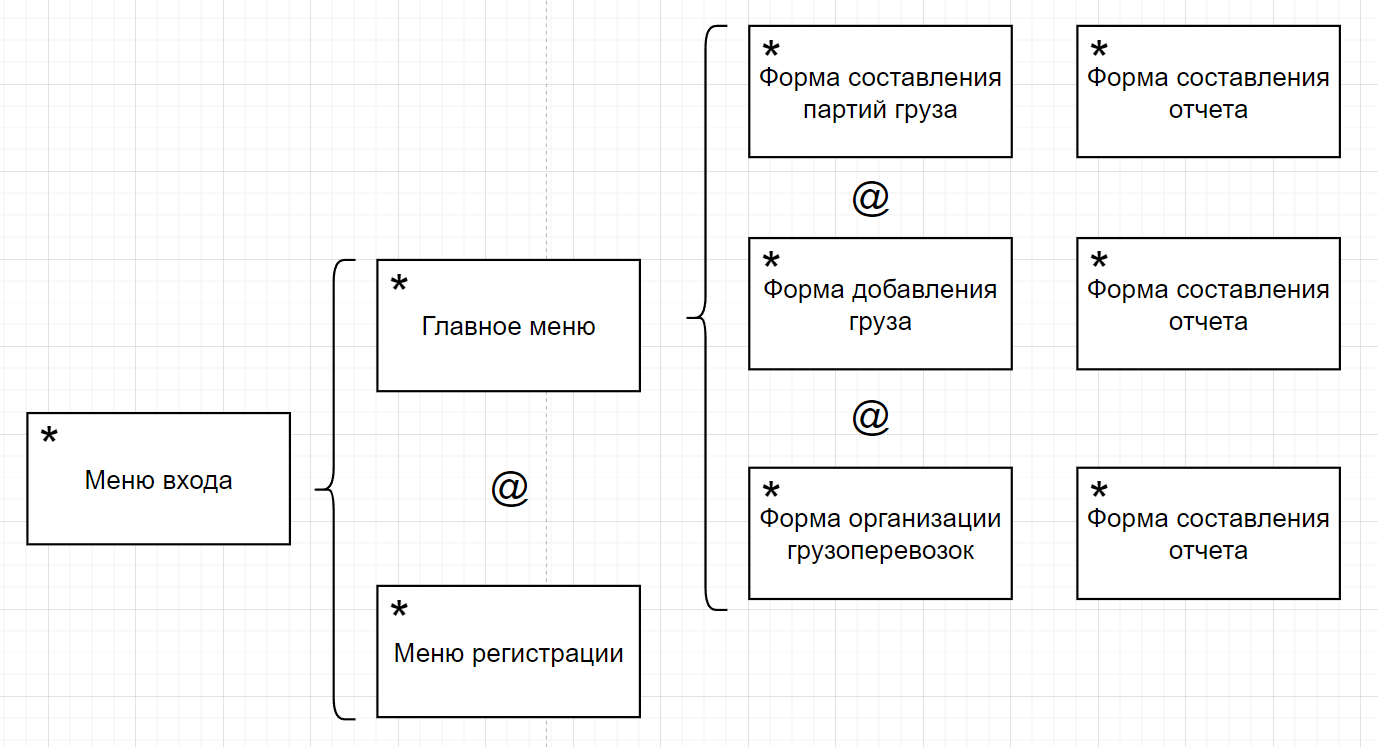


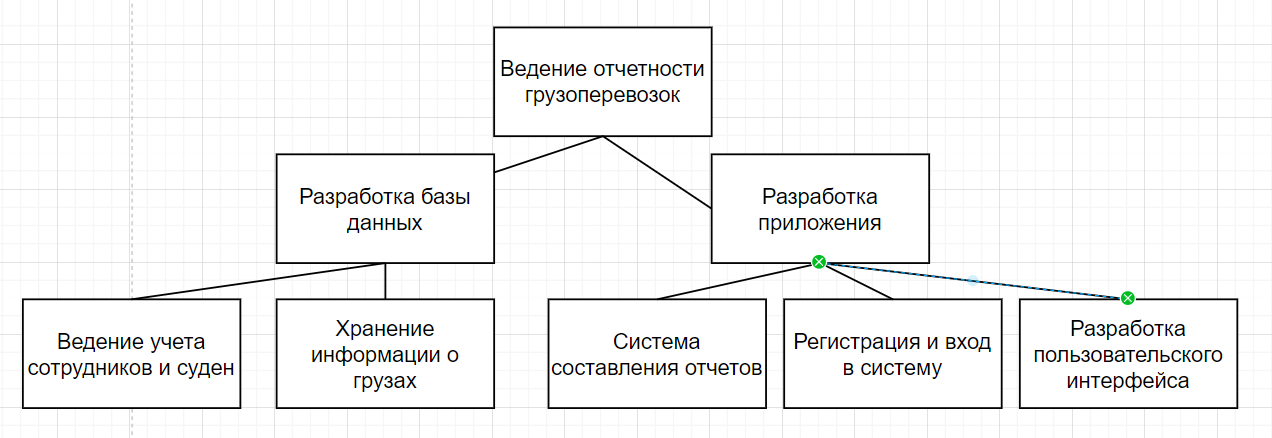




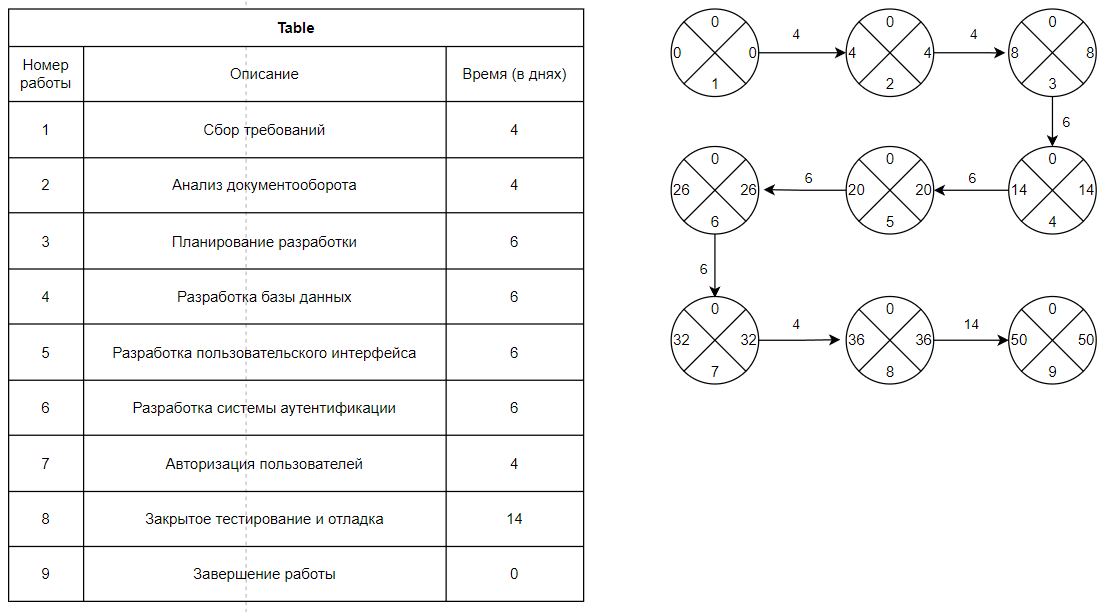












Классификация CASE- средств

1. По типам

Данная классификация отражает функциональное назначение CASE – средства в ЖЦ ПО.

В нее входит:

1. Анализ и проектирование

Средства этого типа используются для поддержки начальных этапов процесса разработки (1-6 работы). Средства данного типа поддерживают известные методологии анализа и проектирования. На выходе генерируются спецификации системы, ее компонентов и интерфейсов, связывающих эти компоненты, архитектура системы, архитектура ПО, технический проект ПО включая алгоритмы и определения структур данных.

1. Проектирование баз данных и файлов

Средства этого типа обеспечивают логическое моделирование данных, автоматическое преобразование данных в третью нормальную форму, автоматическую генерацию схем БД и описания форматов файлов на уровне программного кода.

1. Программирование и тестирование

Поддерживает 7 работу процесса разработки. Данное средство выполняет автоматическую кодогенерацию на основе спецификации или моделей. Содержит графические редакторы, средства поддержки работы с репозиторием, генераторы и анализаторы кодов, генераторы тестов, анализаторы покрытия тестами и отладчики.

1. Реинженерия

Общая цель средств этого типа является поддержка кодировки, изменения, преобразования, реинженерия существующей системы, поддержка документации по проекту. К нему относятся средства документирования, анализаторы программ, средства управления изменениями и конфигурациями ПО, средства реструктурирования и реинженерии, средства обеспечения мобильности ПО.

Средства реинженерии включают в себя статические анализаторы для генерации схем ПО из его кодов и оценки влияния модификации, динамические анализаторы, включающие трансляторы со встроенными отладочными возможностями, документаторы, автоматически обновляющие документацию при изменении кода, редакторы кода – автоматически изменяющие при редактировании кода предшествующие ему структуры.

Средства доступа к спецификациям позволяют выполнить их модификацию и генерацию модификационного кода. Средства реверсной инженерии, транслирующие коды в спецификации или модели.

1. Окружение

К средствам данного типа относят средства поддержки интеграции CASE средств и данных.

1. Управление проектом

Сюда относят средства поддержки ЖЦ ПО.

1. Классификация по категориям

Данная классификация отображает уровень интегрированности CASE средств по выполняемым функциям.

1. Tool

Включает средства самого низкого уровня интегрированности, они решают небольшую автономную задачу при разработке ПО. Низкий уровень интеграции.

1. Toolkit

Набор инструментов и пакет разработчика. Использует репозиторий. Средний уровень интеграции.

1. Workbench

Поддерживает весь ЖЦ ПО, использует репозиторий и поддерживает коллективную работу.

1. Классификация по уровням

Интеграция приложений и информационных систем

Информационные системы предприятия и их подсистемы:

1. Системы с эксплуатационным уровнем

Transaction Processing System – диалоговая обработка запросов

1. Система уровня знаний

Knowledge Work System,

Office Automation System

1. Управление информационными системами

Management Information System,

Decision System

1. Система с стратегическим уровнем

Executive Support System

TPS обслуживает основные и вспомогательные процессы, он является главным источником информации для других информационных модулей.

DSS – главный получатель данных из внутренних и внешних систем.

ESS является изолированным от основных производственных информационных систем, но может быть связан с другими системами при степени высокой автоматизации.

Интеграция информационной системы – объединение информационных систем, связывающее множество документов и отношения данных систем.

Под информационной системой понимается множество отношений различных документов, описывающих некоторые виды сущностей.

EAI – Enterprise application integration – Стандарт интеграции программных структур, объединяющий приложения, разработанных независимо друг от друга так, чтобы они могли работать как единое целое.

Существует два подхода к интеграции:

«Точка-точка» - все компоненты связаны друг с другом напрямую.

Интеграция по шине сервисов.

Уровни интеграции

В основном выделяют пять уровней интеграции:

1. Интеграция бизнес – процессов. Он основан на определении реализации и управления процессами обмена информации между различными бизнес - системами. Он представляет собой автоматизацию бизнес-процессов организации на основе единой инфраструктуры по созданию и управлению бизнес- процессами. Он позволяет моделировать, обеспечение соблюдений правил бизнес-процессов, предоставляют пользователям единый интерфейс, обеспечивать контроль над выполнением, вносить изменения в бизнес-процессы и получать данные для анализа.
2. Интеграция приложений

Основан на объединении данных или функций одного приложения с другим. Данные, которые передаются с помощью этой интеграции, могут быть в виде исходного файла или с необходимыми преобразованиями.

1. Интеграция данных

Основана на идентификации и каталогах данных с целью их дальнейшего использования. Данная интеграция обеспечивает гарантию качества интеграции и бизнес – процессов. При создании нам необходимо учитывать идентификацию, каталог, построение модели метаданных (описание данных). С помощью выполнения этих этапов данные можно распространять или использовать системы данных.

1. Интеграция на основе стандартов

Она основана на использовании стандартных форматов данных.

1. Интеграция платформ

Основывается на процессах и инструментах, с помощью которых можно осуществлять безопасный и оптимальный обмен информацией.

Качество программного интерфейса определяется индексом качества (0-1).

Открытость программного интерфейса также измеряется в пределах от 0 до 1.

Индекс интегрируемости приложения можно определить как индекс качества \* индекс открытости. В результате мы получим числовой показатель, который характеризует способность приложения быть частью другого приложения.

Принцип открытости информационных систем

Информационные системы – исчерпывающий и согласованный набор международных стандартов на информационные технологии и профили функциональных стандартов, которые специфицируют интерфейсы, службы, и поддерживающие их формы чтобы обеспечить их взаимодействие и мобильность программных приложений.

Классификация информационных систем по архитектуре

По степени распределенности:

1. Локальная

В нее входят все компоненты, работающие на одном компьютере. (Базы данных, СУБД, клиентские приложения).

1. Распределенная

Компоненты распределены по нескольким компьютерам. Делится на:

1. Файл – серверные информационные системы.
2. Клиент – серверные. Разделяются на
   1. Двухзвеньевые

Два звена: сервер баз данных, на котором находится БД и СУБД, и рабочая станция, на которой находятся клиентские приложения. Клиентские приложения обращаются к СУБД напрямую.

* 1. Многозвеньевые

Здесь добавляются промежуточные звенья, пользовательские клиентские приложения не обращаются к СУБД напрямую. Они взаимодействуют с промежуточными звеньями.

Архитектура SOA Service Oriented Architecture

Это прикладная архитектура, в которой все функции определены как независимые сервисы с вызываемыми интерфейсами. Обращение к этим сервисам позволяет реализовывать бизнес – процессы. Идея СОА заключается в создании архитектурной платформы, которая обеспечивает быструю консолидацию распределенных компонентов. Они объединяются в единые решения для поддержки бизнес – процессов.

Принципы:

1. Архитектура не привязана к определенной технологии.
2. Независимость организации системы от используемой вычислительной платформы.
3. Независимость от языка программирования.
4. Использование сервисов, независимых от конкретных приложений.
5. Организация сервисов как слабо связанных компонентов для построения системы.

Информационные услуги

Информационная услуга – услуга, ориентированная на удовлетворение информационных потребностей потребителя путем предоставления информационных продуктов. IT сервис в корпоративной среде – IT услуга, которую IT подразделение или внешний провайдер предоставляет бизнес – подразделениям предприятия для поддержки бизнес – процессов.

IT сервисные характеристики:

1. Функциональность
2. Время обслуживания
3. Доступность
4. Надежность
5. Производительность
6. Конфиденциальность
7. Масштаб
8. Затраты

Вебсайт – место в сети.

Портал – информационный ресурс, позволяющий получать информацию по определенной теме и изменять ее путем взаимодействия с компьютером.

Корпоративный информационный портал предназначен для создания единого информационного пространства компании и позволяет интегрировать в единое целое разнородные корпоративные приложения, предоставляя им единый интерфейс доступа.

Требования:

1. Обеспечение информационной поддержки сотрудников и клиентов компании
2. Организация коллективной работы и взаимодействие удаленных рабочих групп
3. Управление правами доступа
4. Управление публикацией
5. Организация доступа к приложениям и данным через веб – браузер.

Аутентификация – проверка принадлежности субъекту доступа предъявляемого им идентификатора.

Авторизация – предоставление определенному лицу прав на выполнение некоторых действий, процесс подтверждения прав пользователей на выполнение некоторых действий.

Авторизация происходит после аутентификации.

Типы интеграций:

1. Вертикальная
2. Горизонтальная

Интеграция ИС и приложений