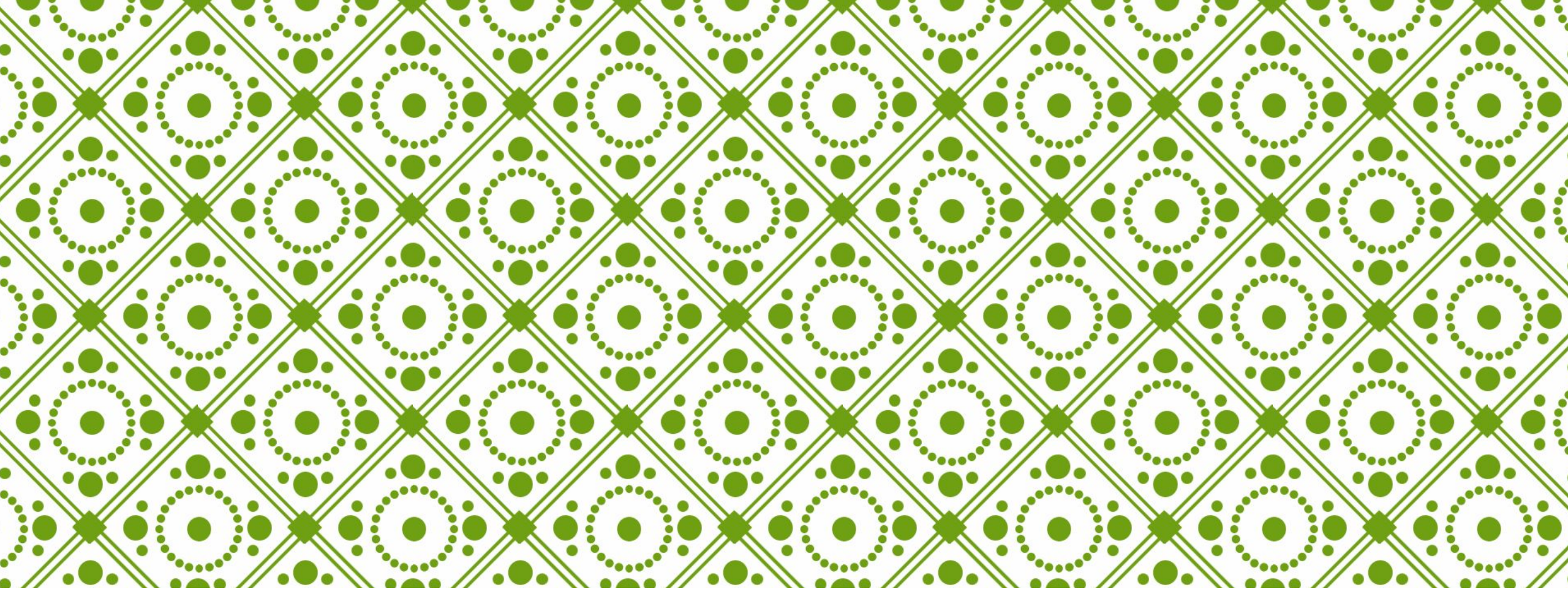


ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

Информационные технологии

ЖЦ ИС - это период создания и использования ИС, начиная с момента возникновения потребности в ИС и заканчивая моментом полного ее выхода из эксплуатации.

Типичный ЖЦ системы начинается с формулировки идеи или потребности, проходит все процессы разработки, производства, эксплуатации и сопровождения системы.

Стандартный ЖЦ состоит из процессов, каждый процесс характеризуется видами деятельности и задачами, которые выполняются на нем. Переход от одного процесса к другому должен быть санкционирован и определены входные и выходные данные.

Модель данного ЖЦ включает в себя процессы:

- определение требований;
- разработка (проектирование, конструирование);
- верификация, валидация, тестирование;
- изготовление;
- эксплуатация;
- сопровождение.

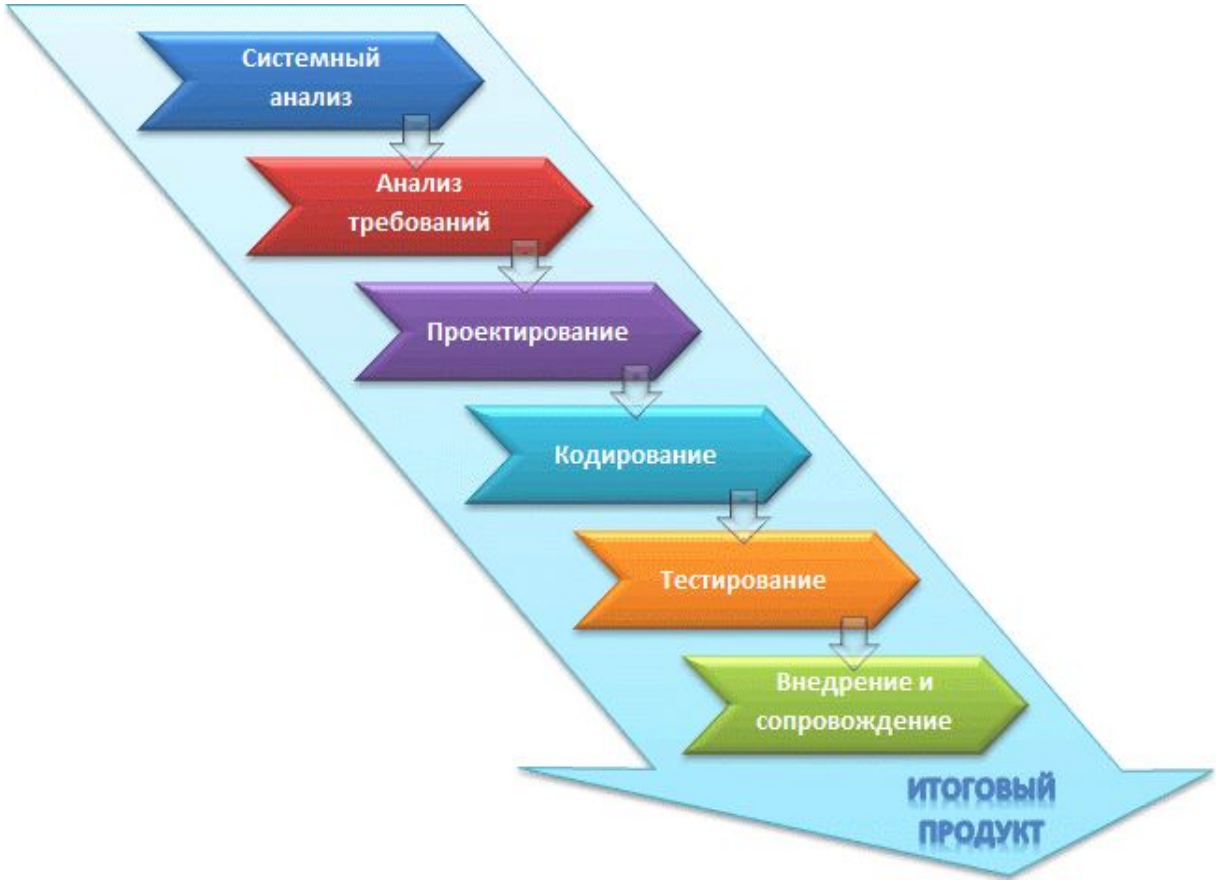
Модель кодирования и устранения ошибок

Самая простая из моделей очень часто применяемая студентами в учебном процессе. Алгоритм этой модели состоит из следующих шагов:

- Шаг 1: постановка задачи
- Шаг 2: создание программы
- Шаг 3: тестирование
- Шаг 4: анализ результата тестирования и возможный переход к шагу 1

Эта модель относится к первой группе и совсем не актуальна при профессиональной разработке программного обеспечения. По таким алгоритмам работали программисты 50-60 лет назад. Излишняя простота в данном случае не позволяет конкурировать с другими существующими моделями.

"Водопад" или каскадная модель жизненного цикла программного обеспечения



Преимущества:

- Последовательное выполнение этапов проекта в строгом фиксированном порядке
- Позволяет оценивать качество продукта на каждом этапе
- На каждой стадии формируется законченный набор документации, программного и аппаратного обеспечения, отвечающий критериям полноты и согласованности;
- Выполняемые в четкой последовательности стадии позволяют уверенно планировать сроки выполнения работ и соответствующие ресурсы (денежные, материальные и людские).

Недостатки:

- Отсутствие обратных связей между этапами
- Реальный процесс разработки информационной системы редко полностью укладывается в такую жесткую схему. Особенно это относится к разработке нетиповых и новаторских систем;
- Жизненный цикл основан на точной формулировке исходных требований к информационной системе. Реально в начале проекта требования заказчика определены лишь частично.

Каскадная модель будет давать отличный результат только в проектах с четко и заранее определенными требованиями и способами их реализации.

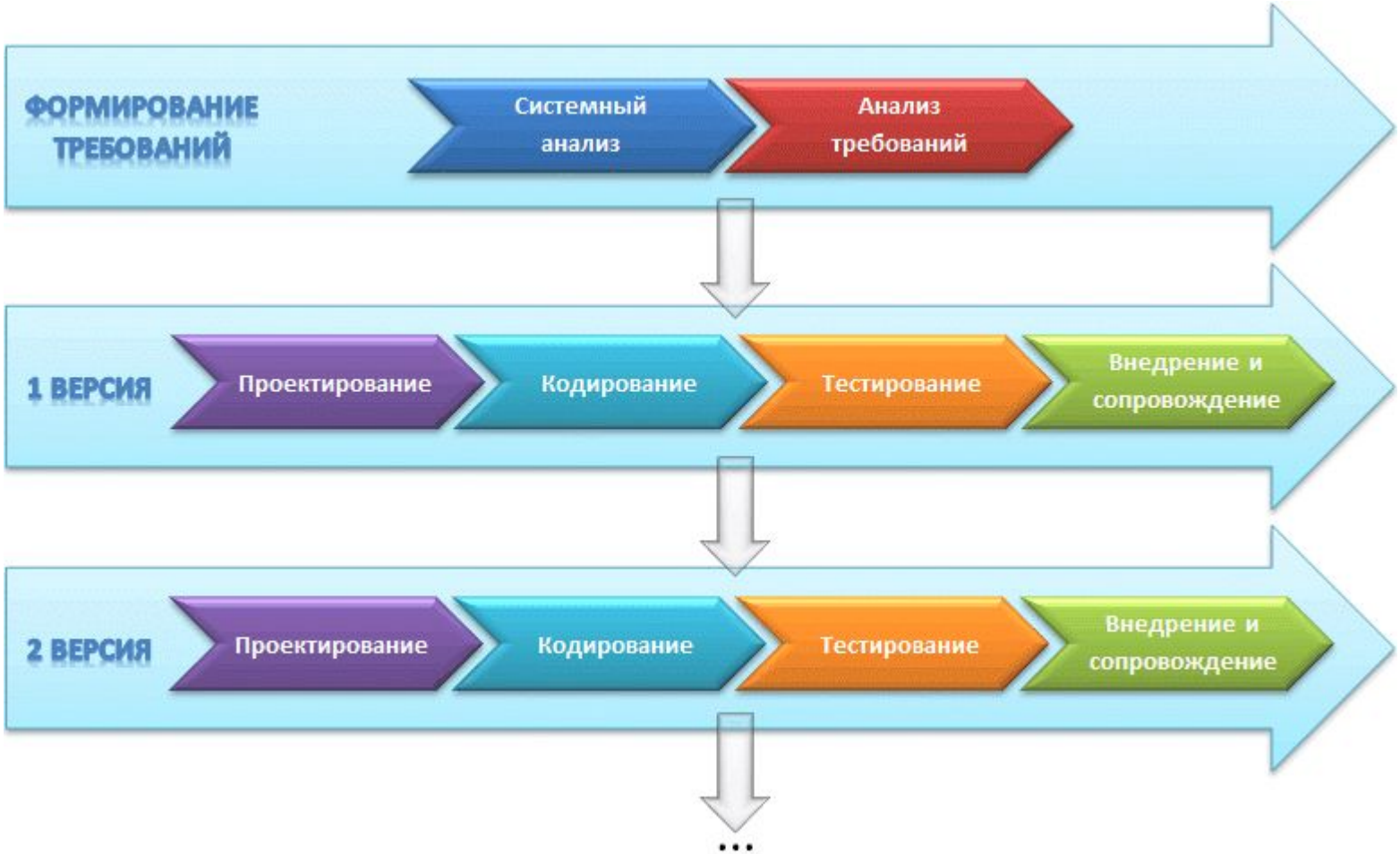
Когда использовать каскадную методологию?

- Только тогда, когда требования известны, понятны и зафиксированы. Противоречивых требований не имеется.
- Нет проблем с доступностью программистов нужной квалификации.
- В относительно небольших проектах.

"Водоворот" или каскадная модель с промежуточным контролем

В этой модели предусмотрен промежуточный контроль за счет обратных связей. Но это достоинство порождает и недостатки. Затраты на реализацию проекта при таком подходе возрастают практически в 10 раз.

Итеративная модель (итерационная модель)



Итерационная модель жизненного цикла не требует для начала полной спецификации требований. Вместо этого, создание начинается с реализации части функционала, становящейся базой для определения дальнейших требований. Этот процесс повторяется. Версия может быть неидеальна, главное, чтобы она работала.

В первой итерации есть лишь набросок Джоконды, во второй — появляются цвета, а третья итерация добавляет деталей, насыщенности и завершает процесс.

Итеративный процесс предполагает, что разные виды деятельности не привязаны намертво к определенным этапам разработки, а выполняются по мере необходимости, иногда повторяются, до тех пор, пока не будет получен нужный результат.

Вместе с гибкостью и возможностью быстро реагировать на изменения, итеративные модели привносят дополнительные сложности в управление проектом и отслеживание его хода. При использовании итеративного подхода значительно сложнее становится адекватно оценить текущее состояние проекта и спланировать долгосрочное развитие событий, а также предсказать сроки и ресурсы, необходимые для обеспечения определенного качества результата.

Когда оптимально использовать итеративную модель?

- Требования к конечной системе заранее четко определены и понятны.
- Проект большой или очень большой.
- Основная задача должна быть определена, но детали реализации могут эволюционировать с течением времени.

Инкрементная модель

- Цикл разработки разделен на более мелкие легко создаваемые модули.
- Каждый модуль проходит через фазы определения требований, проектирования, кодирования, внедрения и тестирования.
- Процедура разработки по инкрементной модели предполагает выпуск на первом большом этапе продукта в базовой функциональности, а затем уже последовательное добавление новых функций, так называемых «инкрементов».
- Процесс продолжается до тех пор, пока не будет создана полная система.

Когда использовать инкрементную модель?

- Когда основные требования к системе четко определены и понятны. В то же время некоторые детали могут дорабатываться с течением времени.
- Требуется ранний вывод продукта на рынок.

Спиральная модель жизненного цикла программного обеспечения

Достоинства:

-работающий продукт выпускается на более ранних стадиях;

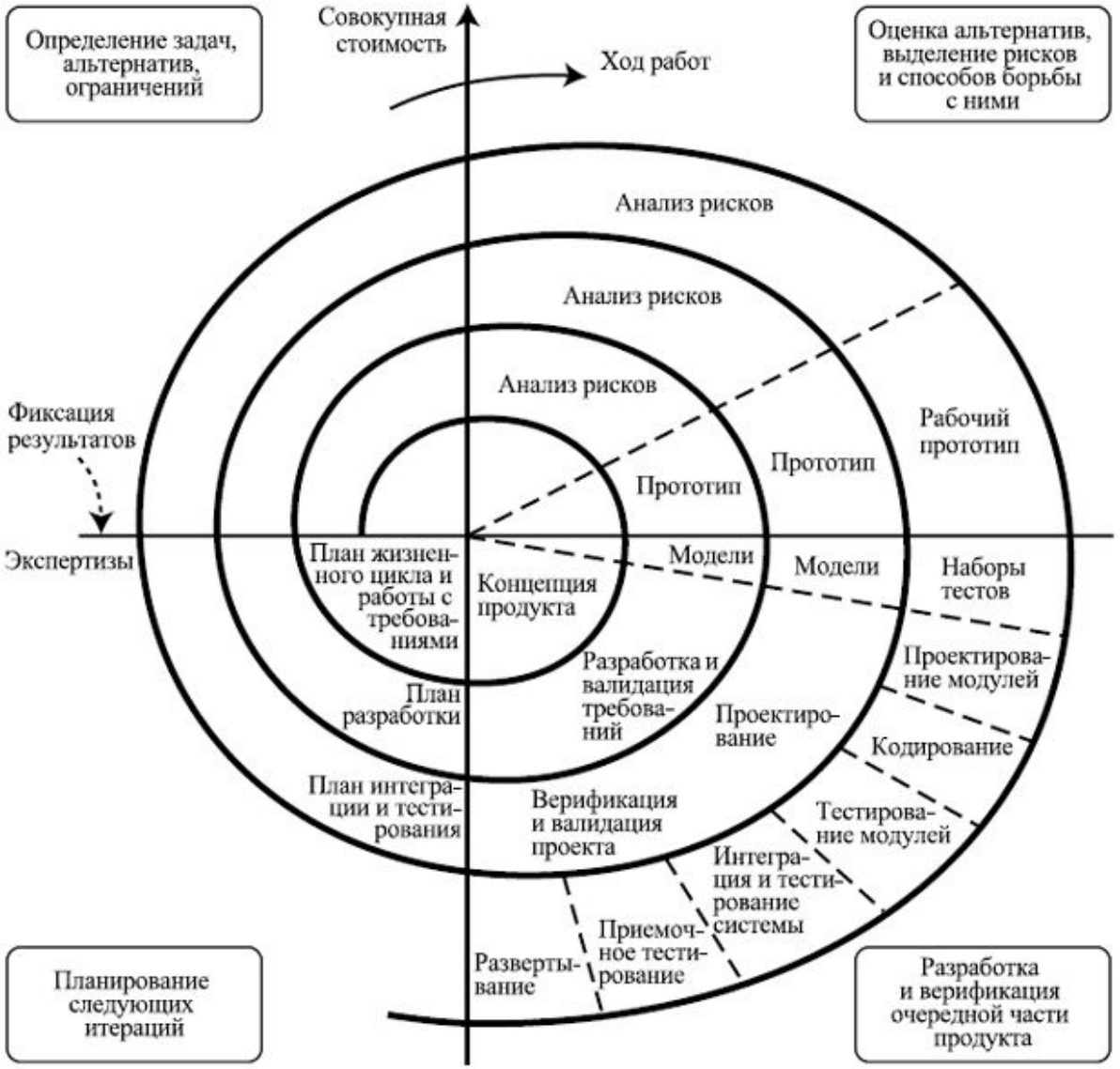
-позволяет получить более надежную и устойчивую систему. По мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации;

- не требуется полного и детального набора требований для начала разработки.

Недостатки:

- требуется очень хорошее знание предметной области;

- затруднены операции временного и ресурсного планирования всего проекта в целом. Для решения этой проблемы необходимо ввести временные ограничения на каждую из стадий жизненного цикла.



«Спиральная модель» похожа на инкрементную, но с акцентом на анализ рисков. Она хорошо работает для решения критически важных бизнес-задач, когда неудача несовместима с деятельностью компании, в условиях выпуска новых продуктовых линеек, при необходимости научных исследований и практической апробации.

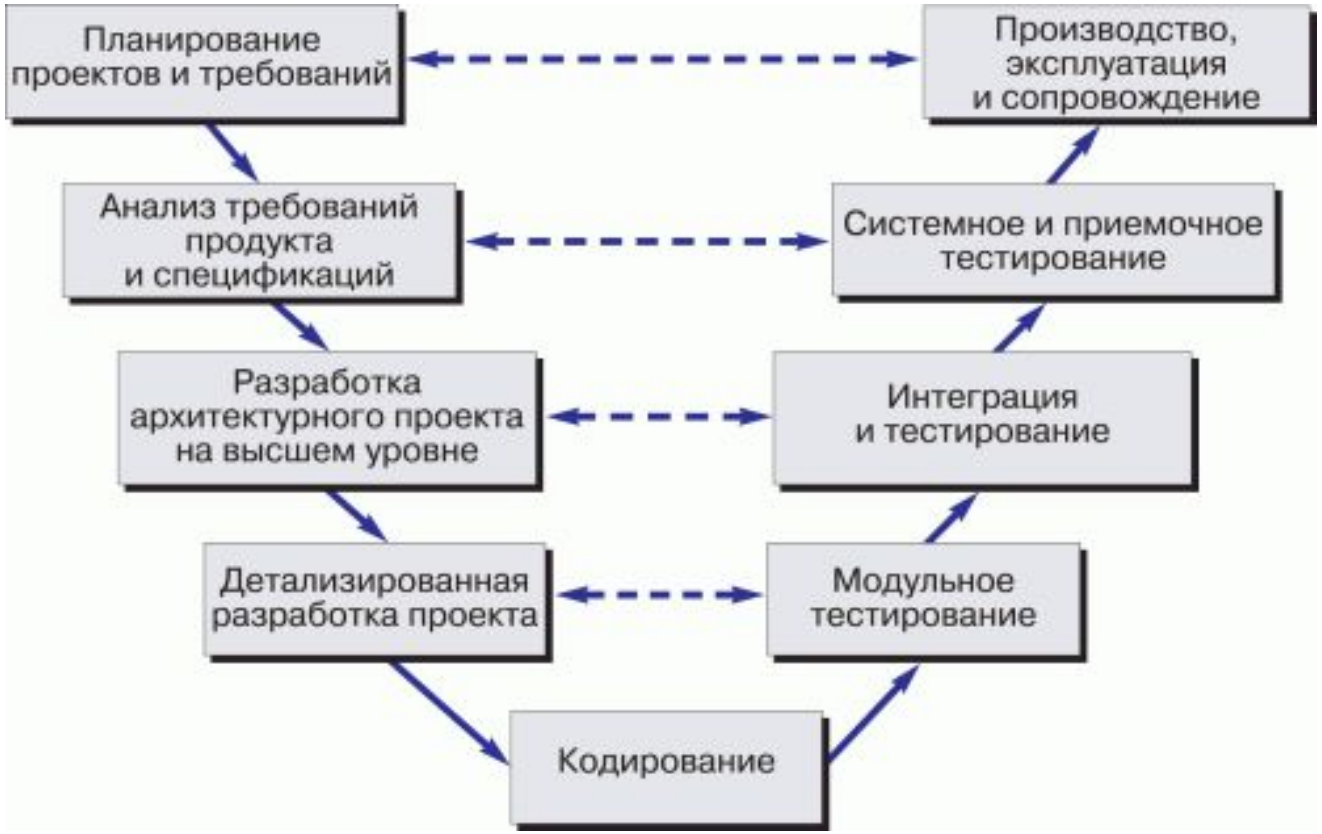
Спиральная модель предполагает 4 этапа для каждого витка:

1. планирование;
2. анализ рисков;
3. конструирование;
4. оценка результата и при удовлетворительном качестве переход к новому витку.

Эта модель не подойдет для малых проектов, она резонна для сложных и дорогих, например, таких, как разработка системы документооборота для банка, когда каждый следующий шаг требует большего анализа для оценки последствий, чем программирование.

V модель - разработка через тестирование

Особенностью модели можно считать то, что она направлена на тщательную проверку и тестирование продукта, находящегося уже на первоначальных стадиях проектирования. Стадия тестирования проводится одновременно с соответствующей стадией разработки, например, во время кодирования пишутся модульные тесты.



Данная модель имеет более приближенный к современным методам алгоритм, однако все еще имеет ряд недостатков. Является одной из основных практик экстремального программирования и предполагает регулярное тестирование продукта во время разработки.

V-образная модель применима к системам, которым особенно важно бесперебойное функционирование. Например, прикладные программы в клиниках для наблюдения за пациентами, интегрированное ПО для механизмов управления аварийными подушками безопасности в транспортных средствах и так далее.

Когда использовать V-модель?

- Если требуется тщательное тестирование продукта, то V-модель оправдает заложенную в себя идею: validation and verification.
- Для малых и средних проектов, где требования четко определены и фиксированы.
- В условиях доступности инженеров необходимой квалификации, особенно тестировщиков.

Быстрая разработка приложений (RAD Model)

RAD-модель — разновидность инкрементной модели. В RAD-модели компоненты или функции разрабатываются несколькими высококвалифицированными командами параллельно, будто несколько мини-проектов. Временные рамки одного цикла жестко ограничены. Созданные модули затем интегрируются в один рабочий прототип.

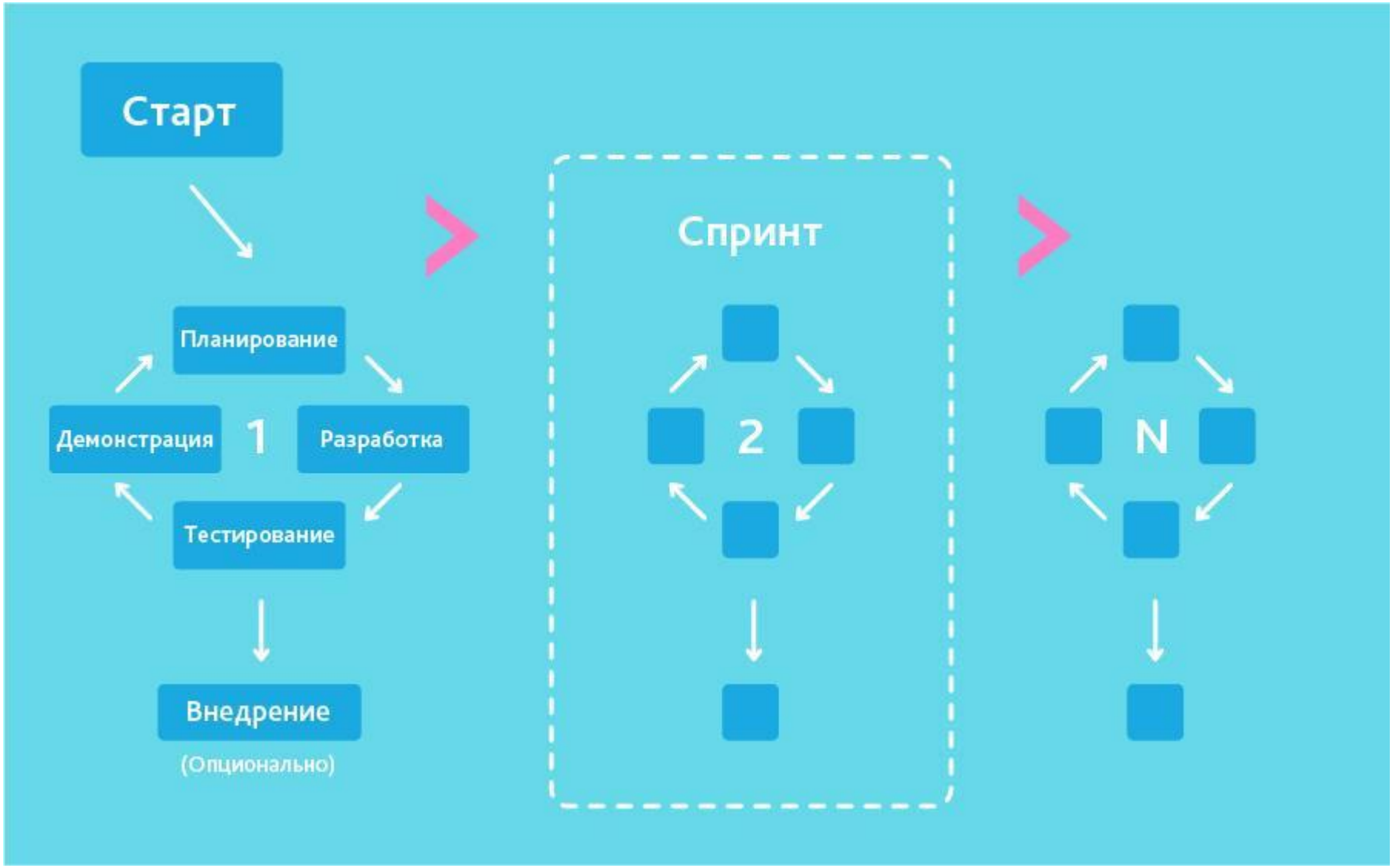
Модель быстрой разработки приложений включает следующие фазы:

- Бизнес-моделирование: определение списка информационных потоков между различными подразделениями.
- Моделирование данных: информация, собранная на предыдущем этапе, используется для определения объектов и иных сущностей, необходимых для циркуляции информации.
- Моделирование процесса: информационные потоки связывают объекты для достижения целей разработки.
- Сборка приложения: используются средства автоматической сборки для преобразования моделей системы автоматического проектирования в код.
- Тестирование: тестируются новые компоненты и интерфейсы.

Когда используется RAD-модель?

Может использоваться только при наличии высококвалифицированных и узкоспециализированных архитекторов.
Бюджет проекта большой, чтобы оплатить этих специалистов вместе со стоимостью готовых инструментов автоматизированной сборки. RAD-модель может быть выбрана при уверенном знании целевого бизнеса и необходимости срочного производства системы в течение 2-3 месяцев.

Гибкая методология разработки «Agile Model»



В «гибкой» методологии разработки после каждой итерации заказчик может наблюдать результат и понимать, удовлетворяет он его или нет. Это одно из преимуществ гибкой модели.
К ее недостаткам относят то, что из-за отсутствия конкретных формулировок результатов сложно оценить трудозатраты и стоимость, требуемые на разработку. Экстремальное программирование (XP) является одним из наиболее известных применений гибкой модели на практике.

В основе такого типа — непродолжительные ежедневные встречи — «Scrum» и регулярно повторяющиеся собрания (раз в неделю, раз в две недели или раз в месяц), которые называются «Sprint».

На ежедневных совещаниях участники команды обсуждают:
- отчёт о проделанной работе с момента последнего Scrum’а;
- список задач, которые сотрудник должен выполнить до следующего собрания;
- затруднения, возникшие в ходе работы.

Когда использовать Agile?

- Когда потребности пользователей постоянно меняются в динамическом бизнесе.
- Изменения на Agile реализуются за меньшую цену из-за частых инкрементов.
- В отличие от модели водопада, в гибкой модели для старта проекта достаточно лишь небольшого планирования.

Комплексы нормативных документов на разработку информационных систем

Обозначение	Наименование
Стандарты ISO/IEC (ИСО/МЭК) в области разработки и документирования программных средств	
ГОСТ Р ИСО/МЭК 12207-02	Информационная технология. Процессы жизненного цикла программных средств
ГОСТ Р ИСО/МЭК 15271-02	Информационная технология. Руководство по ИСО/МЭК 12207 (процессы жизненного цикла программных средств)
ГОСТ Р ИСО/МЭК 9126-93	Информационная технология. Оценка программной продукции. Характеристики качества и руководства по их применению
ГОСТ Р ИСО/МЭК 12119-94	Информационная технология. Пакеты программ. Требования к качеству и тестирование
Комплекс нормативных документов на автоматизированные системы	
ГОСТ 34.003-90	Автоматизированные системы. Термины и определения
ГОСТ 34.201-89	Виды, комплектность и обозначение документов при создании автоматизированных систем
ГОСТ 34.601-90	Автоматизированные системы. Стадии создания
ГОСТ 34.602-89	Техническое задание на создание автоматизированной системы
РД 50-698-90	Автоматизированные системы. Требования к содержанию документов
РД 50-34.126-92	Рекомендации. Правила проведения работ при создании автоматизированных систем
Комплекс стандартов Единой системы программной документации (ЕСПД)	
ГОСТ 19.101-77	Виды программ и программных документов
ГОСТ 19.102-77	Стадии разработки
ГОСТ 19.105-78	Общие требования к программным продуктам
ГОСТ 19.201-78	Техническое задание. Требования к содержанию и оформлению
ГОСТ 19.701-90 (ИСО/МЭК 5807-85)	Схемы алгоритмов программ, данных и систем. Условные обозначения и правила выполнения
Комплекс отраслевых руководящих методических материалов на информационные системы на железнодорожном транспорте (ОРММ ИСЖТ)	
ОРММ ИСЖТ 5.03-00	Процессы жизненного цикла ИС и программных средств
ОРММ ИСЖТ 2.01-00	Требования к составу, содержанию и оформлению документов при создании ИС
ОРММ ИСЖТ 2.02-00	Порядок представления, согласования и утверждения документов, разрабатываемых при создании ИС

Основным нормативным документом, регламентирующим состав процессов ЖЦ ПО, является международный стандарт ISO/IEC 12207: 1995

Следует отметить, что в России создание ПО первоначально, в 70-е гг., регламентировалось стандартами ГОСТ ЕСПД (Единой системы программной документации – серия ГОСТ 19.XXX), которые были ориентированы на класс относительно простых программ небольшого объема, создаваемых отдельными программистами.

Процессы создания автоматизированных систем (АС), в состав которых входит и ПО, регламентированы стандартами ГОСТ 34.601-90 «Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания», ГОСТ 34.602-89 «Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы» и ГОСТ 34.603-92 «Информационная технология. Виды испытаний автоматизированных систем».

Федеральным агентством по техническому регулированию и метрологии РФ 01.03.2012 г. взамен ГОСТ Р ИСО/МЭК 12207-99 принят стандарт ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств», идентичный международному стандарту ISO/IEC 12207:2008 «System and software engineering — Software life cycle processes».

Жизненный цикл – период:

- с момента принятия решения о создании ПС
- до момента полного прекращения эксплуатации

Стандарты:

- ISO/IEC 12207: 2008 “System and software engineering — Software life cycle processes”
  - ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств»
- **Устаревшие ГОСТы:**
  - ГОСТ 19.XXX (Единая система программной документации)
  - ГОСТ 34.XXX (Комплекс стандартов на АС)

Стадии жизненного цикла информационной системы:

1. Предпроектное обследование:

- сбор материалов для проектирования, при этом выделяют формулирование требований, с изучения объекта автоматизации, даются предварительные выводы предпроектного варианта ИС;

- анализ материалов и разработка документации, обязательно дается технико экономическое обоснование с техническим заданием на проектирование ИС.

2. Проектирование:

2.1 предварительное проектирование;

- выбор проектных решений по аспектам разработки ИС;

- описание реальных компонент ИС;

- оформление и утверждение технического проекта (ТП).

2.2 детальное проектирование:

- выбор или разработка математических методов или алгоритмов программ;

- корректировка структур БД;

- создание документации на доставку и установку программных продуктов;

- выбор комплекса технических средств с документацией на ее установку.

2.3 разработка техно-рабочего проекта ИС (ТРП).

2.4 разработка методологии реализации функций управления с помощью ИС и описанием регламента действий аппарата управления.

3. Разработка ИС:

- получение и установка технических и программных средств;
- тестирование и доводка программного комплекса;
- разработка инструкций по эксплуатации программно-технических средств.

4. Ввод ИС в эксплуатацию:

- ввод технических средств;
- ввод программных средств;
- обучение и сертификация персонала;
- опытная эксплуатация;
- сдача и подписание актов приемки-сдачи работ.

5. Эксплуатация ИС:

- повседневная эксплуатация;
- общее сопровождение всего проекта.

Характеристика процессов стандарта ISO/IEC 12207

Процессы данного стандарта разбиты по группам: основные, вспомогательные и организационные.

К основным процессам стандарта относятся:

- приобретение (*acquisition*);
- поставка (supply);
- разработка (development);
- эксплуатация (operation);
- сопровождение (maintenance).

Процесс приобретения инициирует ЖЦ ПО и определяет действия организации-покупателя (или заказчика), которая приобретает автоматизированную систему, программный продукт или сервис.

Процесс поставки определяет действия предприятия-поставщика, которое снабжает покупателя системой, программным продуктом или сервисом.

Процесс разработки состоит в изготовлении исполнителем проекта программного продукта на процессах ЖЦ: разработка требований, проектирование, кодирование, тестирование и интеграция.

Процесс эксплуатации определяет действия оператора по обслуживанию системы, использованию ее пользователями, изучившими ее возможности для удовлетворения своих потребностей в плане обработки данных или вычислений.

Процесс сопровождения состоит в выполнении предписанных действий по инсталляции системы, запуску функций, а также по управлению модификациями и поддержанием системы в рабочем состоянии.

К вспомогательным процессам стандарта относятся процессы:

- документирования (documentation);
- управления конфигурацией (configuration management);
- обеспечения качества (*quality assurance*);
- верификации (verification);
- валидации (validation);
- совместного анализа (оценки) (joint review);
- аудита (audit).

Вспомогательные процессы поддерживают реализацию основных процессов и способствуют получению требуемого качества ПО. Они инициируются другими процессами.

К организационным процессам стандарта относятся процессы :

- управления (management);
- создания инфраструктуры (infrastructure);
- усовершенствования (improvement);
- обучения (training).

РАЗВИТИЕ МОДЕЛИ ПРОЦЕССОВ ЖИЗНЕННОГО ЦИКЛА. ГОСТ Р ИСО/МЭК 15288

Стандарт ГОСТ Р ИСО/МЭК 15288 "*Процессы жизненного цикла* систем" (ГОСТ 15288, 2005) – объемный (почти 60 страниц) и сложный документ, задуманный, *по*-видимому, как расширение ГОСТ Р ИСО/МЭК 12207 на системы, "которые созданы человеком и состоят из одного или нескольких следующих элементов: технические средства, *программные средства*, люди, процессы (например, процесс оценки), процедуры (например, инструкции оператора), *основные средства* и природные ресурсы (например, вода, объекты живой природы, минералы)". Его отличают *точность* и продуманность формулировок, структурированность изложения. Структурно он следует ГОСТ Р ИСО/МЭК 12207 – в частности, здесь, как и там, процессы разбиваются на группы, хотя принцип разбиения совершенно другой. Теперь эти группы выглядят следующим образом:

- процессы соглашения;
- процессы предприятия;
- процессы проекта;
- технические процессы.

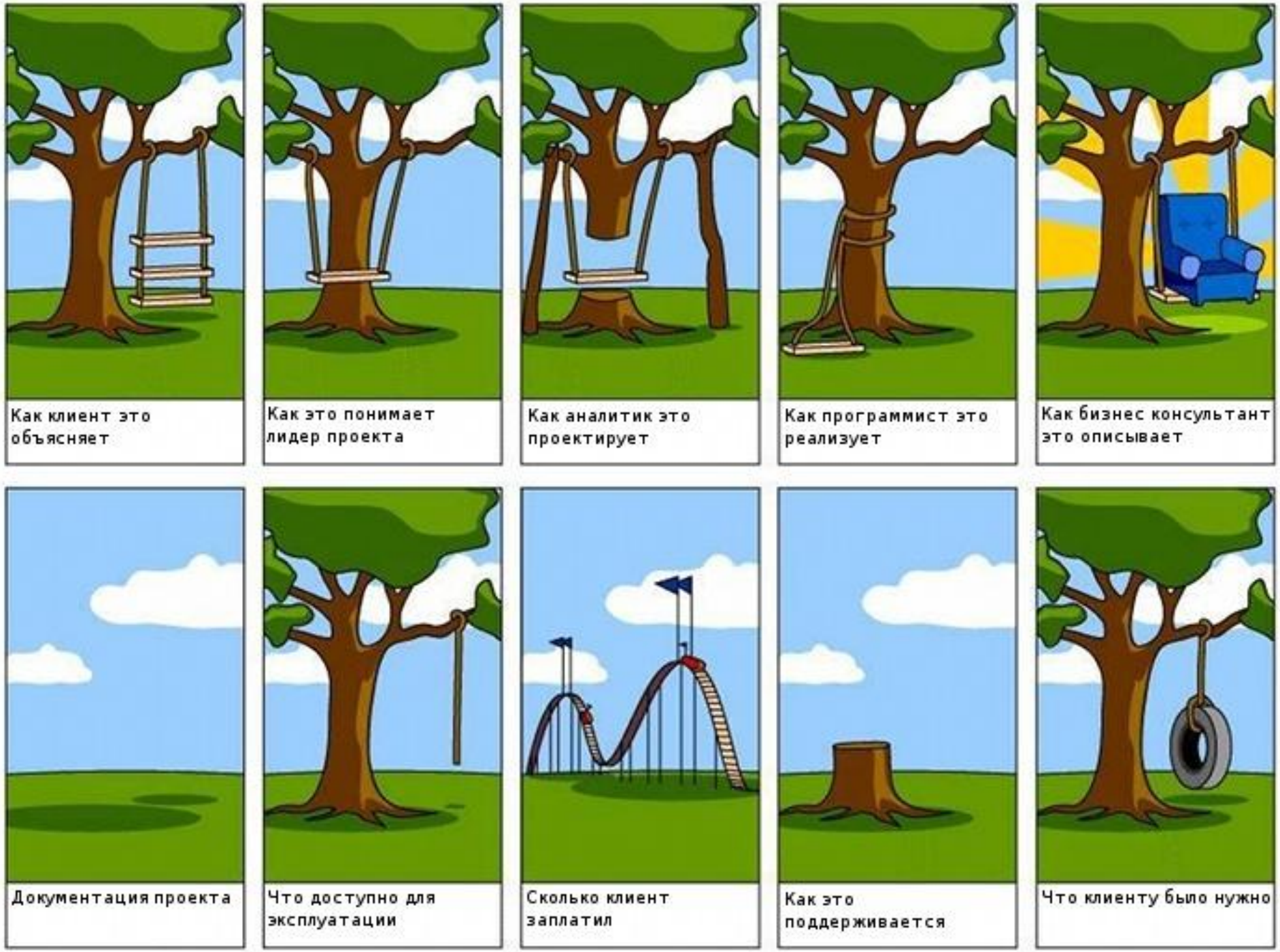
Структура стандарта и *связь* его с ИСО/МЭК 12207 показаны на рисунке. Здесь последний представлен своей последней версией, которая называется Изменение № 1 к ИСО/МЭК 12207.

С точки зрения структуры ГОСТ Р ИСО/МЭК 15288 различия между ГОСТ ИСО/МЭК 12207 и Изменением № 1 к ИСО/МЭК 12207 несущественны.

Как видим, ИСО/МЭК 12207 детализирует процесс реализации элементов системы, которые могут включать и *программные средства*.

В Приложении С к стандарту точно определяется, процессам какого из двух стандартов (т. е. ГОСТ Р ИСО/МЭК 15288 и ГОСТ Р ИСО/МЭК 12207) следует отдавать предпочтение при построении программной системы.

Расширенное описание применения стандарта можно найти по этой ссылке:
http://www.intuit.ru/studies/courses/2298/598/lecture/12856


Как клиент это объясняет
Как это понимает лидер проекта
Как аналитик это проектирует
Как программист это реализует
Как бизнес консультант это описывает
Документация проекта
Что доступно для эксплуатации
Сколько клиент заплатил
Как это поддерживается
Что клиенту было нужно

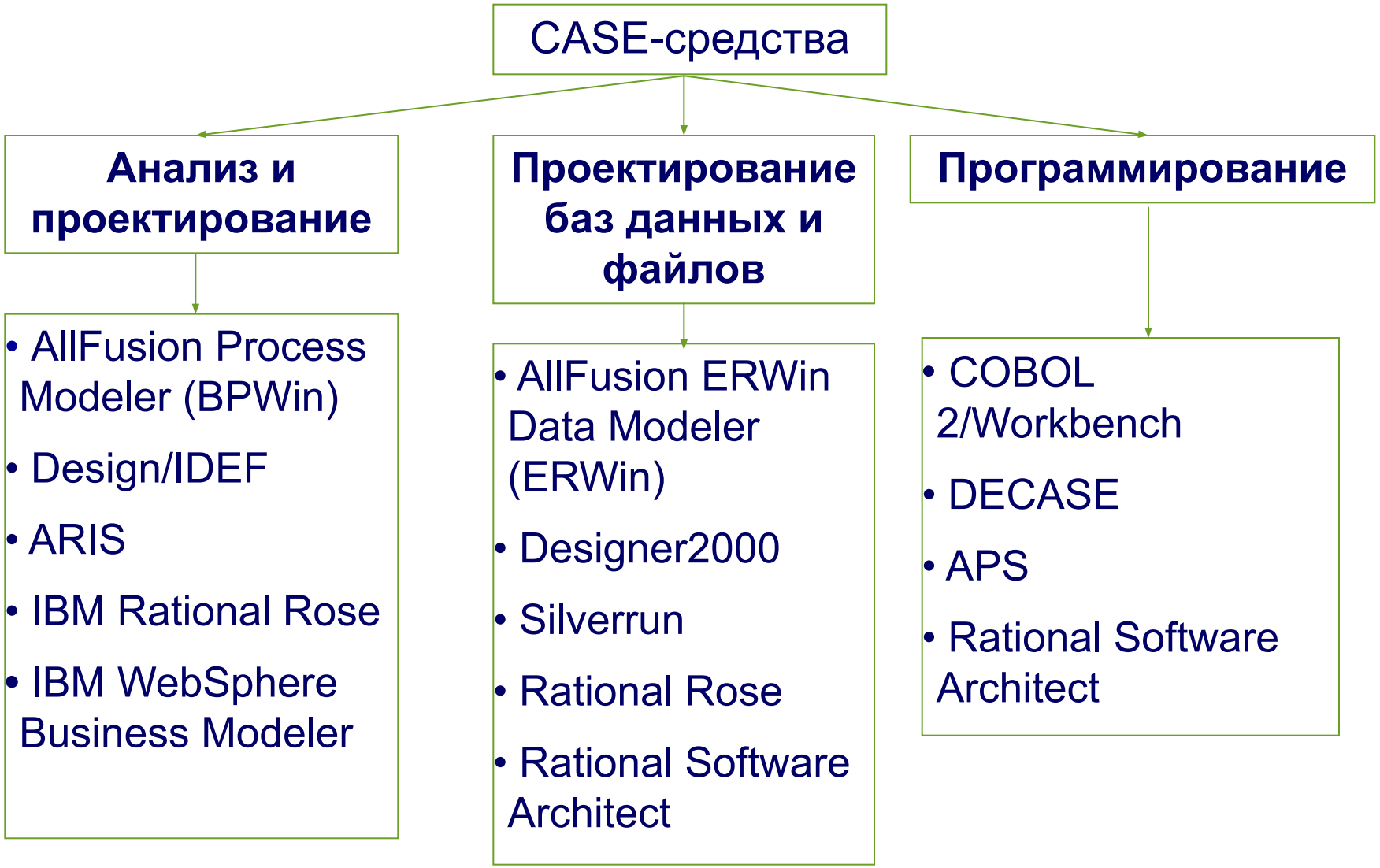
ПОДХОДЫ К МОДЕЛИРОВАНИЮ БИЗНЕС-ПРОЦЕССОВ

2 подхода
к моделированию
бизнес-процессов

Функционально-модульный (структурный) подход – основан на принципе функциональной декомпозиции

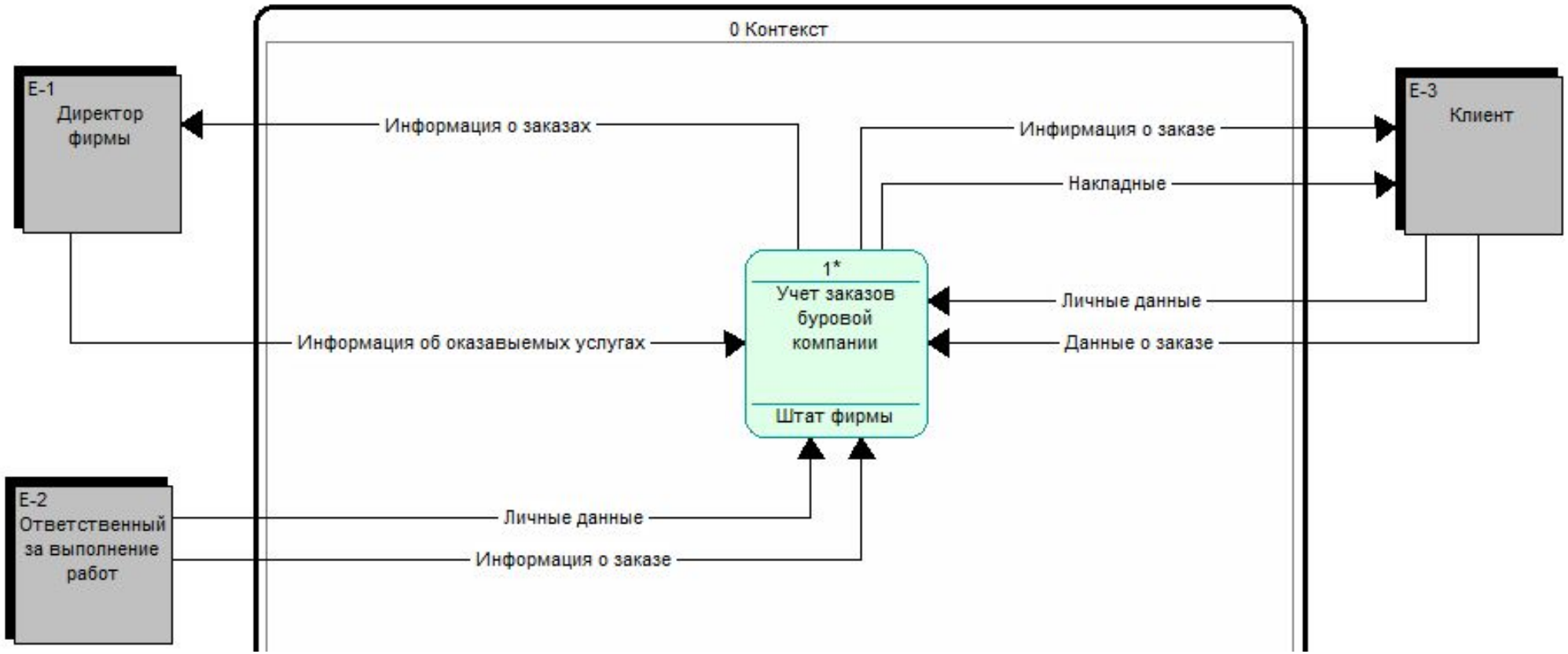
Объектно-ориентированный подход – основан на принципе объектной декомпозиции

ПРИМЕРЫ НЕКОТОРЫХ CASE-СРЕДСТВ

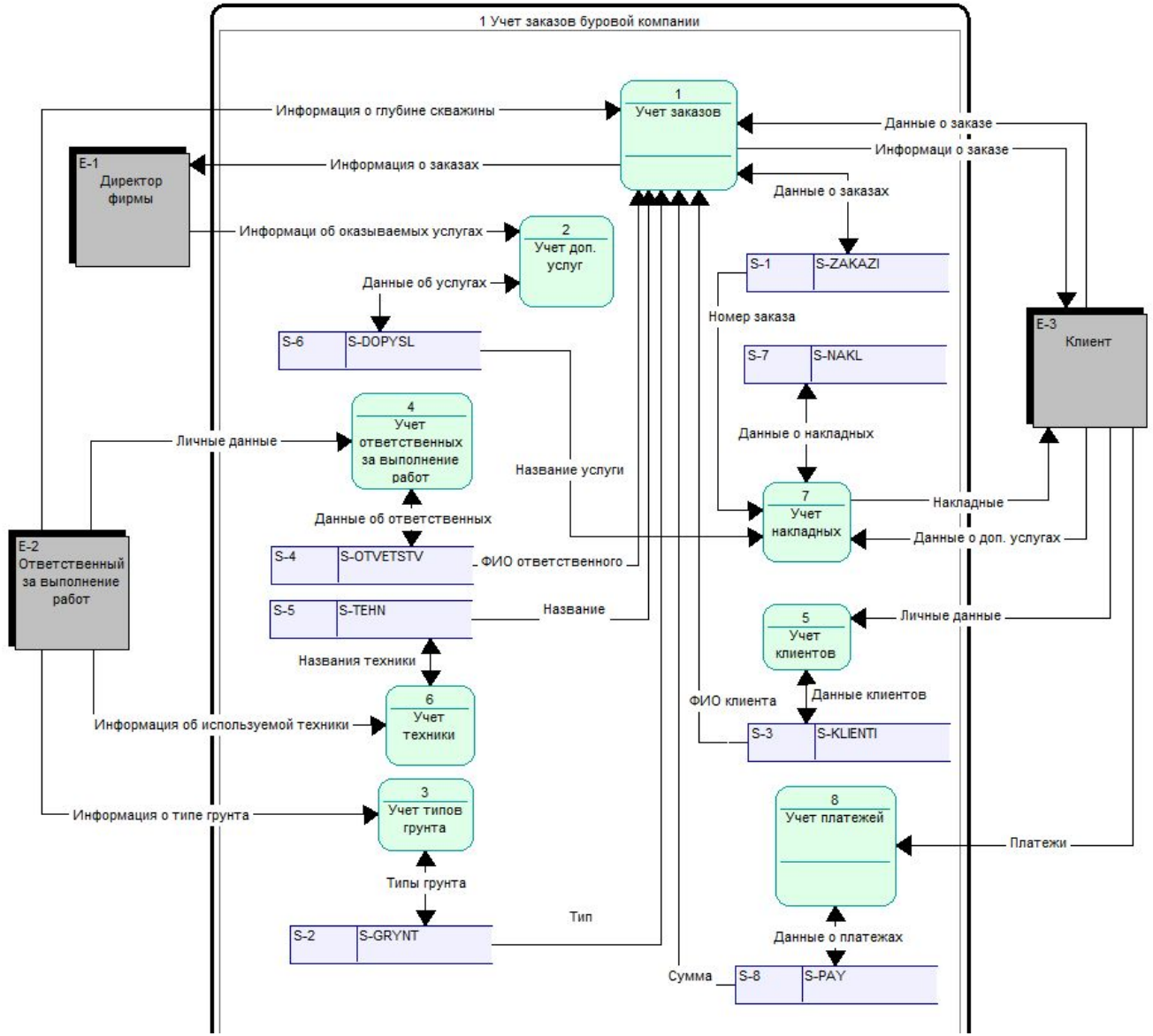


СТРУКТУРНЫЙ ПОДХОД

Методология	Тип разрабатываемой модели
SADT (Structured Analysis and Design Technique, методология структурного анализа и проектирования)	Функциональная
DFD (Data Flow Diagrams, диаграммы потоков данных)	Функциональная или компонентная
ERD (Entity-Relationship Diagrams, диаграммы "сущность-связь")	Информационная
Flowcharts (блок-схемы)	Поведенческая
EPC (Event-driven Process Chain, событийная цепочка процессов)	Функциональная или поведенческая
BPMN (Business Process Model and Notation, модель и нотация бизнес-процессов)	Функциональная или поведенческая

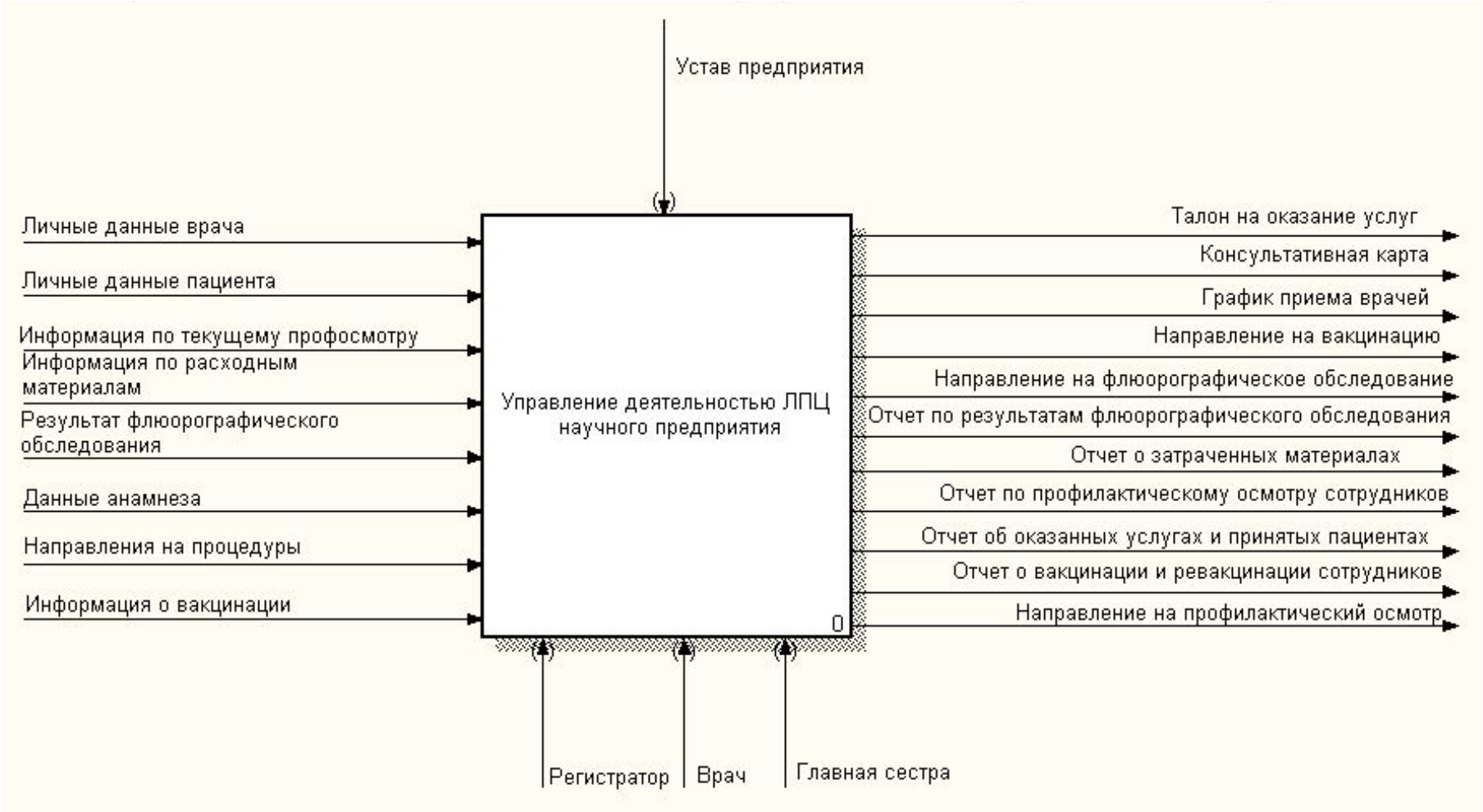


Контекстная диаграмма бизнес-процесса, разработанная с использованием CASE – средства Open Model Shere

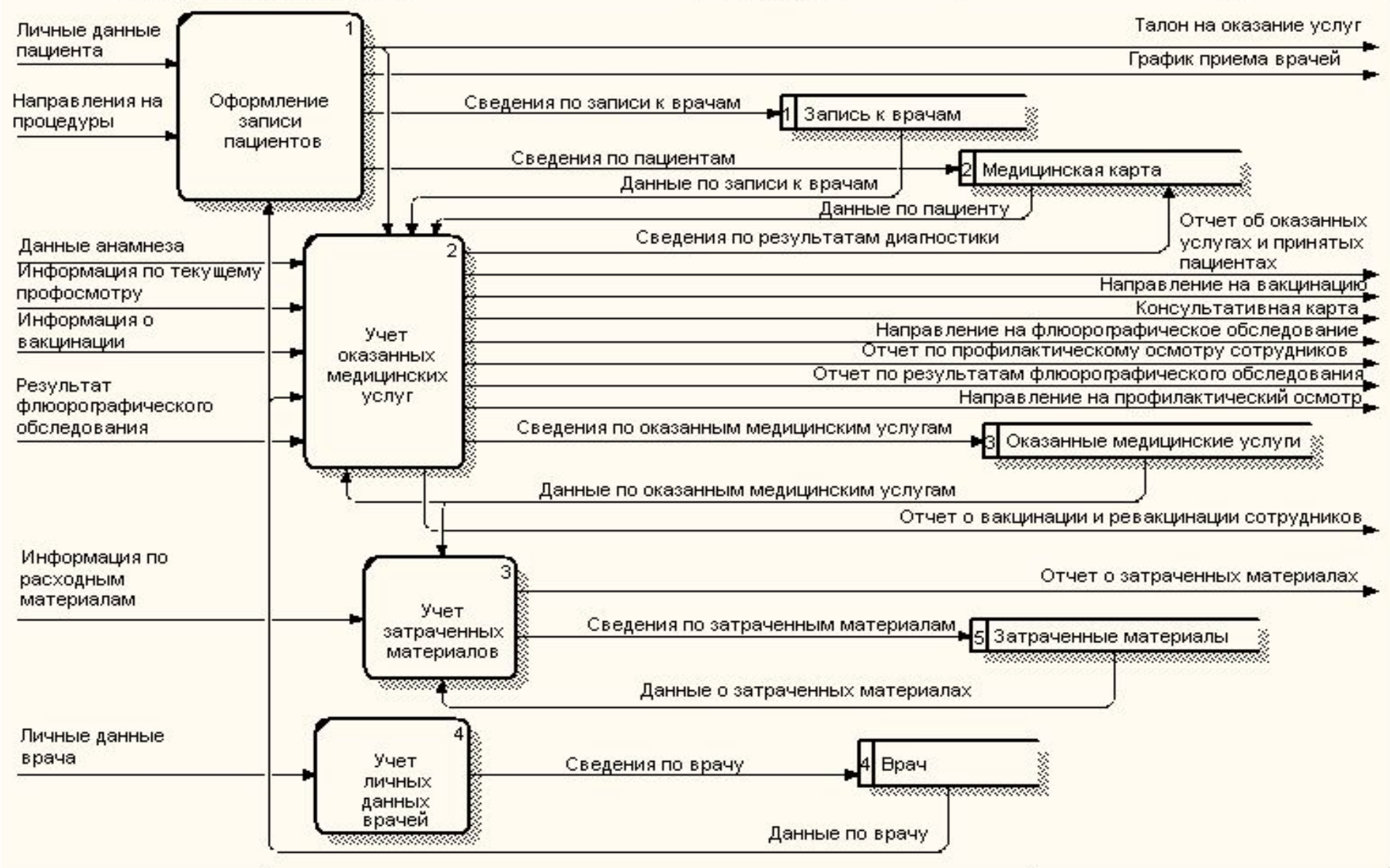


Детализирующая диаграмма бизнес-процесса

SADT (Structured Analysis and Design Technique, методология структурного анализа и проектирования

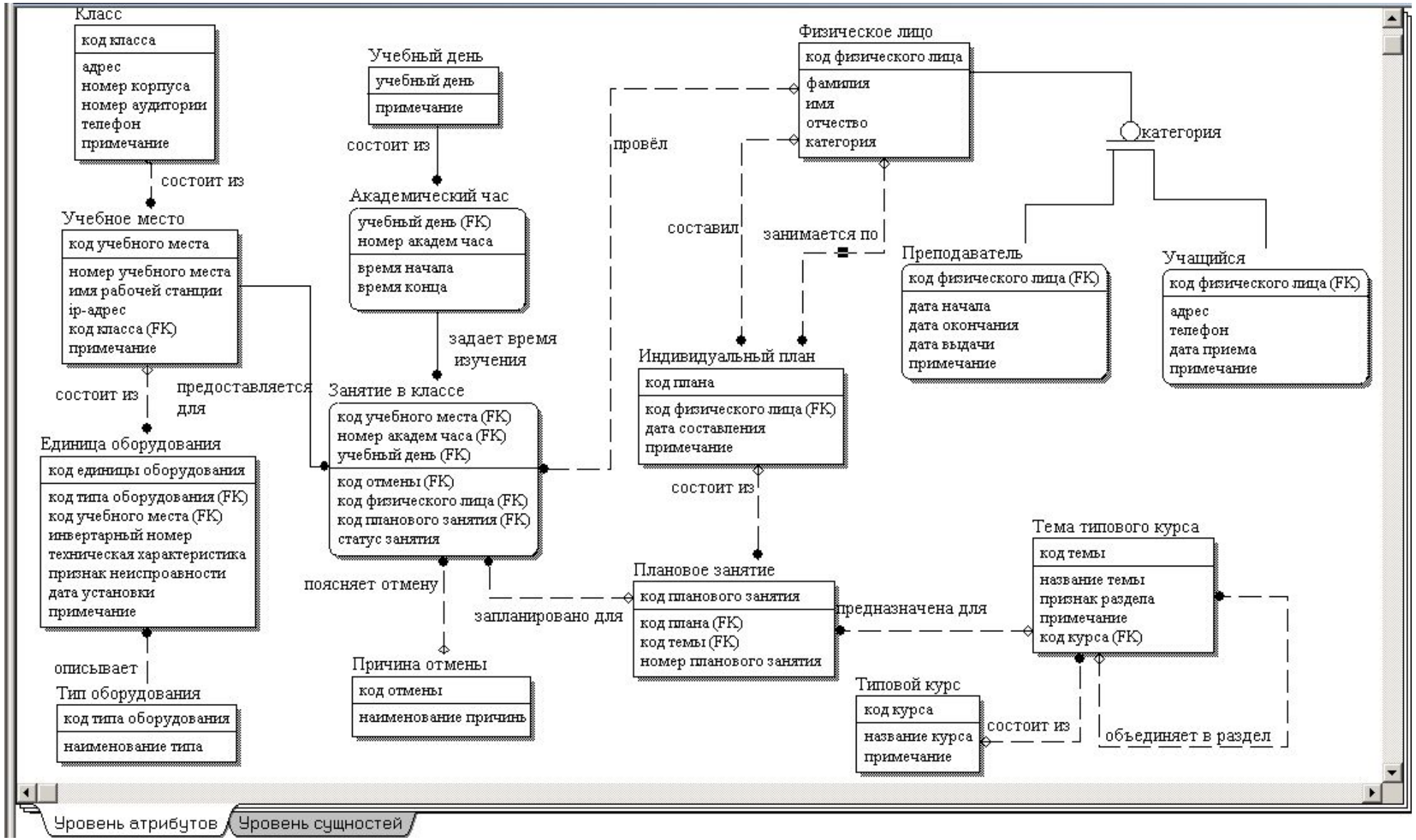


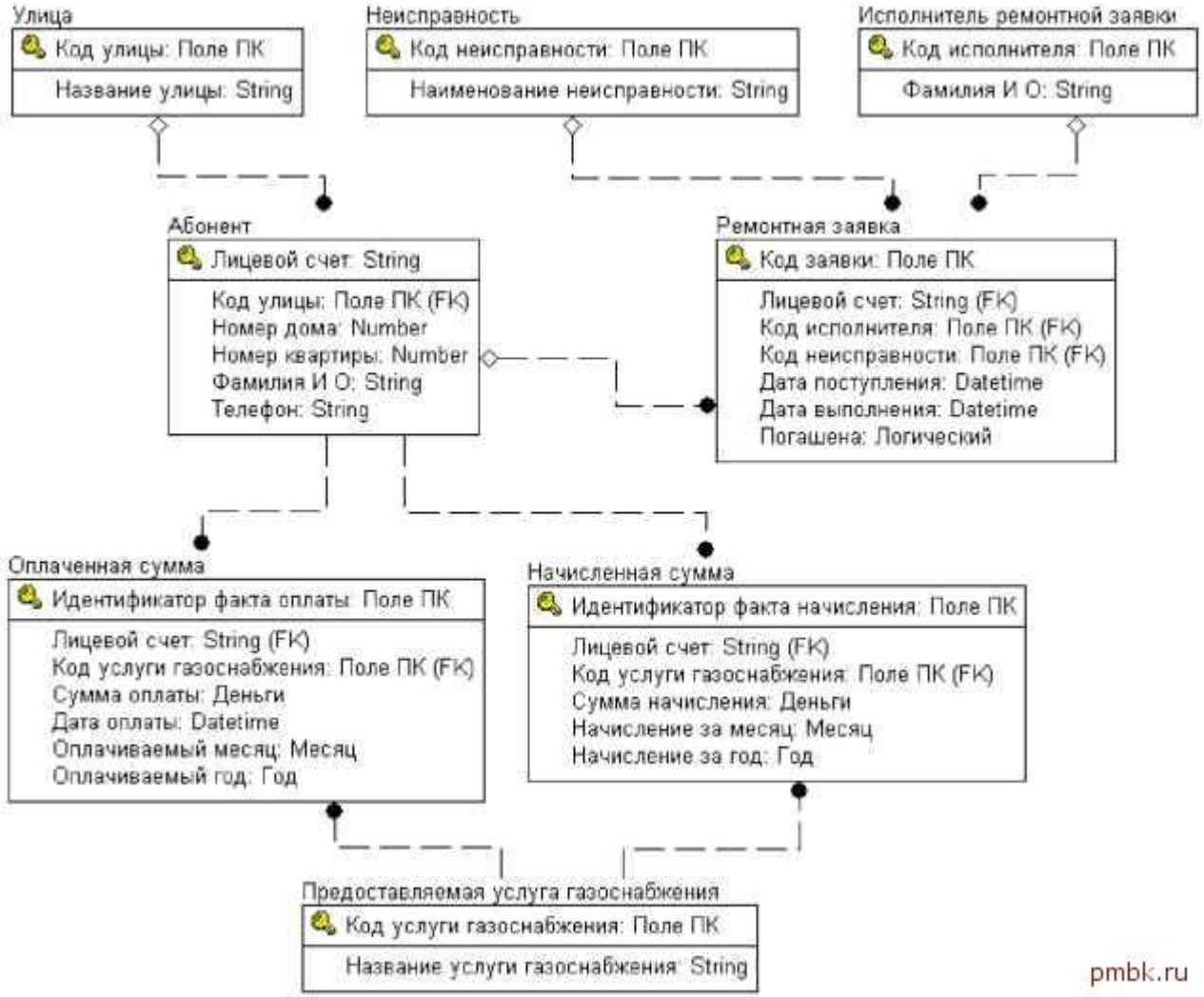
Контекстная диаграмма бизнес-процесса, разработанная с использованием CASE – средства CA Process Modeller

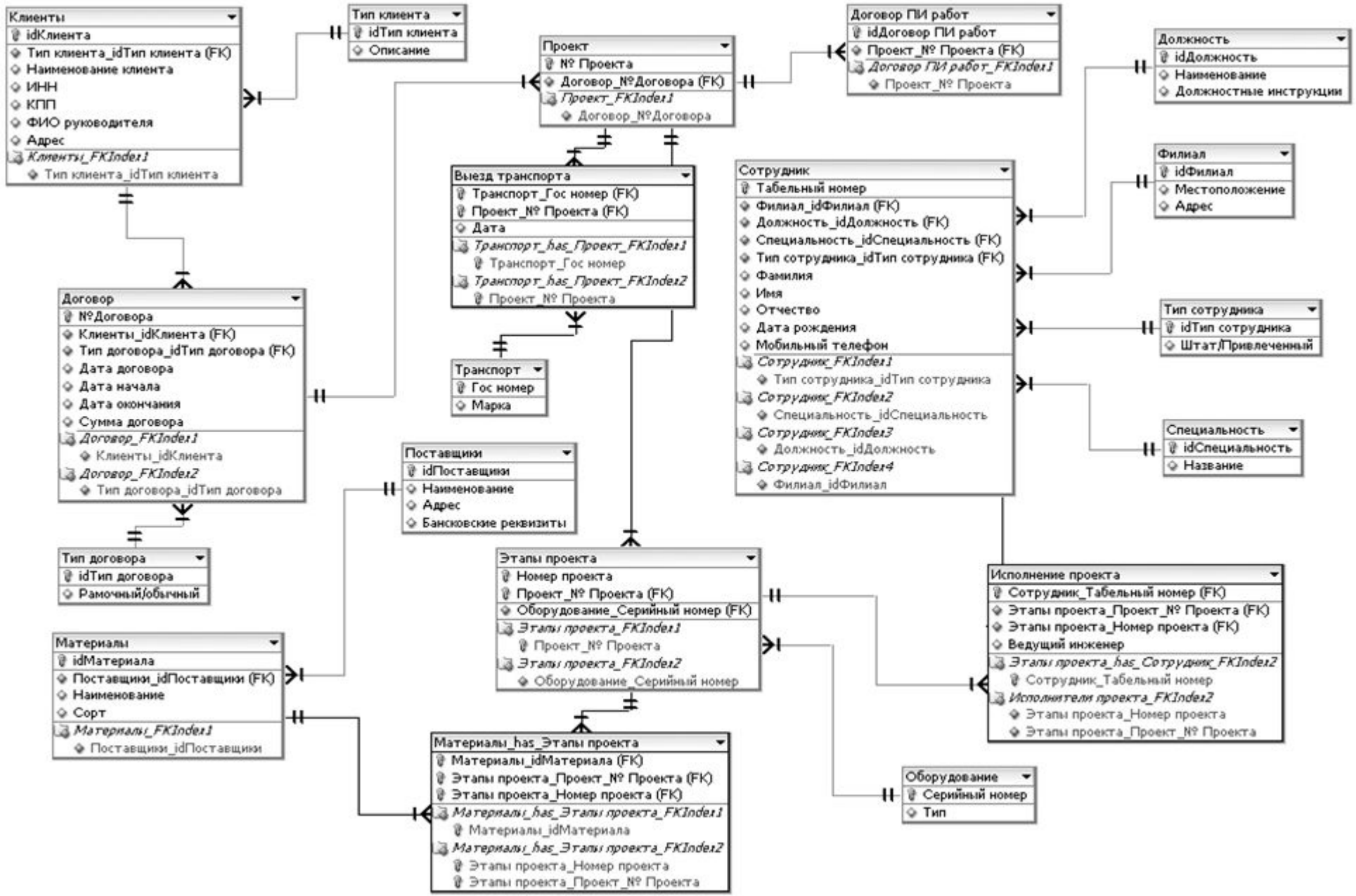


Детализирующая диаграмма бизнес-процесса (DFD (Data Flow Diagrams, диаграммы потоков данных))

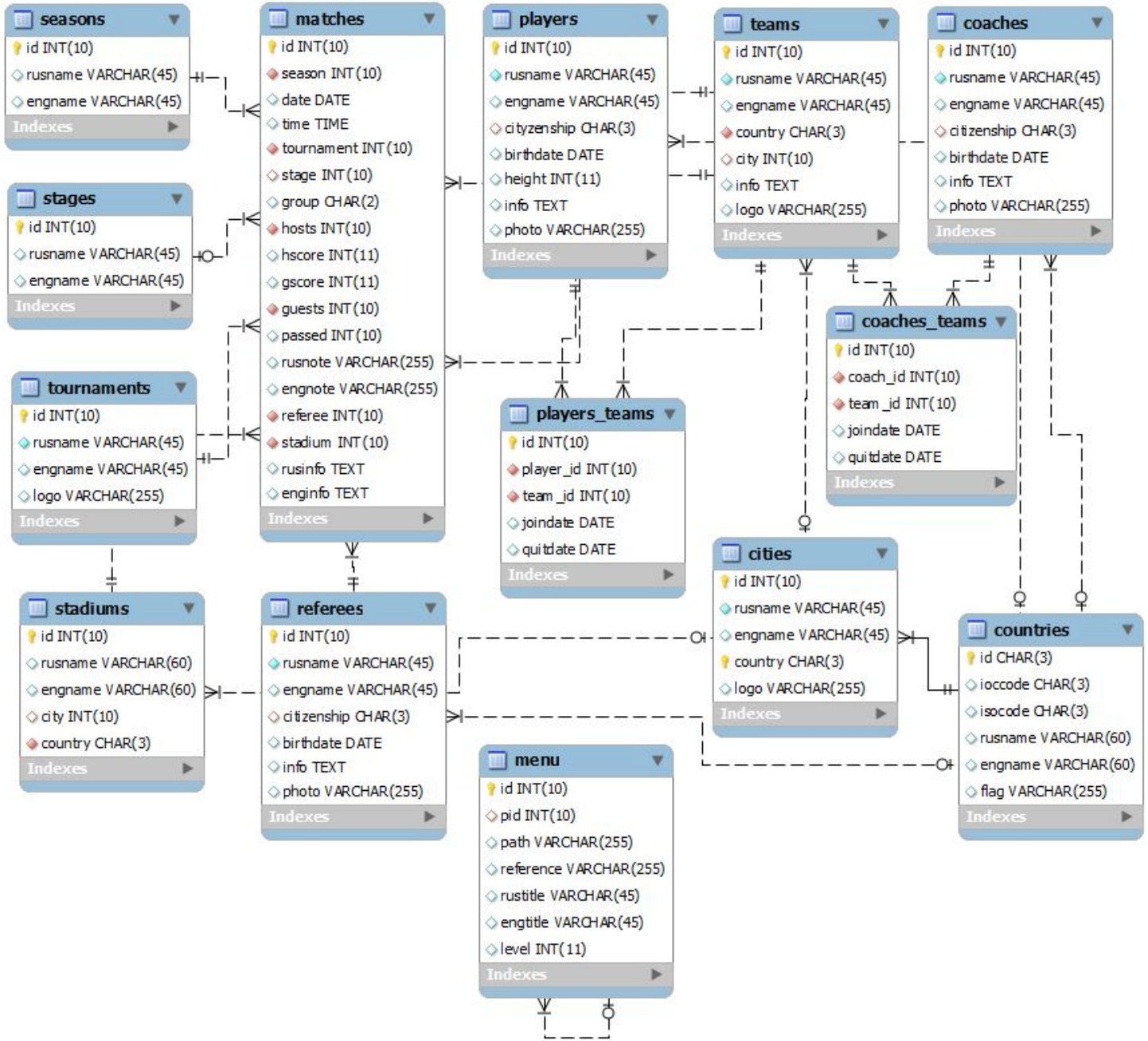
ERD (Entity-Relationship Diagrams,
диаграммы "сущность-связь")

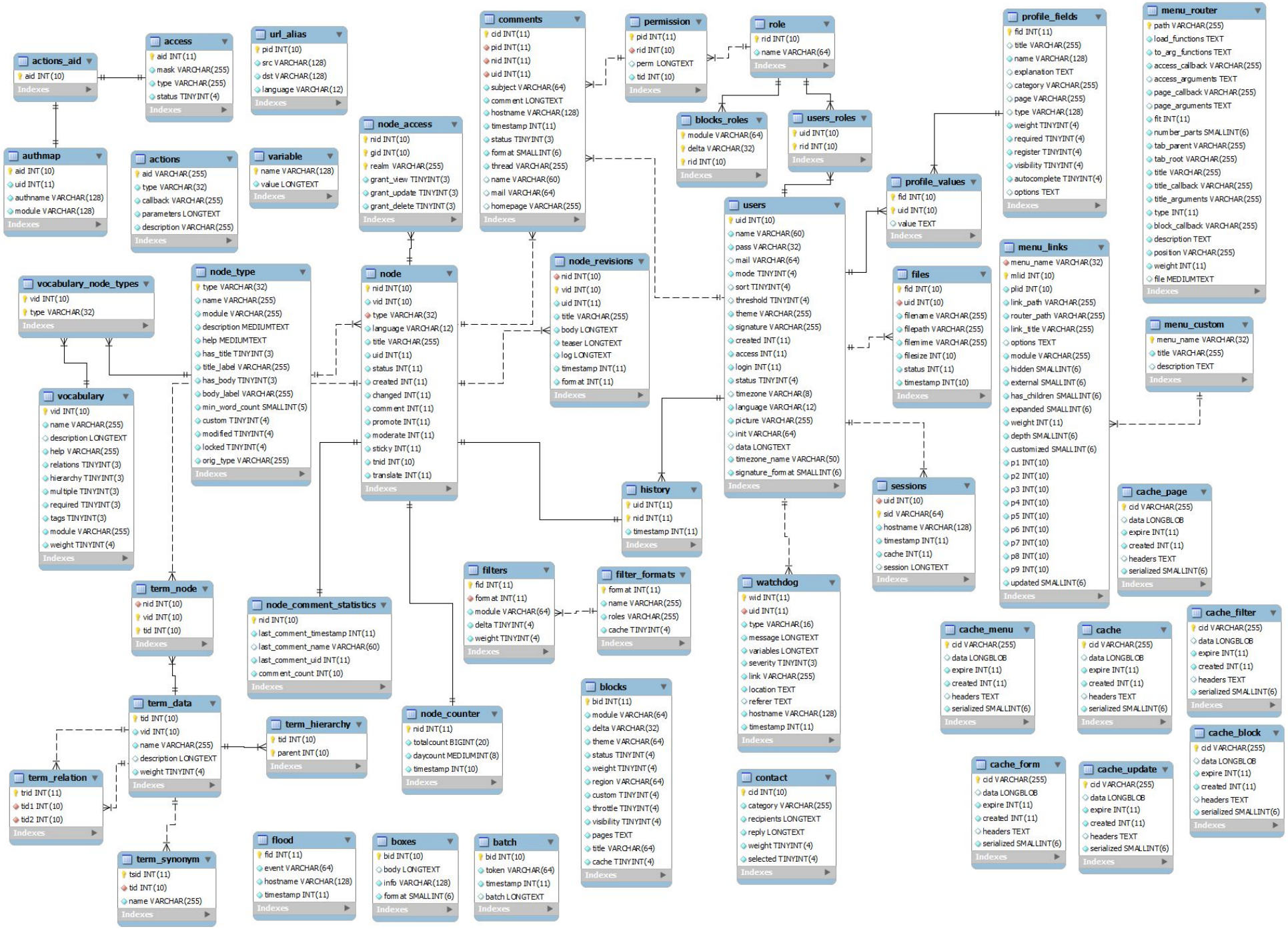


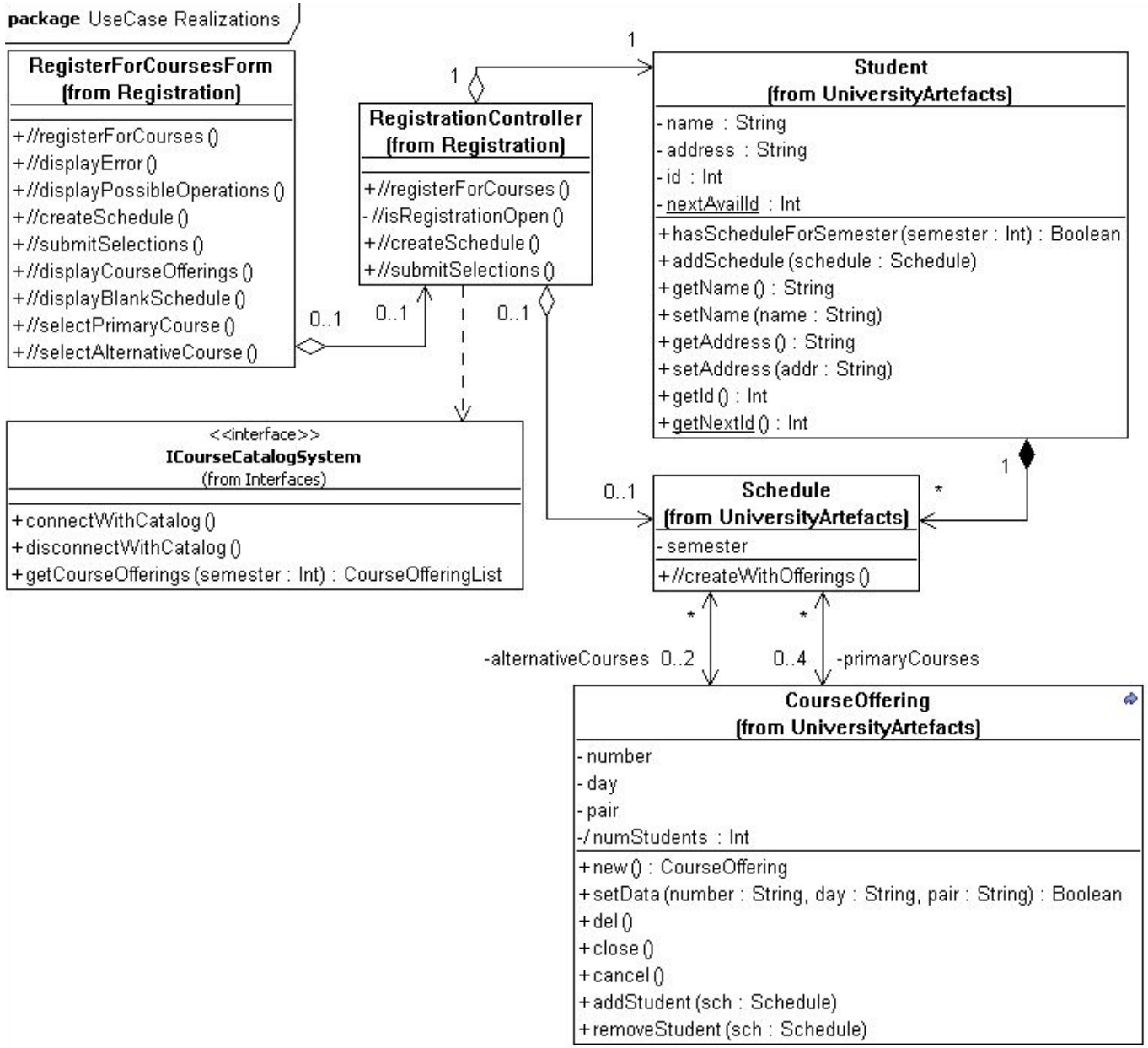

Улица
Код улицы: Поле ПК
Название улицы: String
Неисправность
Код неисправности: Поле ПК
Наименование неисправности: String
Исполнитель ремонтной заявки
Код исполнителя: Поле ПК
Фамилия И О: String
Абонент
Лицевой счет: String
Код улицы: Поле ПК (FK)
Номер дома: Number
Номер квартиры: Number
Фамилия И О: String
Телефон: String
Ремонтная заявка
Код заявки: Поле ПК
Лицевой счет: String (FK)
Код исполнителя: Поле ПК (FK)
Код неисправности: Поле ПК (FK)
Дата поступления: Datetime
Дата выполнения: Datetime
Погашена: Логический
Оплаченная сумма
Идентификатор факта оплаты: Поле ПК
Лицевой счет: String (FK)
Код услуги газоснабжения: Поле ПК (FK)
Сумма оплаты: Деньги
Дата оплаты: Datetime
Оплачиваемый месяц: Месяц
Оплачиваемый год: Год
Начисленная сумма
Идентификатор факта начисления: Поле ПК
Лицевой счет: String (FK)
Код услуги газоснабжения: Поле ПК (FK)
Сумма начисления: Деньги
Начисление за месяц: Месяц
Начисление за год: Год
Предоставляемая услуга газоснабжения
Код услуги газоснабжения: Поле ПК
Название услуги газоснабжения: String

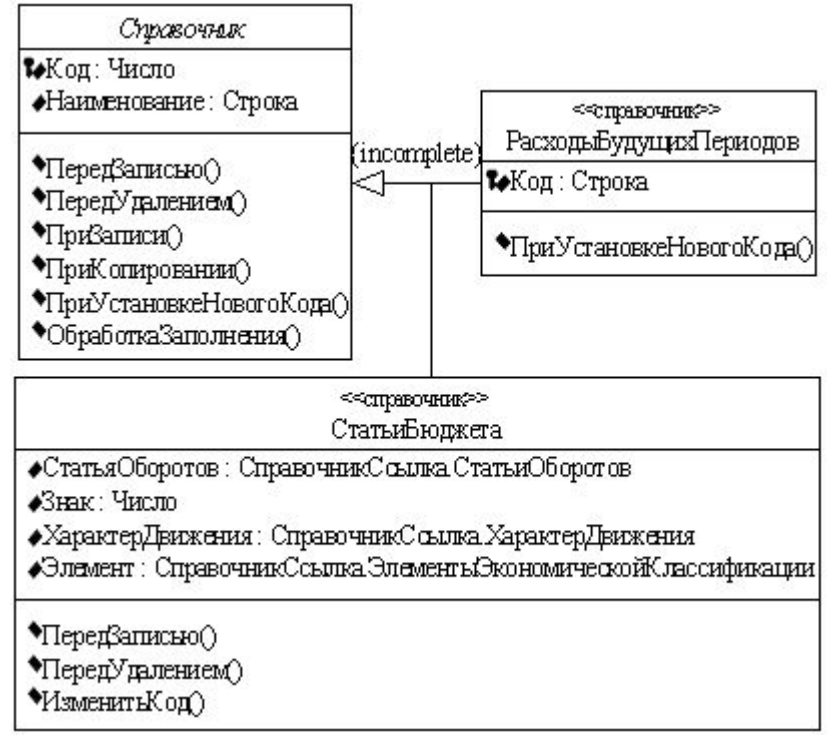

Клиенты
idКлиента
Тип клиента_idТип клиента (FK)
Наименование клиента
ИНН
КПП
ФИО руководителя
Адрес
Клиенты_FKIndex1
Тип клиента_idТип клиента
Тип клиента
idТип клиента
Описание
Проект
№ Проекта
Договор_№Договора (FK)
Проект_FKIndex1
Договор_№Договора
Договор ПИ работ
idДоговор ПИ работ
Проект_№ Проекта (FK)
Договор ПИ работ_FKIndex1
Проект_№ Проекта
Должность
idДолжность
Наименование
Должностные инструкции
Филиал
idФилиал
Местоположение
Адрес
Выезд транспорта
Транспорт_Гос номер (FK)
Проект_№ Проекта (FK)
Дата
Транспорт_has_Проект_FKIndex1
Транспорт_Гос номер
Транспорт_has_Проект_FKIndex2
Проект_№ Проекта
Транспорт
Гос номер
Марка
Сотрудник
Табельный номер
Филиал_idФилиал (FK)
Должность_idДолжность (FK)
Специальность_idСпециальность (FK)
Тип сотрудника_idТип сотрудника (FK)
Фамилия
Имя
Отчество
Дата рождения
Мобильный телефон
Сотрудник_FKIndex1
Тип сотрудника_idТип сотрудника
Сотрудник_FKIndex2
Специальность_idСпециальность
Сотрудник_FKIndex3
Должность_idДолжность
Сотрудник_FKIndex4
Филиал_idФилиал
Тип сотрудника
idТип сотрудника
Штат/Привлеченный
Специальность
idСпециальность
Название
Договор
№Договора
Клиенты_idКлиента (FK)
Тип договора_idТип договора (FK)
Дата договора
Дата начала
Дата окончания
Сумма договора
Договор_FKIndex1
Клиенты_idКлиента
Договор_FKIndex2
Тип договора_idТип договора
Тип договора
idТип договора
Рамочный/обычный
Поставщики
idПоставщики
Наименование
Адрес
Банковские реквизиты
Этапы проекта
Номер проекта
Проект_№ Проекта (FK)
Оборудование_Серийный номер (FK)
Этапы проекта_FKIndex1
Проект_№ Проекта
Этапы проекта_FKIndex2
Оборудование_Серийный номер
Исполнение проекта
Сотрудник_Табельный номер (FK)
Этапы проекта_Проект_№ Проекта (FK)
Этапы проекта_Номер проекта (FK)
Ведущий инженер
Этапы проекта_has_Сотрудник_FKIndex2
Сотрудник_Табельный номер
Исполнители проекта_FKIndex2
Этапы проекта_Номер проекта
Этапы проекта_Проект_№ Проекта
Материалы
idМатериала
Поставщики_idПоставщики (FK)
Наименование
Сорт
Материалы_FKIndex1
Поставщики_idПоставщики
Материалы_has_Этапы проекта
Материалы_idМатериала (FK)
Этапы проекта_Проект_№ Проекта (FK)
Этапы проекта_Номер проекта (FK)
Материалы_has_Этапы проекта_FKIndex1
Материалы_idМатериала
Материалы_has_Этапы проекта_FKIndex2
Этапы проекта_Номер проекта
Этапы проекта_Проект_№ Проекта
Оборудование
Серийный номер
Тип

Создание футбольного сайта




actions_aid
aid INT(10)
Indexes
access
aid INT(11)
mask VARCHAR(255)
type VARCHAR(255)
status TINYINT(4)
Indexes
url_alias
pid INT(10)
src VARCHAR(128)
dst VARCHAR(128)
language VARCHAR(12)
Indexes
comments
cid INT(11)
pid INT(11)
nid INT(11)
uid INT(11)
subject VARCHAR(64)
comment LONGTEXT
hostname VARCHAR(128)
timestamp INT(11)
status TINYINT(3)
format SMALLINT(6)
thread VARCHAR(255)
name VARCHAR(60)
mail VARCHAR(64)
homepage VARCHAR(255)
Indexes
permission
pid INT(11)
rid INT(10)
perm LONGTEXT
tid INT(10)
Indexes
role
rid INT(10)
name VARCHAR(64)
Indexes
profile_fields
fid INT(11)
title VARCHAR(255)
name VARCHAR(128)
explanation TEXT
category VARCHAR(255)
page VARCHAR(255)
type VARCHAR(128)
weight TINYINT(4)
required TINYINT(4)
register TINYINT(4)
visibility TINYINT(4)
autocomplete TINYINT(4)
options TEXT
Indexes
menu_router
path VARCHAR(255)
load_functions TEXT
to_arg_functions TEXT
access_callback VARCHAR(255)
access_arguments TEXT
page_callback VARCHAR(255)
page_arguments TEXT
fit INT(11)
number_parts SMALLINT(6)
tab_parent VARCHAR(255)
tab_root VARCHAR(255)
title VARCHAR(255)
title_callback VARCHAR(255)
title_arguments VARCHAR(255)
type INT(11)
block_callback VARCHAR(255)
description TEXT
position VARCHAR(255)
weight INT(11)
file MEDIUMTEXT
Indexes
authmap
aid INT(10)
uid INT(11)
authname VARCHAR(128)
module VARCHAR(128)
Indexes
actions
aid VARCHAR(255)
type VARCHAR(32)
callback VARCHAR(255)
parameters LONGTEXT
description VARCHAR(255)
Indexes
variable
name VARCHAR(128)
value LONGTEXT
Indexes
node_access
nid INT(10)
gid INT(10)
realm VARCHAR(255)
grant_view TINYINT(3)
grant_update TINYINT(3)
grant_delete TINYINT(3)
Indexes
blocks_roles
module VARCHAR(64)
delta VARCHAR(32)
rid INT(10)
Indexes
users_roles
uid INT(10)
rid INT(10)
Indexes
profile_values
fid INT(10)
uid INT(10)
value TEXT
Indexes
users
uid INT(10)
name VARCHAR(60)
pass VARCHAR(32)
mail VARCHAR(64)
mode TINYINT(4)
sort TINYINT(4)
threshold TINYINT(4)
theme VARCHAR(255)
signature VARCHAR(255)
created INT(11)
access INT(11)
login INT(11)
status TINYINT(4)
timezone VARCHAR(8)
language VARCHAR(12)
picture VARCHAR(255)
init VARCHAR(64)
data LONGTEXT
timezone_name VARCHAR(50)
signature_format SMALLINT(6)
Indexes
menu_links
menu_name VARCHAR(32)
mlid INT(10)
plid INT(10)
link_path VARCHAR(255)
router_path VARCHAR(255)
link_title VARCHAR(255)
options TEXT
module VARCHAR(255)
hidden SMALLINT(6)
external SMALLINT(6)
has_children SMALLINT(6)
expanded SMALLINT(6)
weight INT(11)
depth SMALLINT(6)
customized SMALLINT(6)
p1 INT(10)
p2 INT(10)
p3 INT(10)
p4 INT(10)
p5 INT(10)
p6 INT(10)
p7 INT(10)
p8 INT(10)
p9 INT(10)
updated SMALLINT(6)
Indexes
menu_custom
menu_name VARCHAR(32)
title VARCHAR(255)
description TEXT
Indexes
vocabulary_node_types
vid INT(10)
type VARCHAR(32)
Indexes
node_type
type VARCHAR(32)
name VARCHAR(255)
module VARCHAR(255)
description MEDIUMTEXT
help MEDIUMTEXT
has_title TINYINT(3)
title_label VARCHAR(255)
has_body TINYINT(3)
body_label VARCHAR(255)
min_word_count SMALLINT(5)
custom TINYINT(4)
modified TINYINT(4)
locked TINYINT(4)
orig_type VARCHAR(255)
Indexes
node
nid INT(10)
vid INT(10)
type VARCHAR(32)
language VARCHAR(12)
title VARCHAR(255)
uid INT(11)
status INT(11)
created INT(11)
changed INT(11)
comment INT(11)
promote INT(11)
moderate INT(11)
sticky INT(11)
tnid INT(10)
translate INT(11)
Indexes
node_revisions
nid INT(10)
vid INT(10)
uid INT(11)
title VARCHAR(255)
body LONGTEXT
teaser LONGTEXT
log LONGTEXT
timestamp INT(11)
format INT(11)
Indexes
files
fid INT(10)
uid INT(10)
filename VARCHAR(255)
filepath VARCHAR(255)
filemime VARCHAR(255)
filesize INT(10)
status INT(11)
timestamp INT(10)
Indexes
vocabulary
vid INT(10)
name VARCHAR(255)
description LONGTEXT
help VARCHAR(255)
relations TINYINT(3)
hierarchy TINYINT(3)
multiple TINYINT(3)
required TINYINT(3)
tags TINYINT(3)
module VARCHAR(255)
weight TINYINT(4)
Indexes
history
uid INT(11)
nid INT(11)
timestamp INT(11)
Indexes
sessions
uid INT(10)
sid VARCHAR(64)
hostname VARCHAR(128)
timestamp INT(11)
cache INT(11)
session LONGTEXT
Indexes
cache_page
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
term_node
nid INT(10)
vid INT(10)
tid INT(10)
Indexes
node_comment_statistics
nid INT(10)
last_comment_timestamp INT(11)
last_comment_name VARCHAR(60)
last_comment_uid INT(11)
comment_count INT(10)
Indexes
filters
fid INT(11)
format INT(11)
module VARCHAR(64)
delta TINYINT(4)
weight TINYINT(4)
Indexes
filter_formats
format INT(11)
name VARCHAR(255)
roles VARCHAR(255)
cache TINYINT(4)
Indexes
watchdog
wid INT(11)
uid INT(11)
type VARCHAR(16)
message LONGTEXT
variables LONGTEXT
severity TINYINT(3)
link VARCHAR(255)
location TEXT
referer TEXT
hostname VARCHAR(128)
timestamp INT(11)
Indexes
cache_menu
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
cache
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
cache_filter
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
term_data
tid INT(10)
vid INT(10)
name VARCHAR(255)
description LONGTEXT
weight TINYINT(4)
Indexes
term_hierarchy
tid INT(10)
parent INT(10)
Indexes
node_counter
nid INT(11)
totalcount BIGINT(20)
daycount MEDIUMINT(8)
timestamp INT(10)
Indexes
blocks
bid INT(11)
module VARCHAR(64)
delta VARCHAR(32)
theme VARCHAR(64)
status TINYINT(4)
weight TINYINT(4)
region VARCHAR(64)
custom TINYINT(4)
throttle TINYINT(4)
visibility TINYINT(4)
pages TEXT
title VARCHAR(64)
cache TINYINT(4)
Indexes
contact
cid INT(10)
category VARCHAR(255)
recipients LONGTEXT
reply LONGTEXT
weight TINYINT(4)
selected TINYINT(4)
Indexes
cache_block
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
cache_form
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
cache_update
cid VARCHAR(255)
data LONGBLOB
expire INT(11)
created INT(11)
headers TEXT
serialized SMALLINT(6)
Indexes
term_relation
trid INT(11)
tid1 INT(10)
tid2 INT(10)
Indexes
term_synonym
tsid INT(11)
tid INT(10)
name VARCHAR(255)
Indexes
flood
fid INT(11)
event VARCHAR(64)
hostname VARCHAR(128)
timestamp INT(11)
Indexes
boxes
bid INT(10)
body LONGTEXT
info VARCHAR(128)
format SMALLINT(6)
Indexes
batch
bid INT(10)
token VARCHAR(64)
timestamp INT(11)
batch LONGTEXT
Indexes





ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД К АНАЛИЗУ И ПРОЕКТИРОВАНИЮ ИНФОРМАЦИОННЫХ СИСТЕМ