ПЕРМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ЭКОНОМИЧЕСКИЙ ФАКУЛЬТЕТ

Дисциплина «Информационные технологии в экономике»

Лекция-27
Автоматизированное проектирование информационных систем (CASE-технология)

Пермь, 2020

Изучаемые вопросы

1. Основные понятия CASE-технологий. Классификация CASE-технологий. Архитектура CASE-средства.
2. Функционально-ориентированное проектирование (диаграммы: функциональных спецификации, потоков данных, переходов состояний, инфологических моделей «сущность-связь», структуры программного приложения).
3. Объектно-ориентированное проектирование (диаграммы: прецедентов использования, классов объектов, состояний, взаимодействия объектов, деятельностей, пакетов, компонентов, размещения). Прототипное проектирование (RAD-технология).

Основные понятия CASE-технологий

Термин **CASE** (**Computer Aided System/Software Engineering**) используется в довольно широком смысле.

- Первоначальное значение термина CASE было ограниченно вопросами автоматизации разработки только лишь программного обеспечения.
- CASE-технологии развивались с целью преодоления ограничений при использовании структурной методологии проектирования за счет ее автоматизации и интеграции поддерживающих средств. Основные сложности включают в себя:
  - сложность понимания,
  - высокую трудоемкость и стоимость использования,
  - трудность внесения изменений в проектные спецификации и т.д.
- CASE-технологии только обеспечивают более высокую эффективность методологий проектирования.
- Большинство существующих CASE-систем ориентировано на автоматизацию проектирования программного обеспечения.
- Наибольшая потребность в использовании CASE-систем испытывается на начальных этапах разработки, а именно на этапах анализа и спецификации требований к ИС.

Основные понятия CASE-технологий

Преимущества CASE-технологии по сравнению с традиционной технологией оригинального проектирования:

- Улучшение качества разрабатываемого программного приложения.
- Возможность повторного использования компонентов разработки.
- Поддержание адаптивности и сопровождения ИС.
- Снижение времени создания системы.
- Освобождение разработчиков от рутинной работы по документированию проекта, так как при этом используется встроенный документатор.
- Возможность коллективной разработки ЭИС в режиме реального времени.

Основные понятия CASE-технологий

CASE-технология в рамках методологии включает в себя методы, с помощью которых на основе графической нотации строятся диаграммы, поддерживаемые инструментальной средой.

Методология определяет шаги и этапность реализации проекта, а также правила использования методов, с помощью которых разрабатывается проект.

Метод - это процедура или техника генерации описаний компонентов ЭИС (например, проектирование потоков и структур данных).

Нотация - отображение структуры системы, элементов данных, этапов обработки с помощью специальных графических символов диаграмм, а также описание проекта системы на формальных и естественных языках.

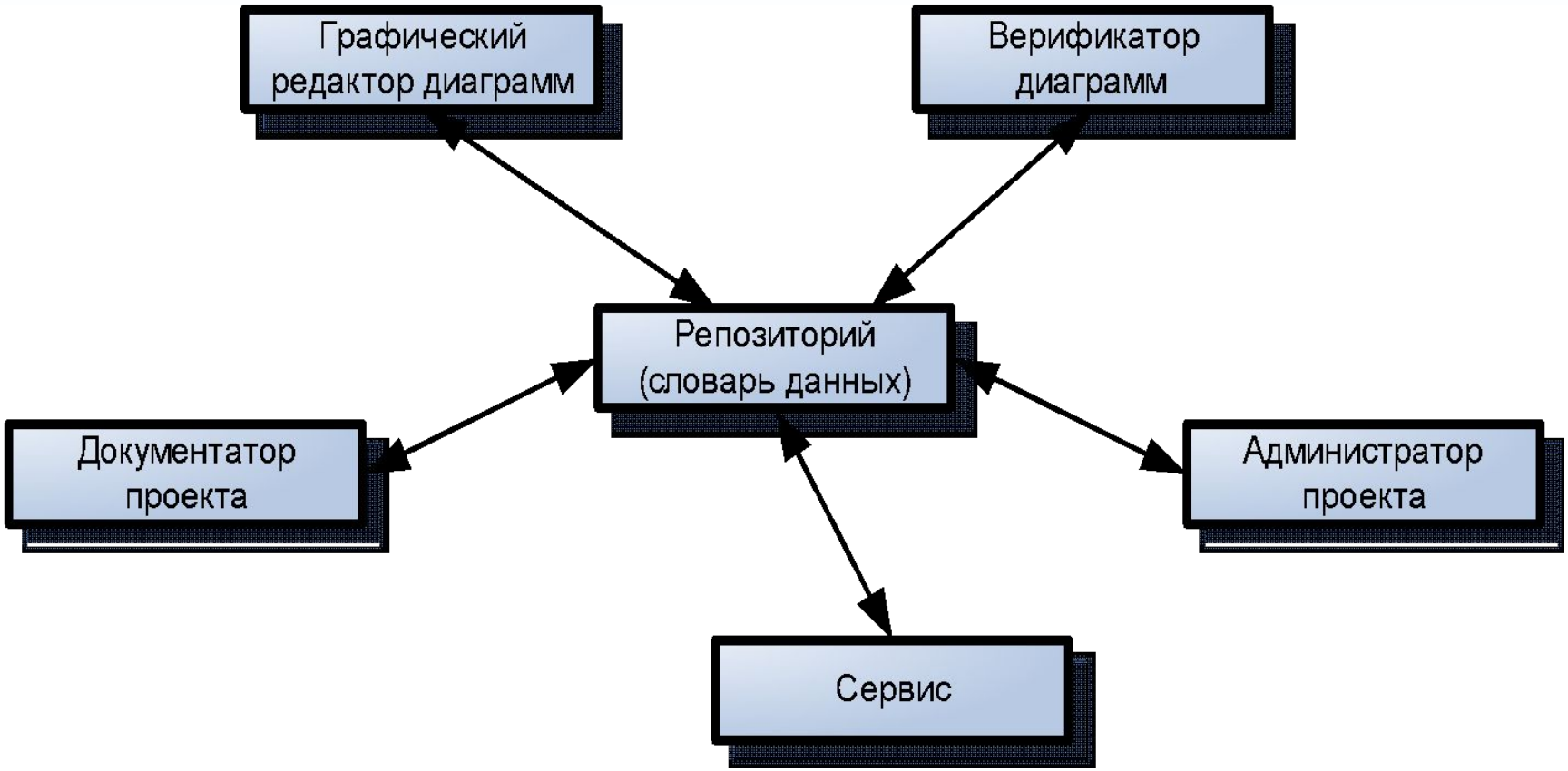
Инструментальные средства CASE - специальные программы, которые поддерживают одну или несколько методологий анализа и проектирования ИС.

Классификация CASE-технологий

Современные CASE-системы классифицируются по следующим признакам:

№	Признак	Описание
1	по поддерживаемым методологиям проектирования	▪ функционально (структурно)-ориентированные, ▪ объектно-ориентированные, ▪ комплексно-ориентированные (набор методологий проектирования).
2	по поддерживаемым графическим нотациям построения диаграмм	▪ с фиксированной нотацией, ▪ с отдельными нотациями, ▪ наиболее распространенными нотациями.
3	по степени интегрированности	▪ tools – отдельные локальные средства, ▪ toolkit – набор неинтегрированных средств, охватывающих большинство этапов разработки ЭИС), ▪ workbench – полностью интегрированные средства, связанные общей базой проектных данных – репозиторием.
4	по типу и архитектуре вычислительной техники	▪ ориентированные на ПЭВМ, ▪ ориентированные на локальную вычислительную сеть (ЛВС), ▪ ориентированные на глобальную вычислительную сеть (ГВС), ▪ смешанного типа.
5	по режиму коллективной разработки проекта	▪ не поддерживающие коллективную разработку, ▪ ориентированные на режим реального времени разработки проекта, ▪ ориентированные на режим объединения подпроектов.
6	по типу операционной системы (ОС)	▪ работающие под управлением MS Windows, ▪ работающие под управлением UNIX, ▪ работающие под управлением различных ОС (WINDOWS, UNIX, OS/2 и др.).

Архитектура CASE-средства



Архитектура CASE-средства

Ядром системы является база данных проекта - **репозиторий**.

Репозиторий содержит информацию об объектах проектируемой ЭИС и взаимосвязях между ними, все подсистемы обмениваются данными с ним.

В репозиторий хранятся описания следующих объектов:

- проектировщиков и их прав доступа к различным компонентам системы;
- организационных структур;
- диаграмм;
- компонентов диаграмм;
- связей между диаграммами;
- структур данных;
- программных модулей;
- процедур;
- библиотеки модулей и т.д.

Архитектура CASE-средства

Графические средства моделирования предметной области позволяют разработчикам автоматизированных ИС в наглядном виде изучать существующую информационную систему, перестраивать ее в соответствии с поставленными целями и имеющимися ограничениями.

Графический редактор диаграмм предназначен для отображения в графическом виде в заданной нотации проектируемой ИС.

Он позволяет выполнять следующие операции:

- создавать элементы диаграмм и взаимосвязи между ними;
- задавать описания элементов диаграмм;
- задавать описания связей между элементами диаграмм;
- редактировать элементы диаграмм, их взаимосвязи и описания.

Архитектура CASE-средства

Верификатор диаграмм служит для контроля правильности построения диаграмм в заданной методологии проектирования ИС.

Он выполняет следующие функции:

- мониторинг правильности построения диаграмм;
- диагностику и выдачу сообщений об ошибках;
- выделение на диаграмме ошибочных элементов.

Документатор проекта позволяет получать информацию о состоянии проекта в виде различных отчетов.

Администратор проекта представляет собой инструменты, необходимые для выполнения следующих административных функций:

- инициализации проекта;
- задания начальных параметров проекта;
- назначения и изменения прав доступа к элементам проекта;
- мониторинга выполнения проекта.

Сервис представляет собой набор системных утилит по обслуживанию репозитория.

Данные утилиты выполняют функции архивации данных, восстановления данных и создания нового репозитория.

Характеристики современного CASE-средства

№	Характеристика	Описание
1	Наличие базы проектных данных, архива или словаря	СУБД и словари данных обеспечивают высокую степень интеграции данных и предоставляют широкие возможности для централизованного сбора, хранения и распределения проектной информации между различными этапами проекта и выполняемыми операциями.
2	Интерфейсы с другими CASE-системами	В процессе проектирования ЭИС могут использоваться различные методологии, поэтому важно, чтобы используемые CASE-системы предоставляли возможности для аффективного использования нескольких методов. При этом должна быть обеспечена терминологическая совместимость различных методологий.
3	Возможности экспорта/импорта	Спецификации, полученные на этапах анализа, проектирования и кодирования для одной ЭИС, могут быть использованы для проектирования другой системы. Повторное проектирование и кодирование могут быть обеспечены при помощи средств экспорта/импорта спецификаций в различные CASE-системы.
4	Многопользовательский режим	Развитые CASE-системы должны обладать возможностями разделения полномочий персонала разработчиков и объединения отдельных работ в общий проект.
5	Открытая архитектура	Открытая к доступу проектировщиков информация об используемых форматах файлов и интерфейсах должна позволять безболезненно переходить от одной CASE-системы к другой.
6	Расширение новыми методологиями	Как и любое программное средство, CASE-сисгема должна обладать возможностью совершенствоваться с учетом появления новых требований или новых предметных областей.

Характеристики современного CASE-средства

№	Характеристика	Описание
7	Наличие графических средств поддержки проектирования	Большинство CASE-систем базируется на графическом отображении методологий. Графические элементы структурных диаграмм и объекты словаря должны позволять декомпозировать различные компоненты проекта и детализировать изображения с той степенью, с какой это необходимо для понимания проектных решений.
8	Обеспечение качества проектной документации	Возможности CASE-системы анализировать и проверять описания и документацию на полноту и непротиворечивость, а также на соответствие принятым в данной методологии стандартам и правилам. В результате анализа должна формироваться информация, указывающая на имеющиеся противоречия или неполноту проектной документации, находящейся в архиве или словаре.
9	Автоматическая генерация отчетов о проектных решениях	Решения (спецификации), созданные в процессе проектирования, служат источником документирования системы. Часто возникает потребность получения твердой копии спецификаций в текстовой или графической форме.
10	Генерация кодов программ	CASE-системы с жесткой ориентацией на конкретные СУБД должны обеспечивать возможность генерации программ в среде этих СУБД.
11	Планирование и управление проектом	Многие развитые CASE-системы имеют в своем составе средства планирования и управления проектом. Спецификации, которые используются этими средствами, представляют собой опорные точки управления, позволяющие определять сроки разработки.

Функционально-ориентированное проектирование

Основными идеями **функционально-ориентированной CASE-технологии** являются идеи структурного анализа и проектирования информационных систем.

Они заключаются в следующем:

1. Декомпозиция всей системы на некоторое множество иерархически подчиненных функций.
2. Представление всей информации в виде графической нотации. Систему всегда легче понять, если она изображена графически.

В качестве инструментальных средств структурного анализа и проектирования выступают следующие диаграммы:

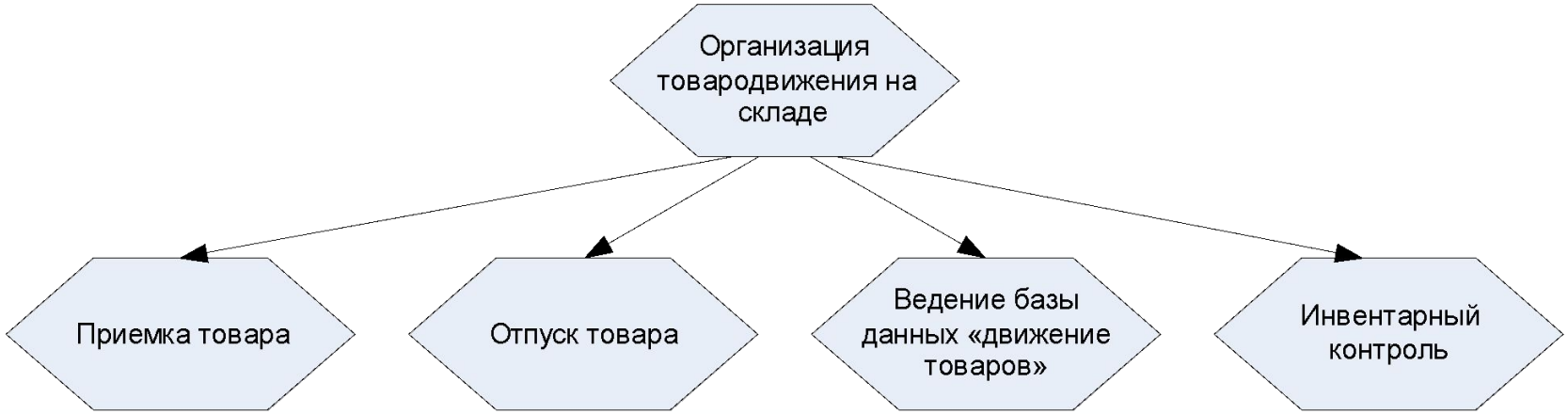
- BFD (Business Function Diagram) ;
- DFD (Data Flow Diagram);
- STD (State Transition Diagram);
- ERD (Entity Relationship Diagram);
- SSD (System Structure Diagram).

Функционально-ориентированное проектирование

Диаграммы функциональных спецификаций позволяют представить общую структуру ИС, отражающую взаимосвязь различных задач (процедур) в процессе получения требуемых результатов.

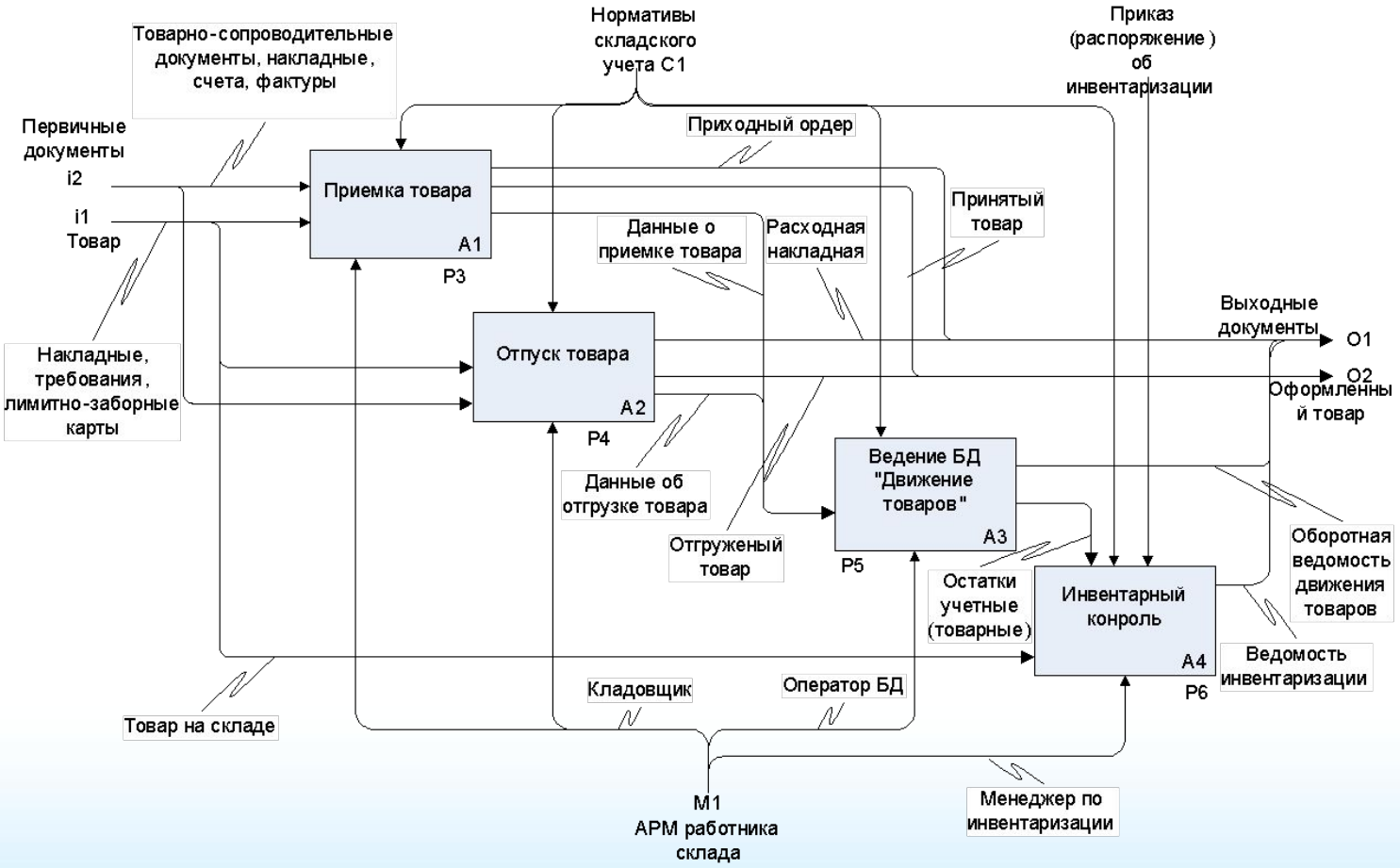
Основными объектами являются:

- **Функция** - некоторое действие информационной системы, необходимое для решения экономической задачи;
- **Декомпозиция функции** - разбиение функции на множество подфункций.



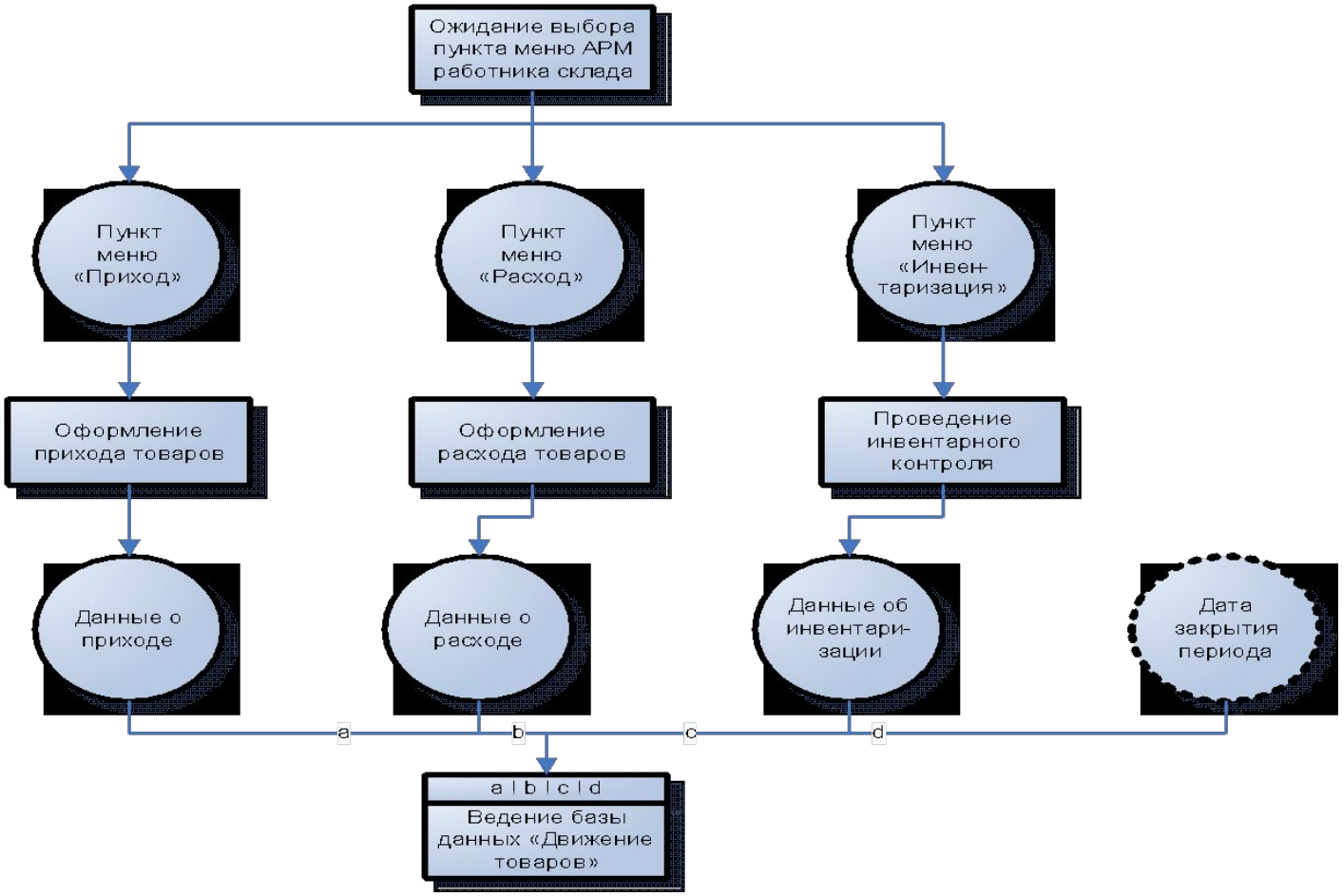
Функционально-ориентированное проектирование

Диаграммы потоков данных отражают передачу информации от одной функции к другой в рамках заданной технологии обработки.



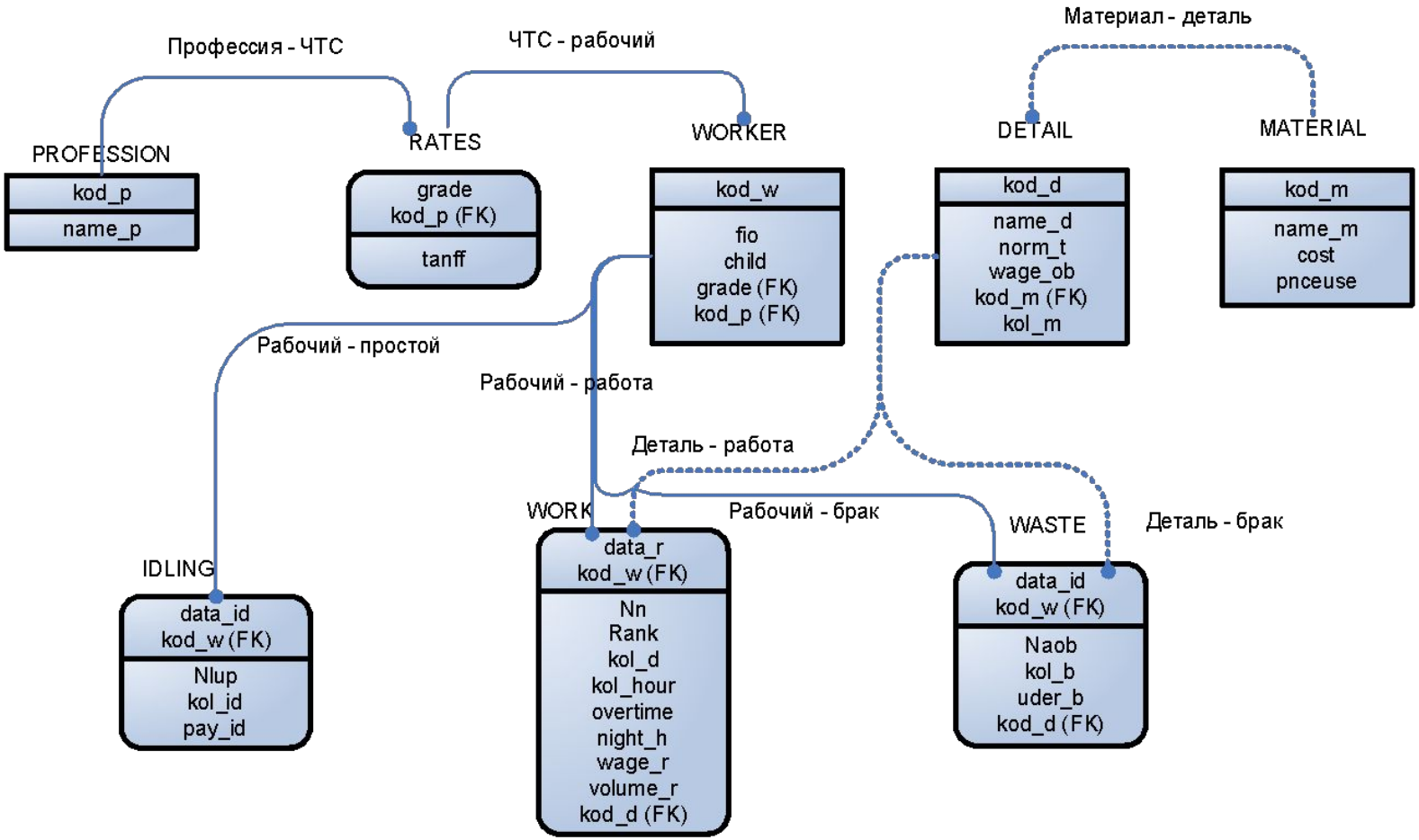
Функционально-ориентированное проектирование

Диаграммы переходов состояний моделируют поведение системы во времени в зависимости от происшедших событий (нажатая клавиша, дата отчетного периода и т.д.).



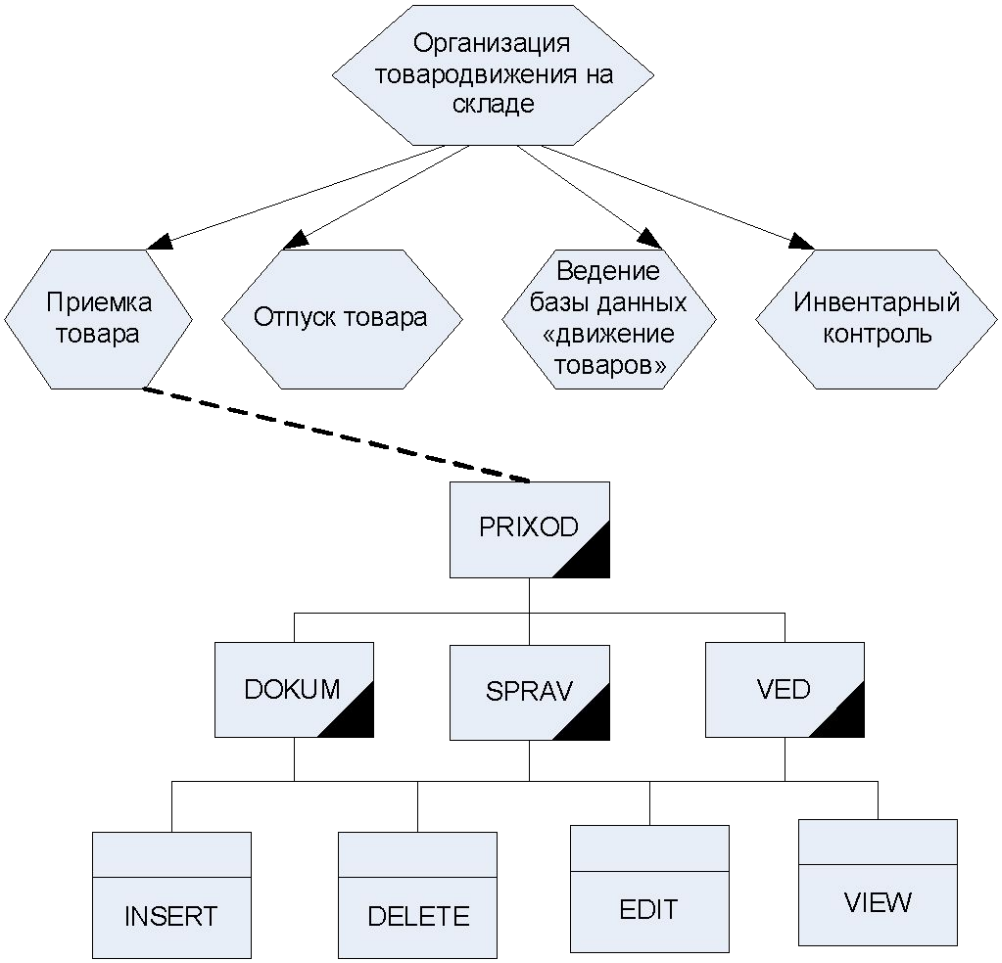
Функционально-ориентированное проектирование

Диаграммы инфологических моделей «сущность-связь» (ER-диаграммы) ориентированы на разработку базы данных, структура которой не зависит от конкретных информационных потребностей и позволяет выполнять любые запросы пользователей.



Функционально-ориентированное проектирование

Диаграмма структуры программного приложения задает взаимосвязь функций и программных модулей, которые их реализуют: меню, формы, отчеты и т.д.



Объектно-ориентированное проектирование

Структурная декомпозиция ИС на основе **объектно-ориентированного подхода** отличается от функционально-ориентированного подхода лучшей способностью отражать динамическое поведение системы в зависимости от возникающих событий.

Модель проблемной области - совокупность взаимодействующих во времени объектов.

Конкретный процесс обработки информации формируется в виде последовательности взаимодействий объектов.

Одна операция обработки данных может рассматриваться как результат одного взаимодействия объектов.

Результат: множество классов объектов с присоединенными методами обработки атрибутов.

Объектно-ориентированный подход предполагает совместное моделирование данных и процессов.

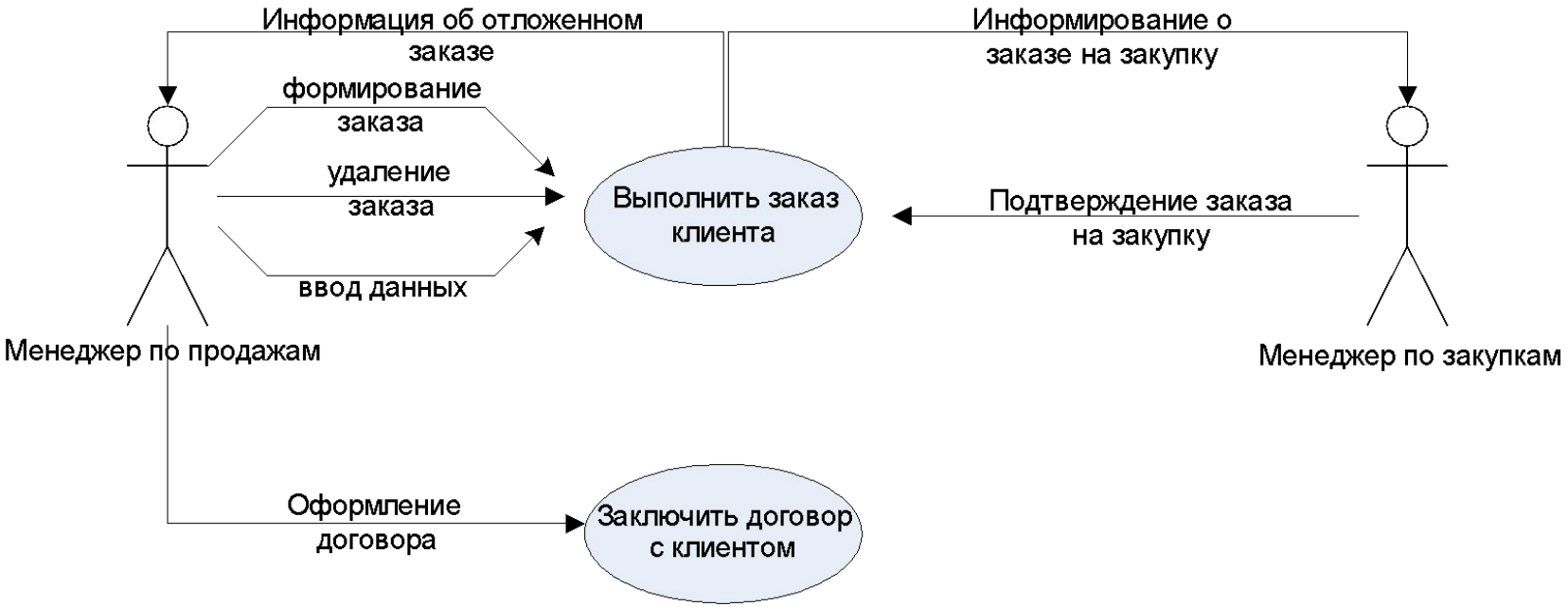
Объектно-ориентированное проектирование

Система объектно-ориентированных моделей в соответствии с нотациями UML включает в себя следующие диаграммы:

1. Диаграмму прецедентов использования (Use-case diagram).
2. Диаграмму классов объектов (Class diagram.
3. Диаграммы состояний (State chart diagram).
4. Диаграммы взаимодействия объектов (Interaction diagram).
5. Диаграммы деятельностей (Activity diagram).
6. Диаграммы пакетов (Package diagram).
7. Диаграмму компонентов (Component diagram.
8. Диаграмму размещения (Deployment diagram).

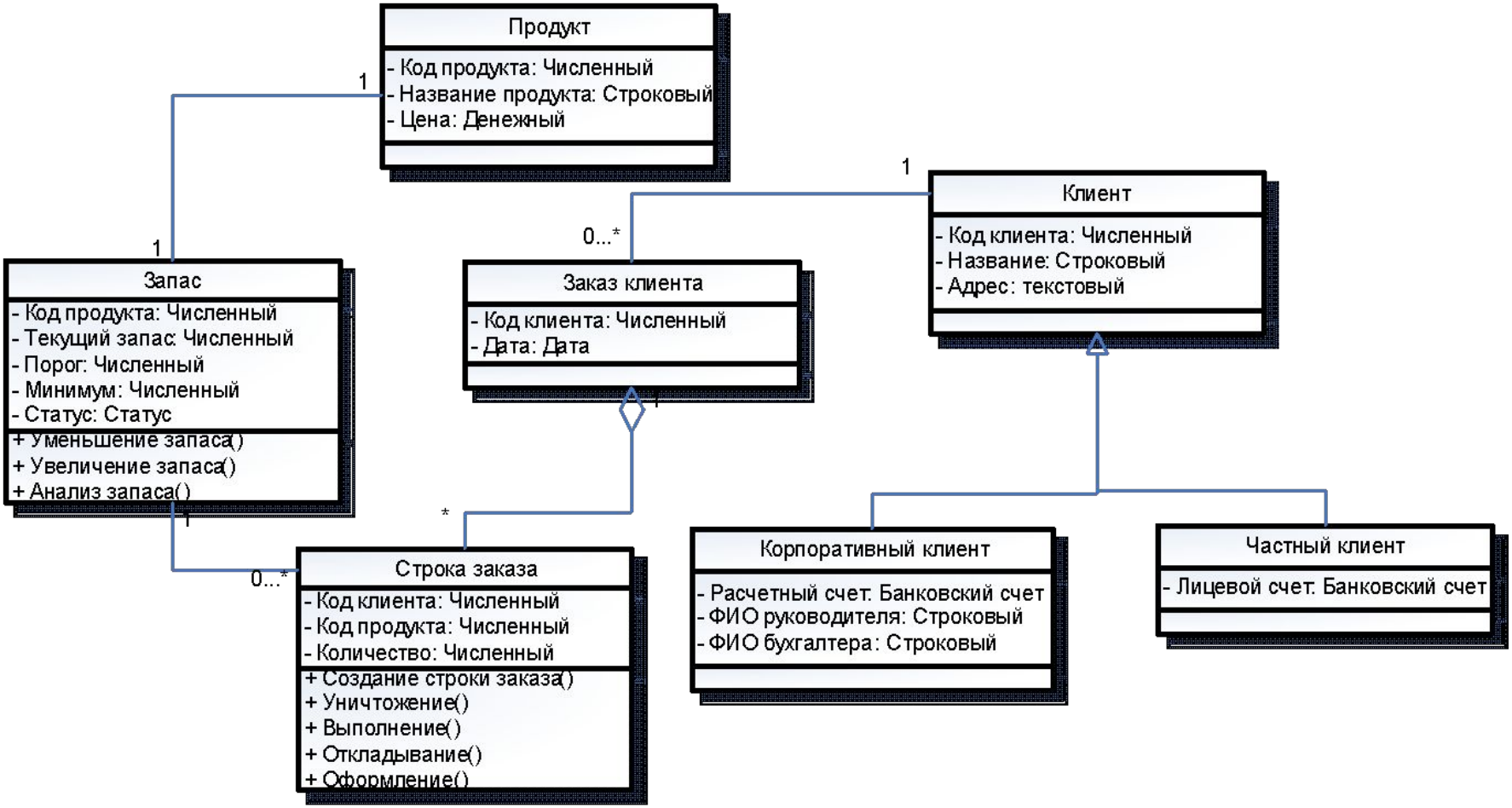
Объектно-ориентированное проектирование

Диаграмма прецедентов использования выявляет основные бизнес-процессы как последовательности транзакций, которые должны выполняться целиком, когда выполнение обособленного подмножества действий не имеет значения без выполнения всей последовательности.



Объектно-ориентированное проектирование

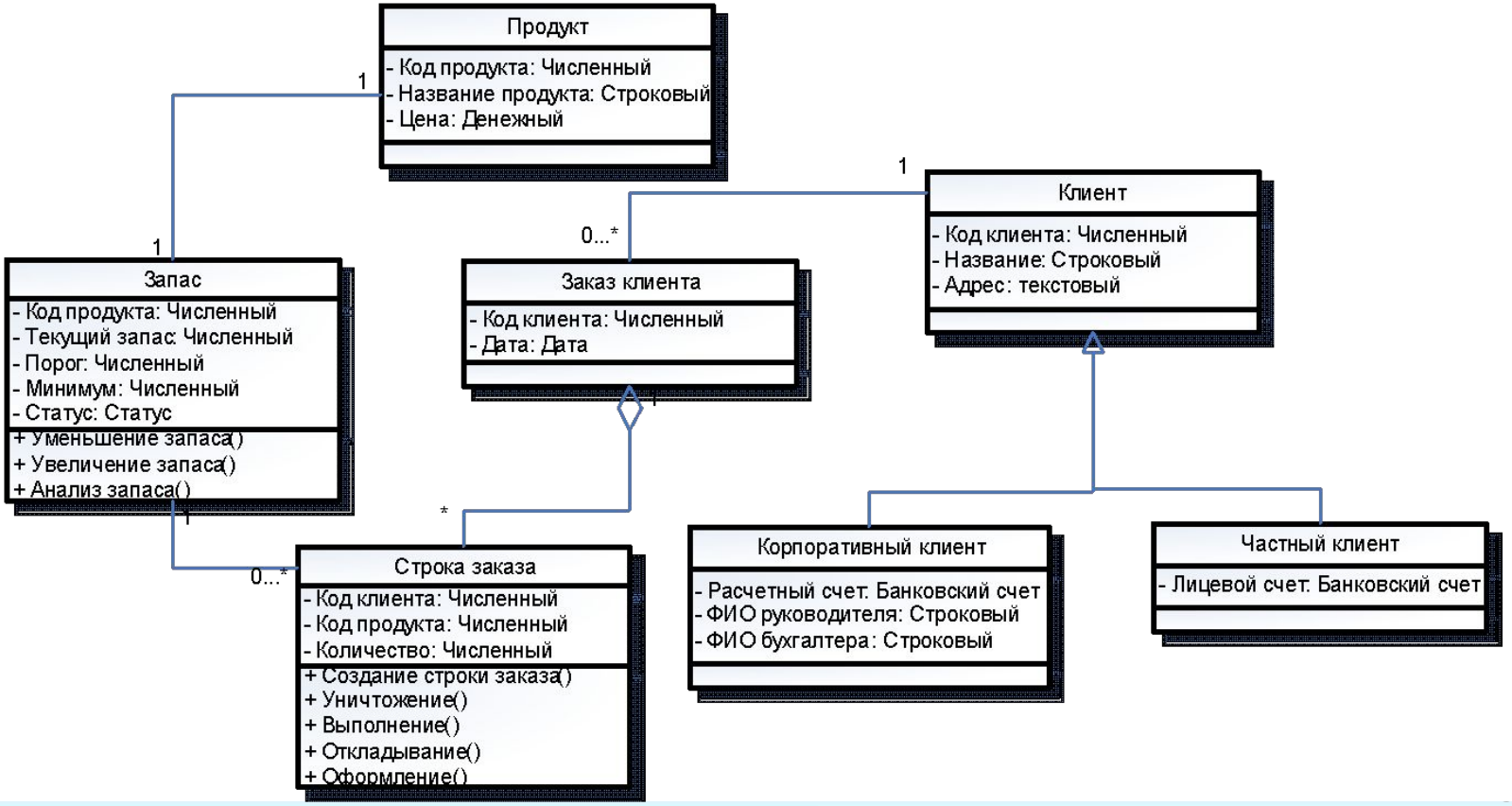
Диаграммы классов объектов отображают статическую структуру классов объектов. Эта диаграмма рассматривает внутреннюю структуру проблемной области, иерархию классов объектов, статические связи объектов.



Объектно-ориентированное проектирование

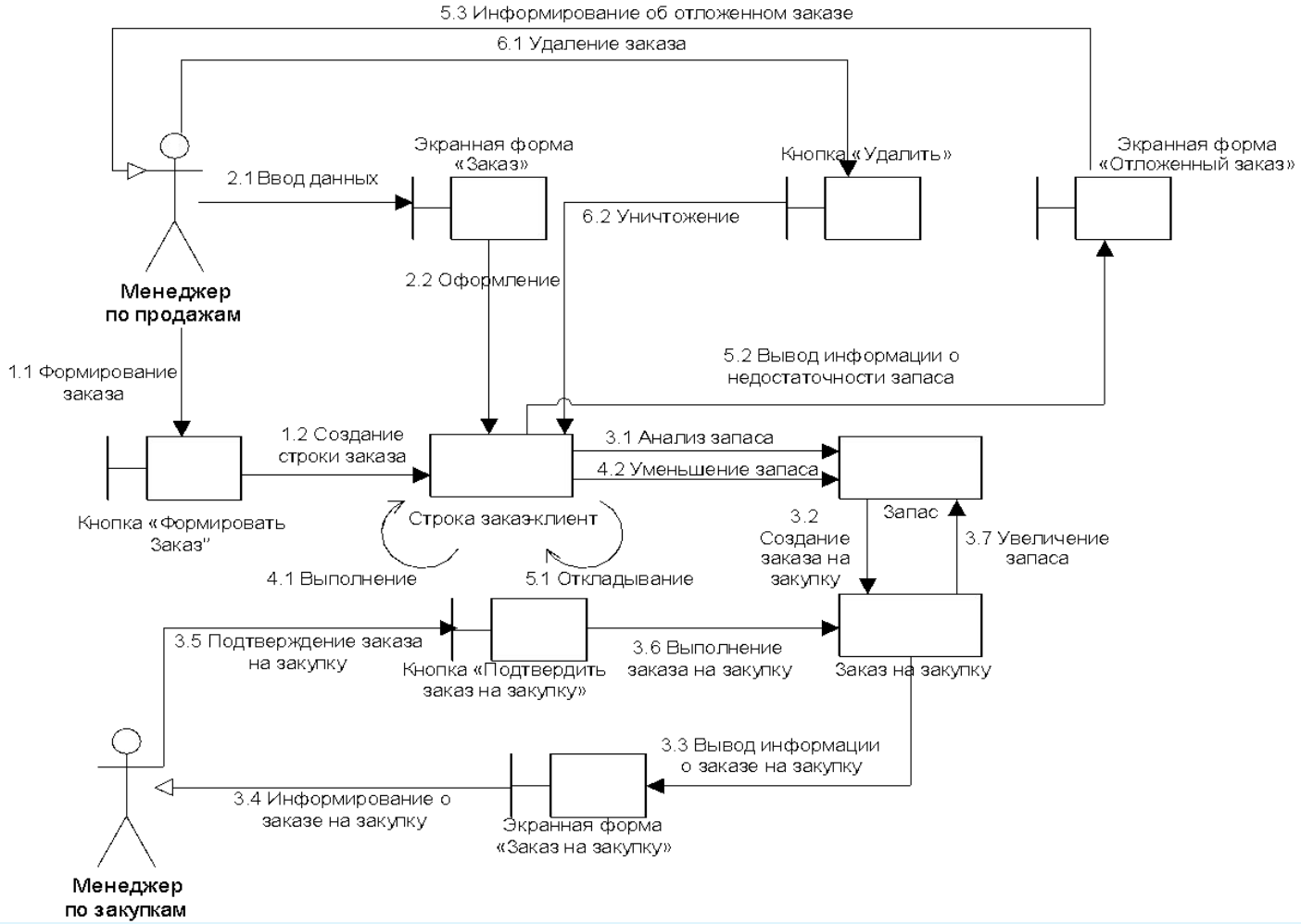
Диаграмма состояний отображает поведение объектов одного класса в динамике, связь состояний объектов с событиями и определяет:

- какие типичные состояния проходит объект;
- какие события ведут к изменению состояния объекта;
- какие действия объект выполняет, когда он получает сообщение об изменении состояния;
- как объекты создаются и уничтожаются (входные и выходные точки диаграммы).



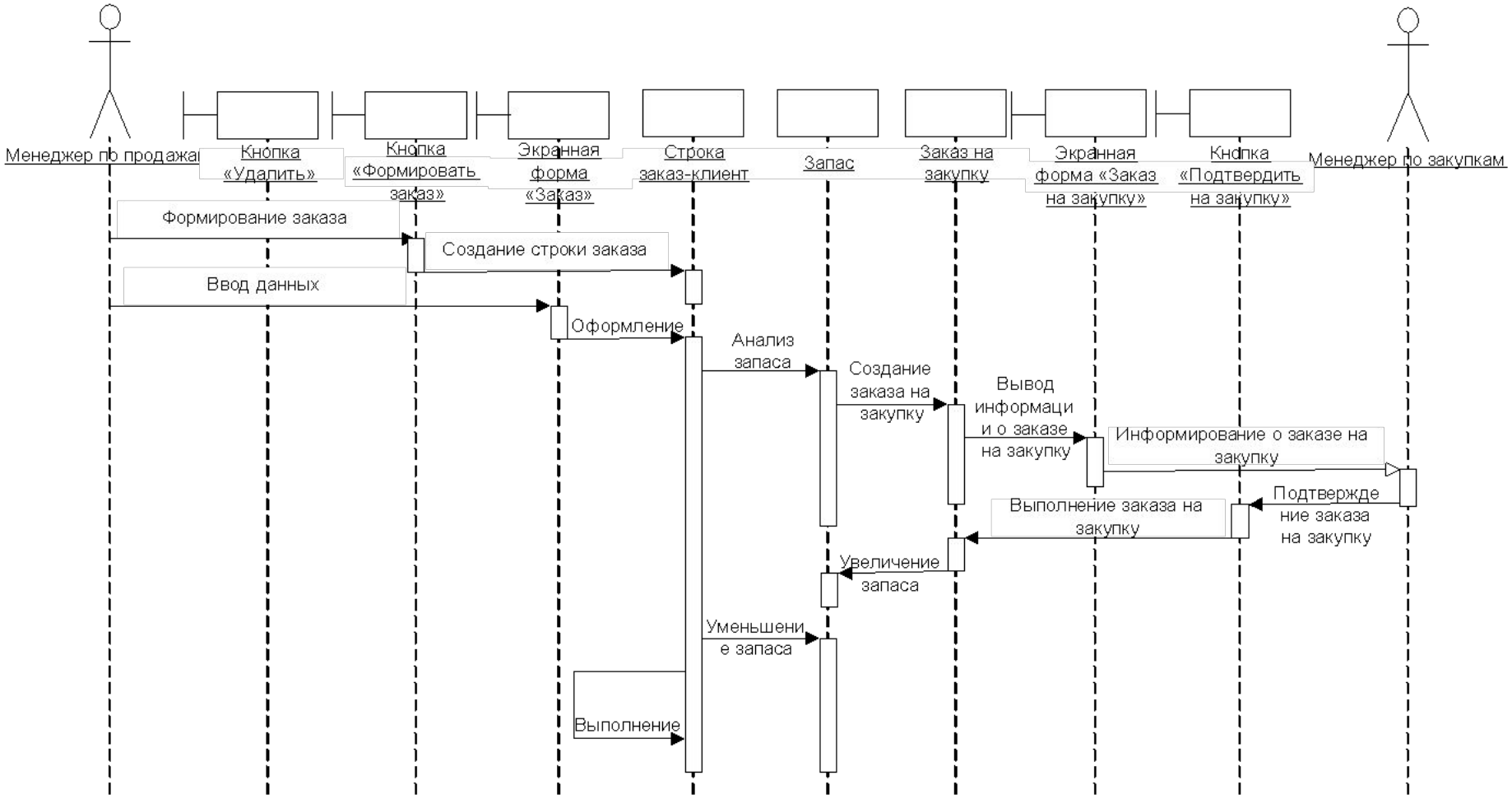
Объектно-ориентированное проектирование

В **диаграмме взаимодействия объектов** отображается последовательность взаимодействий между объектами, которая отображается в виде стрелки между объектами, которая соответствует событию или сообщению от одного объекта к другому, вызывающему выполнение метода, реагирующего на событие (сообщение) объекта. Номер стрелки соответствует номеру события в последовательности.



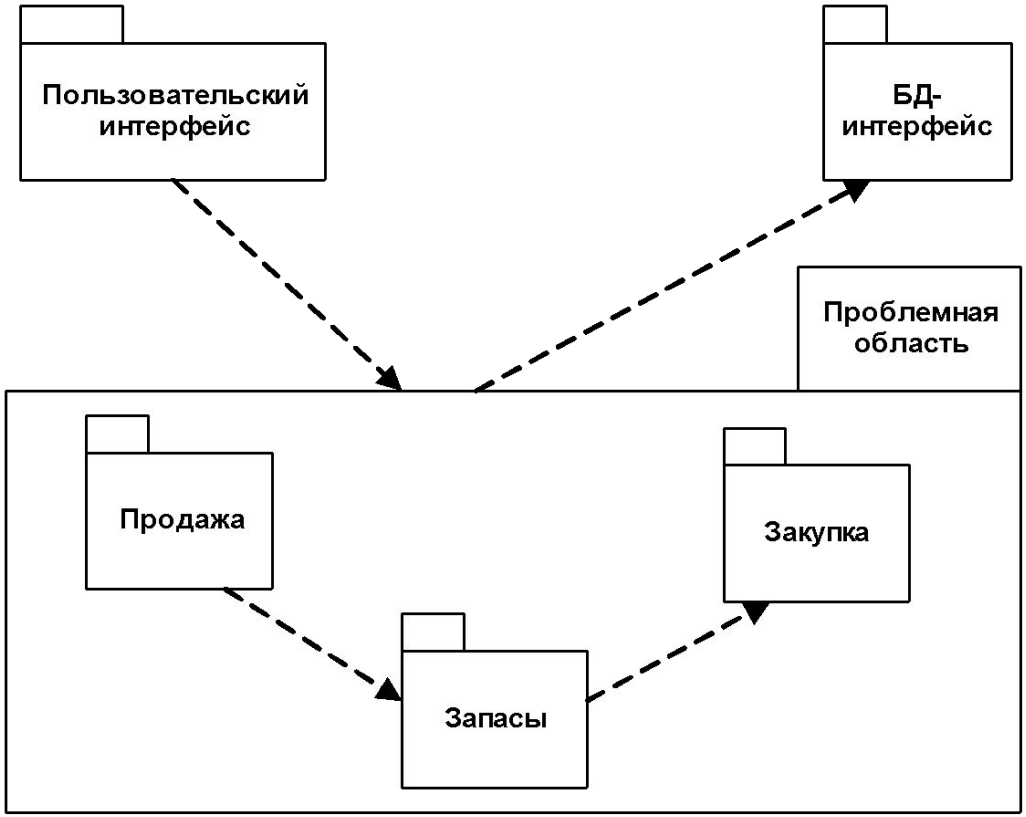
Объектно-ориентированное проектирование

Диаграмма деятельностей может отражать взаимодействие объектов из нескольких прецедентов использования, в частности реализующих отдельно стандартные и альтернативные пути обработки объектов.



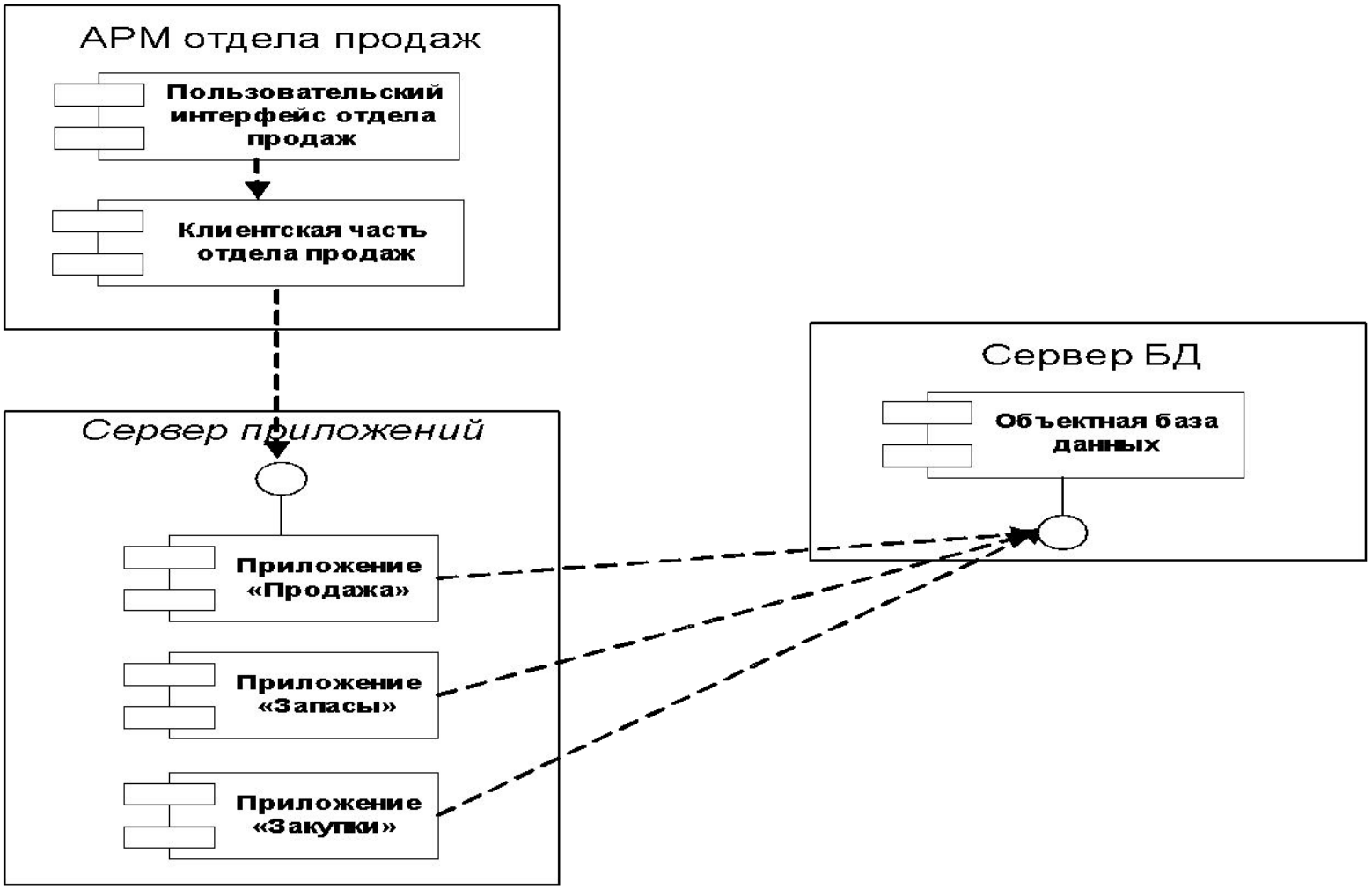
Объектно-ориентированное проектирование

Диаграмма пакетов позволяет описать распределение классов по пакетам.



Объектно-ориентированное проектирование

Диаграммы компонентов и размещения отображает зависимости программных компонентов и топологию расположения компонентов по узлам вычислительной сети.



Прототипное проектирование (RAD-технология)

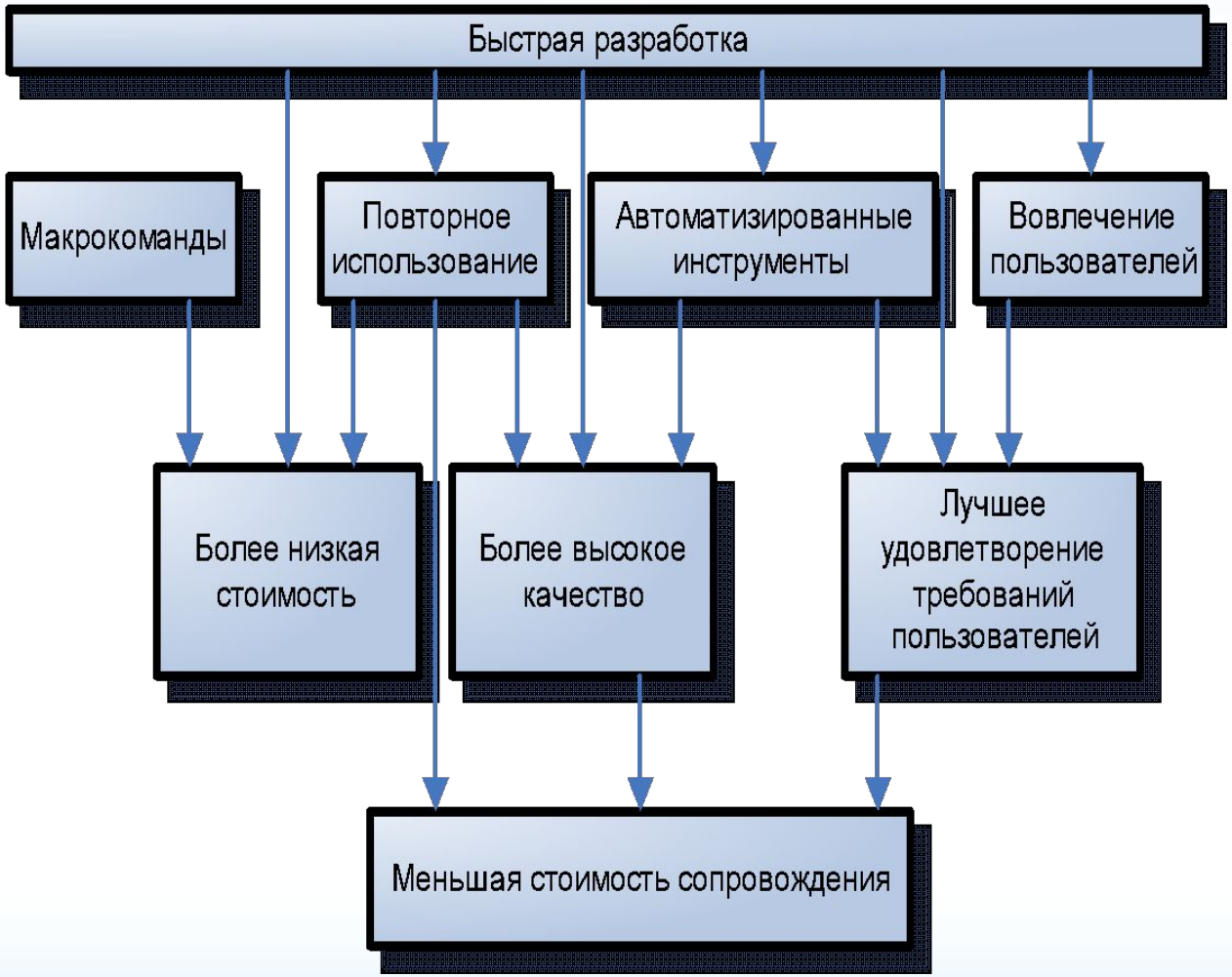
Одним из условий обеспечения высокого качества создаваемых ИС является **активное вовлечение** конечных пользователей в процесс разработки предназначенных для них интерактивных систем, что нашло отражение в методологии прототипного проектирования. Ядром этой методологии является **быстрая разработка приложений RAD** (Rapid Application Development).

Данная технология обеспечивает создание на ранней стадии реализации **действующей интерактивной** модели системы, так называемой **системы-прототипа**, позволяющей:

- наглядно продемонстрировать пользователю будущую систему,
- уточнить его требования,
- оперативно модифицировать интерфейсные элементы:
  - формы ввода сообщений,
  - меню,
  - выходные документы,
  - структуру диалога,
  - состав реализуемых функций.

Прототипное проектирование (RAD-технология)

Основные возможности и преимущества



Прототипное проектирование (RAD-технология)

Приемы для быстрой разработки приложений RAD :

1. Разработка приложения итерациями.
2. Необязательность полного завершения работ на каждом из этапов жизненного цикла для начала работ на следующем.
3. Обязательное вовлечение пользователей в процесс проектирования и построения системы.
4. Высокая параллельность работ.
5. Повторное использование частей проекта.
6. Необходимое применение CASE-средств, обеспечивающих техническую целостность на этапах анализа и проектирования.
7. Применение средств управления конфигурациями, облегчающее внесение изменений в проект и сопровождение готовой системы.
8. Использование автоматических генераторов (мастеров).
9. Использование прототипирования, позволяющего полнее выяснить и удовлетворить потребности конечного пользователя.
10. Тестирование и развитие проекта, осуществляемые одновременно с разработкой нескольких версий прототипа.

Главная задача – как можно быстрее показать пользователям системы работоспособный продукт, тем самым активизируя процесс уточнения и дополнения требований.

Прототипное проектирование (RAD-технология)

Инструментальные средства Прототипного проектирования можно условно разделить на два класса:

- инструменты быстрой разработки приложения в развитых СУБД - класс DEVELOPER и
- интегрированные инструменты быстрой разработки приложений - класс BUILDER.

К инструментам этих классов можно отнести средства 4GL (генераторы компонентов приложений):

- генераторы таблиц базы данных;
- генераторы форм ввода-вывода;
- генераторы запросов;
- генераторы отчетов;
- генераторы меню.

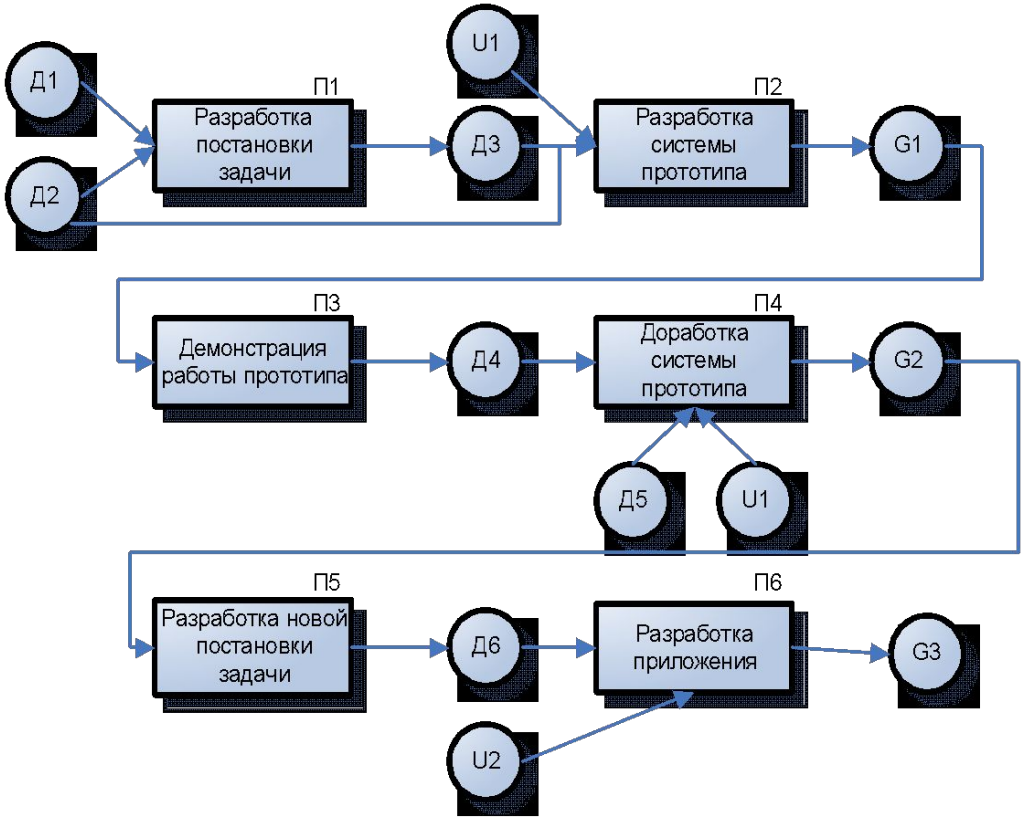
Прототипное проектирование (RAD-технология)

Жизненный цикл создания ИС на основе RAD-технологии предполагает после формирования технического задания и декомпозиции системы независимую разработку подсистем с последующей сборкой, тестированием и внедрением комплексной ИС.

Прототипное проектирование (RAD-технология)

Существуют два базовых варианта организации технологического процесса проектирования с использованием систем-прототипов.

В первом варианте создание системы-прототипа используется для лучшей спецификации требований к разработке ЭИС, после разработки которых сам прототип оказывается ненужным. **Основным недостатком** этот варианта является неэффективное использование системы-прототипа, а именно: прототипы не используются в дальнейшей разработке ИС после того, как выполнили свою первую задачу – устранили неясности в проекте.



Д1 – техническое задание на разработку; Д2 – описание предметной области, Д3 – постановка задачи; U1 – универсум средств быстрой разработки приложений; G1 – приложенне-прототип ; Д4 – результаты работы приложения-прототипа; Д5 – замечания и уточненные требования к ЭИС; G2 – доработанный прототип, Д6 – новая постановка задачн; U2 – универсум средств разработки приложений; G3 – готовое приложение

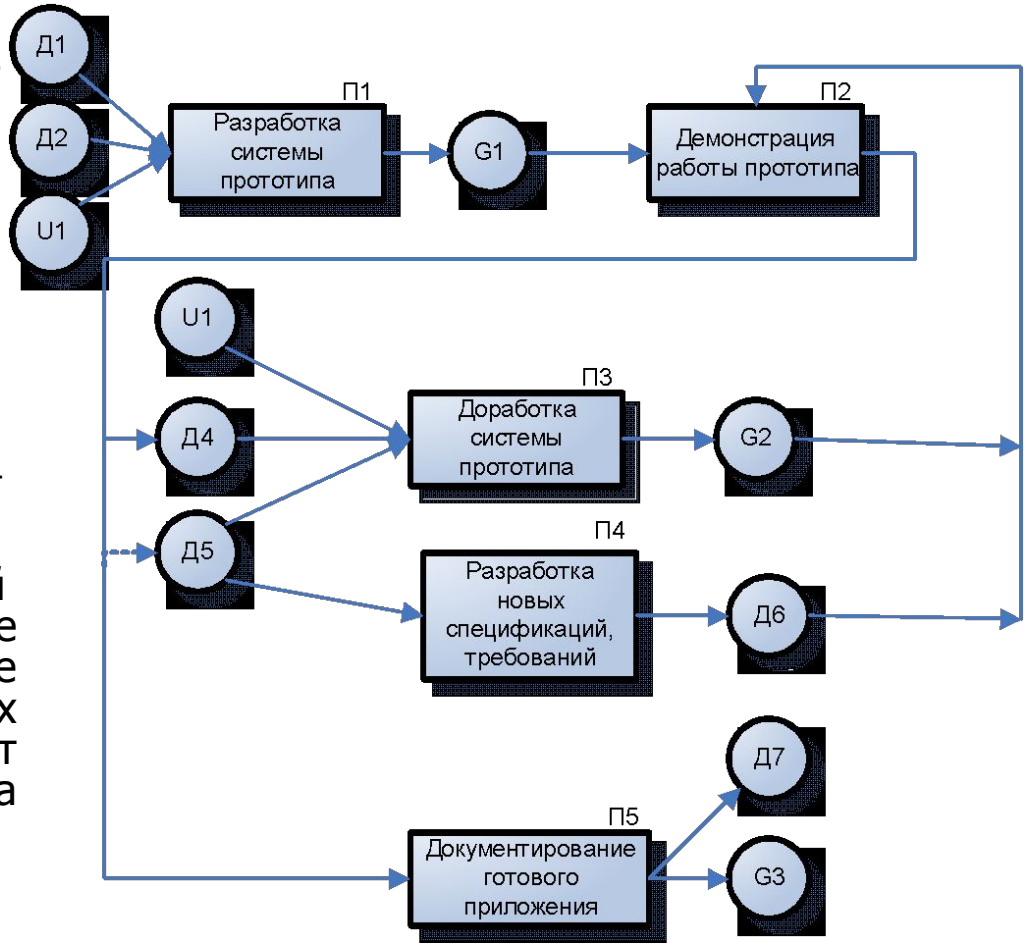
Прототипное проектирование (RAD-технология)

Второй вариант предполагает итерационное развитие системы-прототипа в готовый для эксплуатации программный продукт.

Итерации разработки системы-прототипа включают:

- создание/модификацию системы-прототипа,
- ее демонстрацию пользователю и согласование,
- разработку новых спецификаций-требований к системе,
- новую модификацию и т.д., пока не будет создано готовое приложение.

Основным достоинством прототипной технологии является значительное снижение объема доработок ИС при ее внедрении, который для традиционных методов проектирования, как показывает опыт, соразмерен с затратами на первоначальную реализацию.



Д1 – техническое Задание на разработку; Д2 – описания предметной области; U1 – универсум средств быстрой разработки приложений; G1 – приложение-прототип; Д4 – результаты работы приложения-прототипа; Д5 – замечания и уточнеиные требования к ЭИС; G2 – доработанный прототип; Д6 – новые спецификации-требования; G3 – готовое приложение