Разработка архитектуры ПС.

Отвагин Алексей Владимирович,
доцент каф. ЭВМ, к.т.н., а. 505-5

Содержание

- Понятие архитектуры
- Основные разновидности архитектур
- Архитектурные функции
- Контроль архитектуры

Что такое архитектура?

- Общий взгляд на ПС извне и его представление в виде системы из взаимодействующих компонентов
- Цель проектирования – выбор архитектуры, соответствующей требованиям к ПС
- Задача проектирования – создание описания ПС в виде набора подсистем

Учет требований при создании архитектуры

- Основные цели разработки:
  - Расширение – поддержка добавления новых возможностей в приложение
  - Изменение – облегчение смены требований в процессе эксплуатации приложения
  - Простота – упрощение с целью возможности быстрой реализации
  - Эффективность – достижение эксплуатационных характеристик

Структуризация

- Выполняется в виде блочной диаграммы, отражающей иерархию компонентов
- Каждый блок соответствует компоненту, блоки внутри блоков – вложенным подсистемам
- Стрелки указывают поток данных или управления между подсистемами

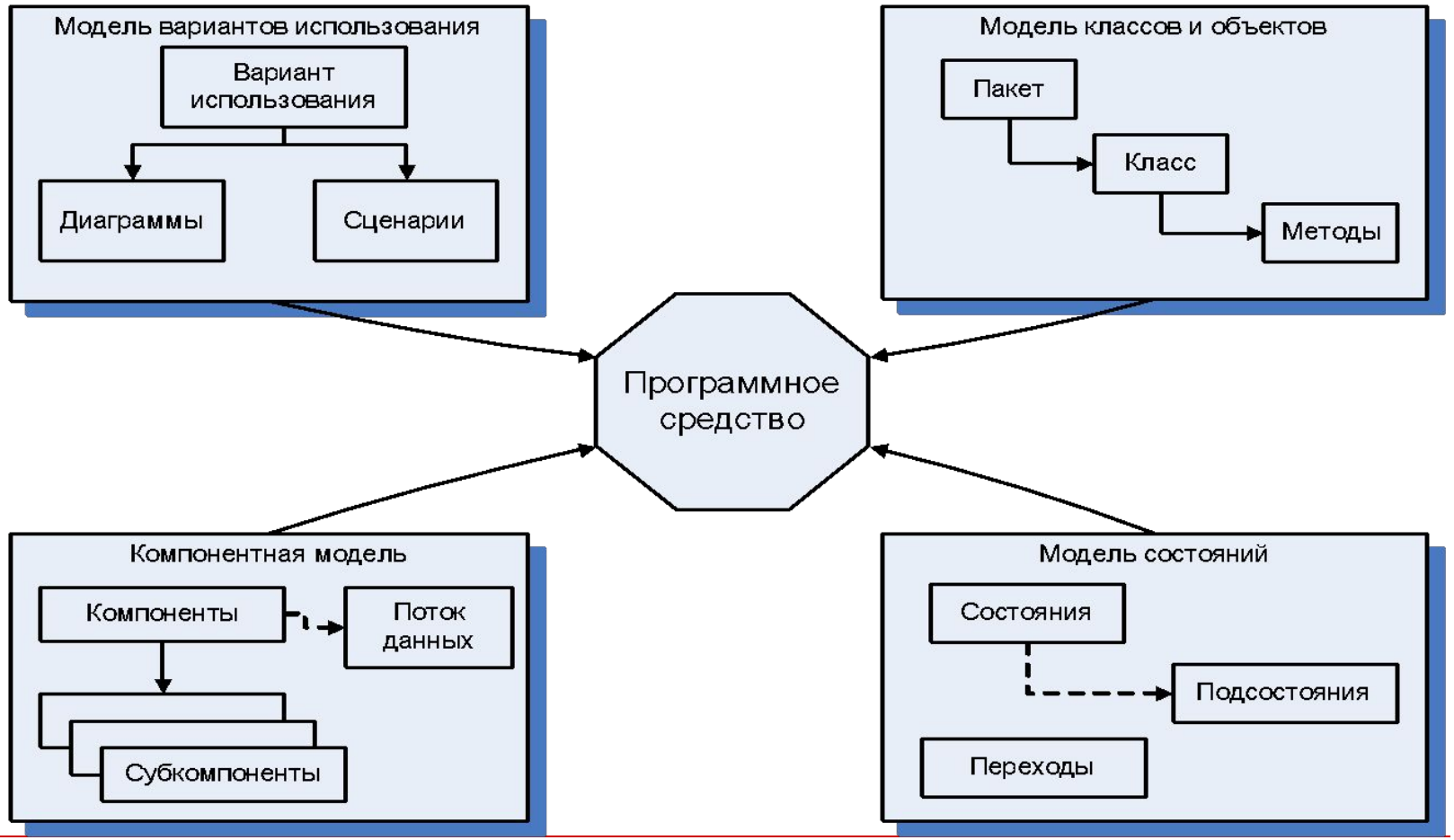
Повторное использование архитектуры

- Многие системы строятся на общих принципах, свойственных приложениям определенной предметной области
- Многие ПС имеют различные версии, основанные на одной и той же архитектуре
- Используются шаблоны архитектур – справочный код (reference code)

Архитектурные модели

- Представляют описание архитектуры с различных сторон
  - Статические модели
  - Динамические модели процессов
  - Модели интерфейсов между компонентами
  - Модели отношений (например, DFD)
  - Модели развертывания

Основные классы архитектурных моделей



Системная организация

- Отражает базовую стратегию структуризации системы
- Четыре основных типа организации:
  - Системы потоков данных
  - Репозиторий разделяемых данных
  - Разделяемые сервисы и серверы
  - Абстрактная машина (слоистая архитектура)

Системы потоков данных

- Процесс обработки управляется данными
- Данные перемещаются между этапами обработки
- Основные элементы: поток данных, процесс, файл

Варианты систем потоков данных

- Трансформер – входящий поток преобразуется во внутренний формат и обрабатывается последовательностью операций. После обработки поток преобразуется в выходной формат.
- Поток транзакций – элемент потока обрабатывается согласно своему типу по собственному пути. Каждый процесс в потоке может иметь несколько выходов.

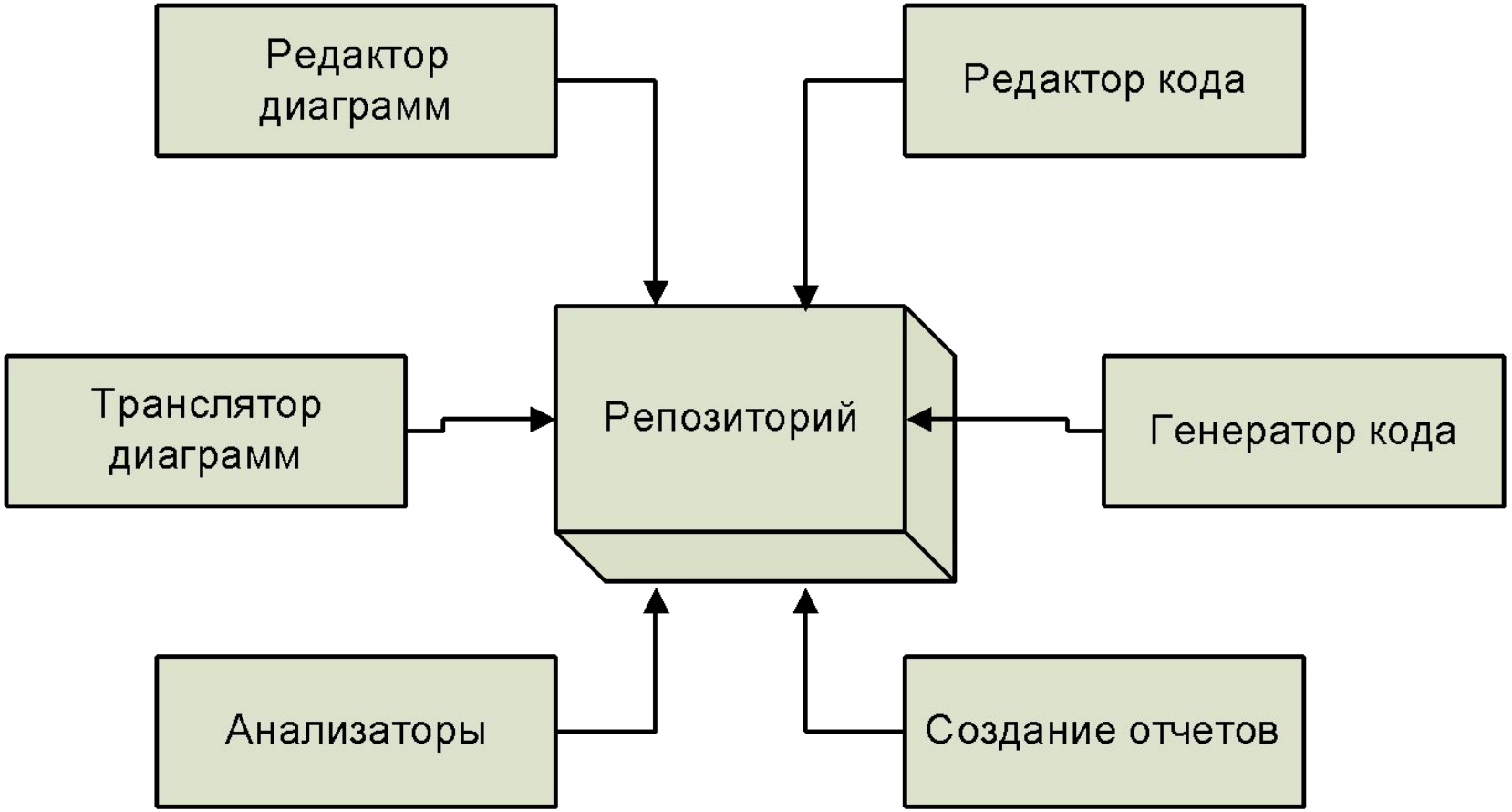
Репозиторий

- Данные содержатся в общей базе (репозитории)
- Компоненты обмениваются данными неявно
- Необходимы средства синхронизации
- Удобно для обмена большими объемами данных

Виды репозиториев

- Пассивный – входящий поток событий или транзакций вызывает обработку данных в репозитории (пример - СУБД)
- Активный – состояние репозитория вызывает процессы обработки и изменения (пример – blackboard systems)

Пример: CASE-система



Клиент-сервер

- Модель распределенной системы, отражающая распределение данных и процессов по компонентам
- Основные компоненты:
  - Множество серверов, определяющих услуги, предоставляемые ПС
  - Множество клиентов, использующих сервисы системы
  - Коммуникационная сеть для организации доступа

Характеристики модели «клиент-сервер»

- Преимущества
  - Простое и очевидное распределение данных
  - Ориентированность на сетевые среды
  - Простота расширяемости (ввод новых услуг или изменение сервисов)
- Недостатки
  - Нет общих разделяемых данных
  - Обмен данными может быть неэффективным
  - Требуется служба имен, указывающая перечень сервисов и их размещение

Виды модели «клиент-сервер»

- «Толстый» клиент
- «Толстый» сервер
- Сбалансированная система
- Peer2Peer
- Распределенный клиент или сервер

Разновидности серверов

- Сервер без состояния – не хранит состояния соединений с клиентами и текущих операций
- Сервер с состоянием – следит за действиями клиентов и позволяет сократить количество взаимодействий между сервером и клиентами

Модель абстрактной машины

- Описывает интерфейс между подсистемами
- Представляет систему в виде набора слоев (абстрактных машин), каждый из которых предоставляет некоторый сервис
- Слои известны извне только через свой интерфейс

Характеристики абстрактной машины

- Преимущества:
  - Повышение уровня абстракции – возможность легко выразить сложные задачи
  - Простота сопровождения за счет изолированности слоев
  - Повторное использование кода
  - Стандартизация
- Недостатки:
  - Сложность выделения слоев в системах
  - Потери производительности при большом количестве слоев

Модели управления

- Определяют потоки управления между подсистемами
- Основные модели:
  - Централизованное управление – одна из подсистем управляет деятельностью всех остальных
  - Управление на основе событий – каждая подсистема реагирует на поступающие события, источниками которых являются другие подсистемы или внешняя среда

Централизованное управление

- Модель «вызов-возврат»
  - Процедурная модель, когда вызов проходит от процедур верхнего уровня к нижнему. Применяется в последовательных системах.
- Модель менеджера
  - Один из компонентов определяет моменты старта, завершения и взаимодействия между остальными элементами системы.

Управление на базе событий

- Действия системы зависят от поступающих извне событий
- Две основных модели:
  - Широковещательная модель
  - Системы с прерываниями

Широковещательная модель

- Событие передается всем подсистемам
- Подсистема реагирует на событие, если она умеет его обрабатывать
- Применяется коммутатор – система подписки на определенные события

Характеристики широковещательной модели

- Преимущества
  - Простота расширения – подсистема регистрируется в коммутаторе и получает события
  - Нет нужды знать имя и расположение корреспондента
- Недостатки
  - Неизвестно, будет ли обработано событие
  - Возможен конфликт из-за наличия систем, регистрирующих одинаковые события

Система с прерываниями

- Гарантирует быстрый ответ на внешние события
- Содержит независимый механизм определения прерываний и вызова соответствующего обработчика
- Обработчик может вызывать дополнительные процедуры

Слоистая архитектура

- Состоит из уровней – логически связанных коллекций элементов программного обеспечения
- Уровень скрывает свою функциональность от других и предоставляет ее через интерфейс
- Система легко модернизируется

Архитектурные функции

- Обеспечивают взаимодействие между подсистемами
- Расширяют спецификацию ПС
- Реализуют механизмы взаимодействия (порты, сообщения и др.)

Система портов

- Порт – подсистема для обслуживания очереди сообщений
- Имеет общесистемное обозначение
- Во многих случаях реализуется на уровне ОС

Гибкие и жесткие порты

- Жесткий порт – явно указанный системный порт (по имени или номеру)
- Гибкий порт – виртуальный порт, используемый в программе. При запуске связывается с фиксированным жестким портом.

Системы с сообщениями

- Обеспечивают эффективное удаленное взаимодействие
- Требуют наличия службы имен
- Сообщения могут быть ориентированы на пользователя
- Наиболее развитая система архитектурных функций