Дисциплина

СПЕЦИФИКАЦИЯ, ПРОЕКТИРОВАНИЕ И АРХИТЕКТУРА ПРОГРАММНЫХ СИСТЕМ

Лекция 1

ПРОГРАММНАЯ ИНЖЕНЕРИЯ. ОСНОВНЫЕ ПОНЯТИЯ И СТАНДАРТЫ.

ЧЕМ ПРОГРАММНАЯ ИНЖЕНЕРИЯ ОТЛИЧАЕТСЯ ОТ РАЗРАБОТКИ ПРОГРАММ?

Программная инженерия. Определение.

Программная инженерия — это область компьютерной науки и технологии, которая занимается построением программных систем, настолько больших и сложных, что для этого требуется участие слаженных команд разработчиков различных специальностей и квалификаций.

СИСТЕМА – ЭТО…?

Система - совокупность взаимодействующих компонентов, объединенных для выполнения определенной функции или набора функций, и взаимосвязей между ними. Система существует для выполнения одной или более миссий в своем окружении.

Миссия - это применение или действие, для которого одно или несколько заинтересованных лиц планируют использовать систему в соответствии с некоторым набором условий.

Окружение, или контекст, совокупность экономических, эксплуатационных, технологических, политических и других факторов, влияющих на систему.

Заинтересованное лицо - это физическое лицо, группа или организация (или ее категории), которые заинтересованы в системе или имеют связанные с ней задачи.

Автоматизированная система

Это совокупность:

- функциональных и информационных процессов конкретной предметной области;
- средств и методов сбора, хранения, анализа, обработки и передачи информации, зависящих от специфики области применения;
- методов управления процессами решения функциональных задач, а также информационными, материальными и денежными потоками в предметной области.

Автоматизированная система: определение из стандартов

Это совокупность составных частей:

- система баз данных: база данных (БД) вместе с системой управления базами данных (СУБД);
- прикладное программное обеспечение;
- персонал;
- организационно-методическое (нормативное) обеспечение;
- технические средства.

Предмет дисциплины

Преимущественно-программная система – это любая автоматизированная система, в которой программное обеспечение оказывает значительное влияние на проект, конструкцию, развертывание и развитие всей системы.

Системный подход

Системный подход — это методология исследования объектов любой природы как систем, которая ориентирована на:

- ✓ раскрытие структуры и составных частей объекта, обеспечивающих его целостность;
- ✓ выявление многообразных типов связей между составными частями объекта и с его окружением;
- ✓ раскрытие механизмов реализации связей и обеспечения требуемого поведения объекта.

Задание
на погружение в контекст

ВЕЛОСИПЕД

Подведем итог выполнения задания

1. Видение системы и мотивационные факторы различны для разных заинтересованных сторон проекта
2. При разработке системы отталкиваемся от потребностей клиента, общаемся с потребителем
3. Требования – основа разработки
4. Необходим единый язык для общения
5. Систему создаем преимущественно из стандартных компонентов на основе успешного опыта
6. Делаем то, что нужно заказчику,
   и не делаем того, что заказчику не нужно
7. Выполняем проект в условиях ограничений

ЧТО ТАКОЕ ПРОЕКТИРОВАНИЕ ?

Цель проектирования системы

Создание облика системы, которая:

- удовлетворяет заданным (возможно, неформальным) функциональным спецификациям;
- согласована с накладываемыми нефункциональными ограничениями;
- удовлетворяет явным и неявным требованиям по эксплуатационным качествам и потреблению ресурсов;
- удовлетворяет явным и неявным критериям по дизайну;
- удовлетворяет требованиям к самому процессу разработки, таким, например, как продолжительность и стоимость, а также привлечение дополнительных инструментальных средств.

Архитектура – результат проектирования системы

Архитектура системы

Это базовая ***организация системы****, воплощенная в ее* ***компонентах****, их* ***отношениях*** *между собой и с* ***окружением****, а также* ***принципы****, определяющие* ***проектирование и развитие*** *системы [IEEE 1471]*

Набор базовых требований к системе, не зависящих от предметной области, определяется **классом** и **метафорой** системы

Классификация систем

Метафоры автоматизированных систем

1. Учетная система
2. Аналитическая система
3. Потоковая система
4. Моделирующая система

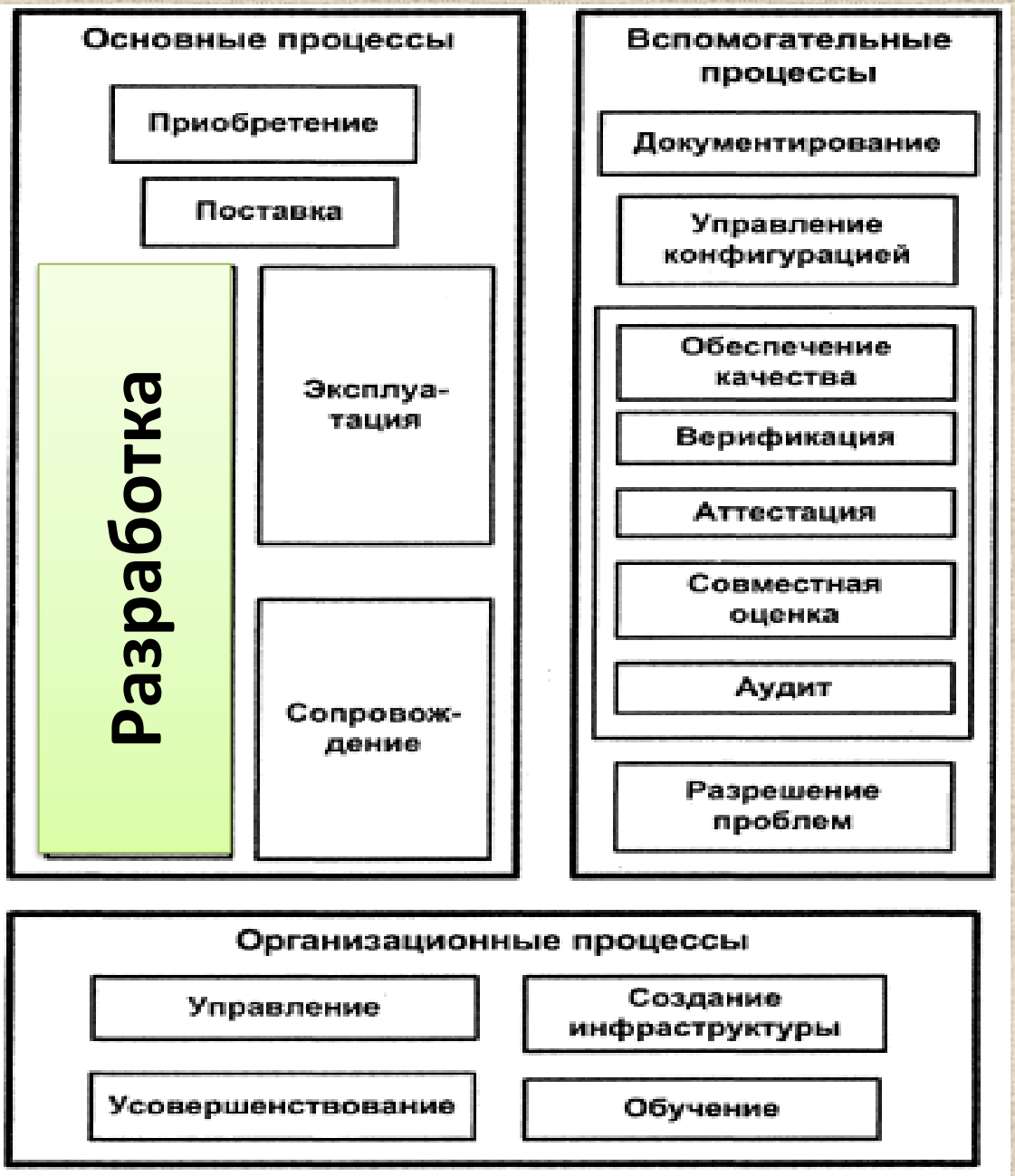
Место проектирования в жизненном цикле программной системы

Жизненный цикл системы - период времени, который начинается с момента принятия решения о необходимости создания программной системы и заканчивается в момент полного изъятия программной системы из эксплуатации.

стандарт ISO/IEC 12207:
1995 «Information Technology - Software Life Cycle Processes»

ГОСТ Р ИСО/МЭК 12207-99:
Процессы жизненного цикла программных средств

Структура ЖЦ программной системы (ГОСТ Р ИСО/МЭК 12207-99)



Процесс разработки программной системы

Данный процесс состоит из следующих работ:

1) подготовка процесса;

2) анализ требований к системе;
3) проектирование системной архитектуры;
4) анализ требований к программным элементам;
5) проектирование программной архитектуры;
6) **техническое проектирование программных компонентов**;

7) программирование и тестирование программных элементов;
8) сборка программных элементов;
9) испытания программных элементов;
10) сборка системы;
11) испытания системы;
12) ввод в действие системы.

Анализ требований к системе

Определение функциональных возможностей информационной системы, пользовательских требований, требований к надежности и безопасности, к внешним интерфейсам и др.

Критерии для требований к автоматизированной системе:
- реализуемость;
- подтверждаемость (возможности проверки) при тестировании.

Критерии для требований к программному обеспечению:
- соответствие требованиям к системе;
- реализуемость;
- подтверждаемость (возможности проверки) при тестировании.

Проектирование системной архитектуры

Определение компонентов оборудования системы, программного обеспечения и операций, выполняемых персоналом, эксплуатирующим систему.

Архитектура системы должна соответствовать требованиям, предъявляемым к системе, а также принятым проектным стандартам и методам.

Проектирование системной архитектуры включает следующие задачи:

- ✓ трансформацию требований к системе в архитектуру, определяющую на высоком уровне структуру технических средств и программного обеспечения
- ✓ распределение всех требований к системе между элементами архитектуры;
- ✓ разработку и документирование программных интерфейсов взаимодействия
- ✓ разработку баз данных на высоком уровне;
- ✓ разработку и документирование предварительных требований к тестам и плана интеграции программного обеспечения.

Анализ требований к программным элементам

Разработчик должен установить и документально оформить следующие требования к программным элементам:

- ✓ функциональные и технические требования, включая производительность, физические характеристики и окружающие условия, под которые должен быть создан программный элемент архитектуры;
- ✓ требования к внешним интерфейсам программного объекта архитектуры;
- ✓ требования защиты, включая требования, относящиеся к допустимой точности информации;
- ✓ эргономические требования, включая требования, относящиеся к ручным операциям, взаимодействию «человек-машина», персоналу и областям, требующим концентрации внимания человека, связанным с чувствительностью объекта к ошибкам человека и квалификации персонала;
- ✓ требования к определению данных и базе данных;
- ✓ требования к документации пользователя;
- ✓ требования к обслуживанию и др.

Проектирование программной архитектуры

(применительно к каждому программному элементу архитектуры)

- трансформировать требования к программному элементу в архитектуру, которая описывает общую структуру и определяет компоненты программного элемента и уточнить требования, с точки зрения облегчения технического проектирования;
- разработать и документально оформить общий (эскизный) проект внешних интерфейсов программного элемента и интерфейсов между компонентами;
- разработать и документально оформить общий (эскизный) проект базы данных программного элемента;
- разработать и документально оформить предварительные версии документации пользователя;
- определить и документально оформить предварительные общие требования к испытаниям (тестированию) программного элемента и график сборки

Техническое проектирование программных элементов

(применительно к каждому программному компоненту архитектуры программного элемента)

- описать компонент программного элемента на уровне модулей, которые можно программировать (кодировать), компилировать и тестировать независимо;
- разработать и документально оформить описание внешних интерфейсов программного компонента, интерфейсов между программными модулями
- разработать и документально оформить детальный проект базы данных;
- уточнить пользовательскую документацию;
- определить и документально оформить требования к испытаниям и программе испытаний программных модулей;
- уточнить общие требования к испытанию (тестированию) и программе сборки программного компонента

Модель жизненного цикла программной системы

Модель жизненного цикла ПО - структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении ЖЦ. Модель ЖЦ зависит от специфики, масштаба и сложности проекта и специфики условий, в которых система создается и функционирует.

Модель ЖЦ ПО включает в себя:
1) стадии;
2) результаты выполнения работ на каждой стадии;
3) ключевые события — точки завершения работ и принятия решений.

Спиральная модель Барри Боэма

- отказ от фиксации требований и назначение приоритетов пользовательским требованиям;
- разработка последовательности прототипов, начиная с требований наивысшего приоритета;
- идентификация и анализ риска на каждой итерации;
- использование каскадной модели для реализации окончательного прототипа;
- оценка результатов по завершении каждой итерации и планирование следующей итерации.

как будет полностью завершена работа на текущей стадии) без возвратов на пройденные стадии.

Спиральная модель

Модели ЖЦ «Быстрое макетирование». «Быстрая разработка приложений», «Гибкая разработка»

Составляющие подхода:

- небольшие группы разработчиков (от 3 до 7 человек), выполняющих работы по проектированию отдельных подсистем;
- команда взаимно-заменяемых профессионалов, одержимых одной целью
- вовлечение заказчика и потребителя в процесс разработки
- короткий, но тщательно проработанный производственный график (от двух недель до трех месяцев);
- повторяющийся цикл, при котором разработчики по мере того, как приложение начинает обретать форму, запрашивают и реализуют в продукте требования, полученные в результате взаимодействия с заказчиком (потребителем).

Основные принципы гибкой разработки

- ✓ Разработка приложений итерациями
- ✓ Необязательность полного завершения работ на каждой из стадий жизненного цикла программной системы
- ✓ Обязательность вовлечения пользователей в процесс разработки
- ✓ Применение средств управления конфигурацией, облегчающих внесение изменений в проект и сопровождение готовой системы
- ✓ Использование прототипов, позволяющих полнее выяснить и удовлетворить потребности пользователей
- ✓ Тестирование и развитие проекта, осуществляемые одновременно с разработкой
- ✓ Ведение разработки немногочисленной хорошо управляемой командой профессионалов
- ✓ Грамотное руководство разработкой системы, четкое планирование и контроль выполнения работ
- ✓ Отсутствие требований по документированию проекта

Преимущества и недостатки гибкой разработки

Преимущества:

- ✓ Производительность работы коллектива очень высока.
- ✓ Взаимосвязи с заказчиком являются конструктивными.

Недостатки:

- ✓ При быстром макетировании очень тяжело привести проект к завершающей фазе.
- ✓ Проект, выполняемый с помощью метода быстрого макетирования, сложно планировать и финансировать.
- ✓ Метод быстрого макетирования неприменим для разработки ПО большим коллективом разработчиков.
- ✓ В результате быстрого макетирования можно не получить ничего, кроме прототипа системы.

ОСНОВНОЙ ПРИНЦИП ПРОЕКТИРОВАНИЯ ?

Иерархическая декомпозиция

Принципы «правильной» декомпозиции:

1. **Слабая связанность** (Low Coupling) –
   количество связей между отдельными подсистемами должно быть минимальным.

2. **Сильное сцепление** (High Cohesion) –
   связность отдельных частей внутри каждой подсистемы должна быть максимальной.

3. **Инкапсуляция** – каждая подсистема должна скрывать свое содержимое от других подсистем.

4. **Интерфейсы** – каждая подсистема должна иметь четко определенный интерфейс с другими подсистемами.

КАКИЕ ПОДХОДЫ (МЕТОДОЛОГИИ) ПРОЕКТИРОВАНИЯ СУЩЕСТВУЮТ?

Подходы к проектированию программных систем

Основной принцип структурного подхода – алгоритмическая декомпозиция

Правила структурного проектирования

1. Каждый «черный ящик» должен реализовывать единственную функцию системы.
2. Функция каждого «черного ящика» должна быть легко понимаема независимо от сложности ее реализации.
3. Связь между «черными ящиками» должна вводиться только при наличии связи между соответствующими функциями системы.
4. Связи между «черными ящиками» должны быть простыми, насколько это возможно, для обеспечения независимости между ними.

Подходы к проектированию программных систем

Основа объектно-ориентированного подхода – объектная декомпозиция

Принципы объектно-ориентированного проектирования:

1. Абстрагирование.
2. Инкапсуляция.
3. Модульность.
4. Иерархия.
5. Полиморфизм.

Предметно-ориентированное проектирование (Domain Driven Design)

Качественное проектирование и разработка программного обеспечения определяются выделением и управлением **единой моделью предметной области (доменом)**.

Совокупность формализованной предметной области и процесса управления требованиями позволяют преодолевать сложности частых изменений требований, устраняют проблемы коммуникаций как с заказчиком (потребителем), так и внутри проектной команды.

Предметная область - часть реального мира, рассматриваемая в пределах данного контекста.
Под контекстом может пониматься область исследования или область, определяющая границы некоторой деятельности.

Основные идеи предметно-ориентированного проектирования

1. В большинстве программных проектов основное внимание должно быть сосредоточено на логической структуре предметной области и взаимосвязях в ней.
2. Архитектура сложного программного обеспечения должна основываться на модели предметной области.
3. Итеративный характер разработки.
4. Тесное взаимодействие между разработчиками и специалистами в предметной области.

Выбор модели в предметно-ориентированном проектировании

1. Модель предметной области и архитектура программы взаимно определяют друг друга.
2. Модель предметной области лежит в основе языка, на котором говорят все члены группы разработчиков.
3. Модель предметной области - это дистиллированное знание. Модель представляет собой согласованный между разработчиками способ структуризации знаний из предметной области, а также выделения элементов, представляющих наибольший интерес.

Наиболее известные в области информационных технологий определения понятия модель

✓**Модель** – simulator – программа либо устройство, обеспечивающее имитацию характеристик и поведения определённого объекта.

✓**Модель** (Model) [стандарт ISO-15704] Абстрактное представление реальности в какой-либо форме (в математической, физической, символической, графической, дескриптивной), предназначенное для представления определенных аспектов этой реальности и позволяющее отвечать на изучаемые вопросы.

✓**Модель** – это абстракция, описывающая моделируемую систему (объект) с определенной точки зрения и на определенном уровне абстрагирования.

✓**Моделирование** - разработка модели объекта для использования в процессах проектирования, производства, тестирования, эксплуатации с целью получения знаний об объекте. Моделирование предполагает построение и изучение моделей проектируемых (конструируемых) объектов, реально существующих предметов и явлений.

✓**Моделирование** - разработка в процессе проектирования модели объекта, содержащей знания (решения) о проектируемом объекте (структура, функционирование и т.п.). Причем, в процессе использования знаний компоненты модели могут уточняться и получать дополнительную или другую интерпретацию.

Модель – это всегда упрощение действительности, однако с их помощью можно описывать и объяснять достаточно сложные вещи.

Ключевые вопросы проектирования

- ✓ Как проводить декомпозицию?
- ✓ Как организовать и объединить компоненты в единую систему?
- ✓ Как обеспечить необходимую производительность?
- ✓ Как обеспечить приемлемое качество системы?
- ✓ Параллелизм (Concurrency)
- ✓ Контроль и обработка событий (Control and Handling of Events)
- ✓ Распределение компонентов (Distribution of Components)
- ✓ Обработка ошибок и исключительных ситуаций и обеспечение отказоустойчивости (Errors and Exception Handling and Fault Tolerance)
- ✓ Взаимодействие и представление (Interaction and Presentation)
- ✓ Сохраняемость данных (Data Persistence)

Заключение

1. **Система** - совокупность взаимодействующих компонентов, объединенных для выполнения определенной функции или набора функций, и взаимосвязей между ними. Система существует для выполнения одной или более миссий в своем окружении.
2. **Преимущественно-программная система** – это любая автоматизированная система, в которой программное обеспечение оказывает значительное влияние на проект, конструкцию, развертывание и развитие всей системы.
3. Системный подход, требования, знания предметной области, эффективные коммуникации внутри команды и с заказчиком с использованием средств визуализации – **основа проектирования** автоматизированной системы.
4. **Архитектура программной системы** - базовая организация системы, воплощенная в ее компонентах, их отношениях между собой и с окружением, а также принципы, определяющие проектирование и развитие системы.
5. Набор базовых требований к автоматизированной системе, не зависящих от предметной области, определяется **классом** и **метафорой** системы.
6. **Декомпозиция** – основной принцип проектирования. Способ декомпозиции определяется выбранной методологией проектирования.
7. **Модели** – это всегда упрощение действительности, однако с их помощью можно описывать и объяснять достаточно сложные вещи.
8. **Модель жизненного цикла программной системы** определяет последовательность этапов разработки программной системы.

Спасибо за внимание!

Вопросы?

Нормативно-методическое обеспечение проектирования автоматизированных систем

Документы НМО регламентируют:

- ✓ порядок разработки, внедрения и сопровождения программных систем;
- ✓ общие требования к составу программной системы и связям между его компонентами, а также к его качеству;
- ✓ виды, состав и содержание проектной и программной документации.

Стандарты Российской Федерации

- ГОСТ ЕСПД (Единой Системы Программной Документации — серия ГОСТ 19.XXX)
- 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»
- ГОСТ 34.603-92 «Информационная технология. Виды испытаний автоматизированных систем»

Классификация НМО

- ✓ **по виду регламентации**
  (стандарт, руководящий документ, положение, инструкция, т.п.);
- ✓ **по статусу регламентирующего документа**
  (международный, отраслевой, предприятия);
- ✓ **по области действия документа**
  (заказчик, подрядчик, проект);
- ✓ **по объекту регламентации или методического обеспечения**.

База НМО

- международные стандарты ISO/IEC
  (ISO – International Organization of Standardization - Международная организация по стандартизации, IEC — International Electrotechnical Commission — Международная комиссия по электротехнике);

- стандарты Российской Федерации ГОСТ Р;

- стандарты организации-заказчика.