Лекция 4
Модели процесса создания программного обеспечения

Лекция 4
Модели процесса создания программного обеспечения
4.1 Каскадная модель **«Waterfall Model»**

- Модификации

Общепринятая линейная модель

Классическая итерационная

(У. Ройс 1970 г). Обратная связь после каждого этапа

Каскадная модель

Завершение каждого этапа проверкой

Строгая каскадная модель

Минимизация возвратов к пройденным этапам

Лекция 4

Модели процесса создания программного обеспечения

4.1 Каскадная модель **«Waterfall Model»**

Лекция 4

Модели процесса создания программного обеспечения

4.1 Модификации каскадной модели

Итерационная модель ЖЦ

Лекция 4
Модели процесса создания программного обеспечения
4.1 Каскадная модель

Достоинства:

Имеется план и график по всем этапам

Ход конструирования упорядочен

Имеется богатый опыт использования

Недостатки:

Не всегда соответствует реальным проектам (отсутствие гибкости)

Часто всех требований на начальном этапе нет

Результат доступен только в конце

Лекция 4
Модели процесса создания программного обеспечения
4.2 Модель: Прототипирование (макетирование).

- **Применяется, когда имеются не все требования**
- **Позволяет быстро увидеть некоторые свойства продукта**
  - Удобство
  - Внешний вид
  - Применимость
- **Применяется при проектировании**
  - Информационных систем
  - Программных продуктов с графическим пользовательским интерфейсом
- **Используются средства быстрой разработки приложений**

Visual Basic, .Delphi,C# и тд

Лекция 4
Модели процесса создания программного обеспечения
4.2 Этапы прототипирования.

Лекция 4
Модели процесса создания программного обеспечения
4.2 Этапы прототипирования

Лекция 4
Модели процесса создания программного обеспечения
4.2 Прототипирования

Достоинства

Обеспечивает определение полных требований к ПО

Недостатки

Не является моделью жизненного цикла

Заказчик может принять макет за продукт

Разработчик может принять макет за продукт

Лекция 4

Модели процесса создания программного обеспечения

4.3 Инкрементная модель

- Объединяет классический подход и макетирование
- Весь проект делится на инкременты – версии продукта с определенной функциональностью
- Для каждого инкремента выполняется:

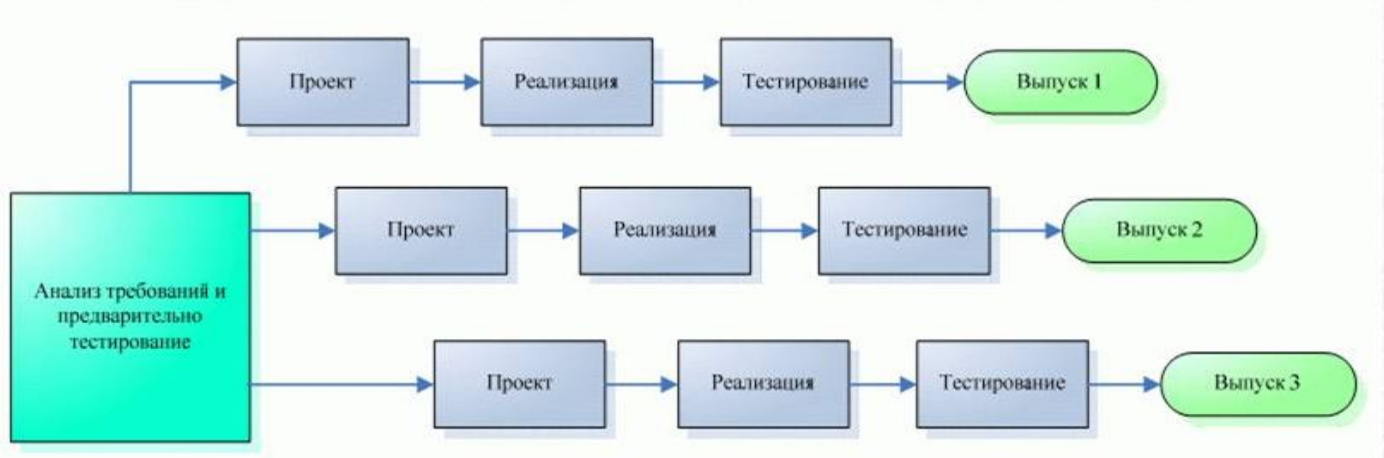
Анализ

Проектирование

Разработка

Тестирование

- Результат каждого инкремента – работающий продукт



Лекция 4
Модели процесса создания программного обеспечения
4.3 Инкрементная модель

Достоинства

Имеется план и график по всем этапам конструирования

Промежуточные версии доступны заказчику

Недостатки

Часто всех требований нет

Не всегда можно спланировать содержание версий

Отсутствует гибкость

Лекция 4
Модели процесса создания программного обеспечения
4.4 Спиральная модель

Предложена Б. Боемом в 1988г

Базируется:

Классическом ЖЦ

Макетирование

Дополнена анализом рисков

Основные компоненты

Планирование

Анализ

Конструирование

Оценивание

Достоинства

- Адекватно отражает эволюционный характер проектирования
- Позволяет учитывать риски на каждом витке эволюции
- Использует моделирование

Недостатки

- Высокие требования к заказчику
- Трудность контроля времени разработки и управления им

Лекция 4
Модели процесса создания программного обеспечения
4.5 Быстрая разработка приложений **RAD** - модель

RAD (Rapid Application Development)

- Инкрементная стратегия конструирования
- Использование компонентно-ориентированного конструирования
- Обеспечение очень короткого цикла разработки (60-90 дней)
- Ориентирована в основном на разработку ИС

Лекция 4
Модели процесса создания программного обеспечения
4.5 **RAD** – модель. Основные этапы.

- Бизнес-моделирование
- Моделирование данных
- Моделирование процесса
- Сборка (генерация) приложений
- Тестирование и объединение (интеграция).

Лекция 4

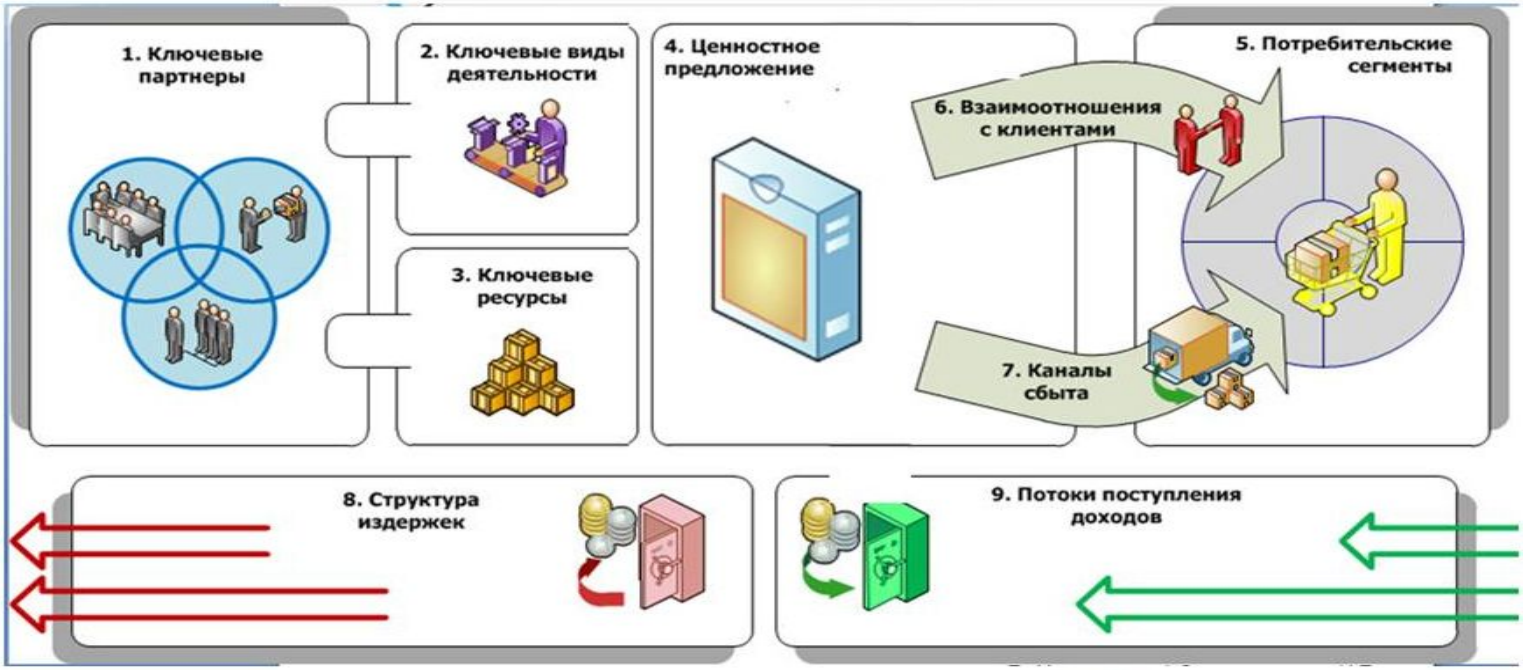
Модели процесса создания программного обеспечения

4.5.1 **RAD** – модель. Бизнес-моделирование.

Моделируется информационный поток между конкретными функциями, которые программа решает.

Определяется:

Какая информация создается
Кто ее создает
Кто ее обрабатывает
Где информация применяется



Лекция 4
Модели процесса создания программного обеспечения
4.5.2 **RAD** – модель. Моделирование данных.

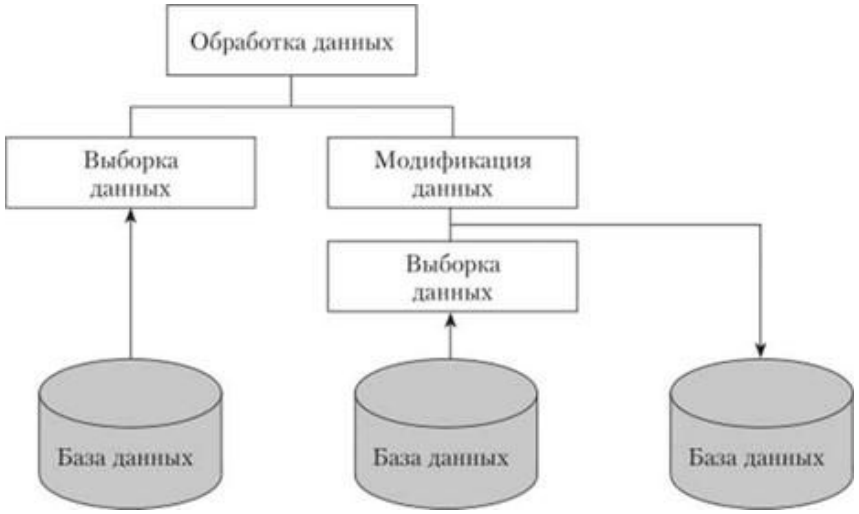
- По информационному потоку формируется набор объектов данных
- Определяются свойства объектов
- Специфицируются отношения между объектами

Лекция 4
Модели процесса создания программного обеспечения
4.5.3 **RAD** – модель. Моделирование обработки.

Определение преобразований объектов данных

Создаются описания для

- добавления объектов данных
- Модификация объектов данных
- Удаление объектов данных
- Поиск объектов данных



Лекция 4
Модели процесса создания программного обеспечения
4.5.4 **RAD** – модель. Генерация приложения

- Применяются языки 4 поколения.
- Используются готовые компоненты
- Создаются повторно используемые компоненты
- Применение средств автоматической сборки

Лекция 4
Модели процесса создания программного обеспечения
4.5.5 **RAD** – модель. Тестирование и объединение

- **Тестирование упрощается из-за повторного использования компонентов**

Они не требуют автономного тестирования

- **Используется интеграционное тестирование**

Лекция 4
Модели процесса создания программного обеспечения
4.5.6 **RAD** – модель. Ограничения

- **Область применения** – проектирование информационных систем.
- **Производительность** не является критичной
- Необходимо большое количество разработчиков
- Отсутствие технических рисков

Лекция 4
Модели процесса создания программного обеспечения
4.6 **Унифицированный процесс Rational (RUP)**

- Начало разработки 1995г. Первая версия RUP – 1998г.
- Наиболее глубоко проработанная методология

Grady Booch
Грэди Буч

James Rumbaugh
Джеймс Рамбо

Ivar Jacobson
Айвар Якобсон

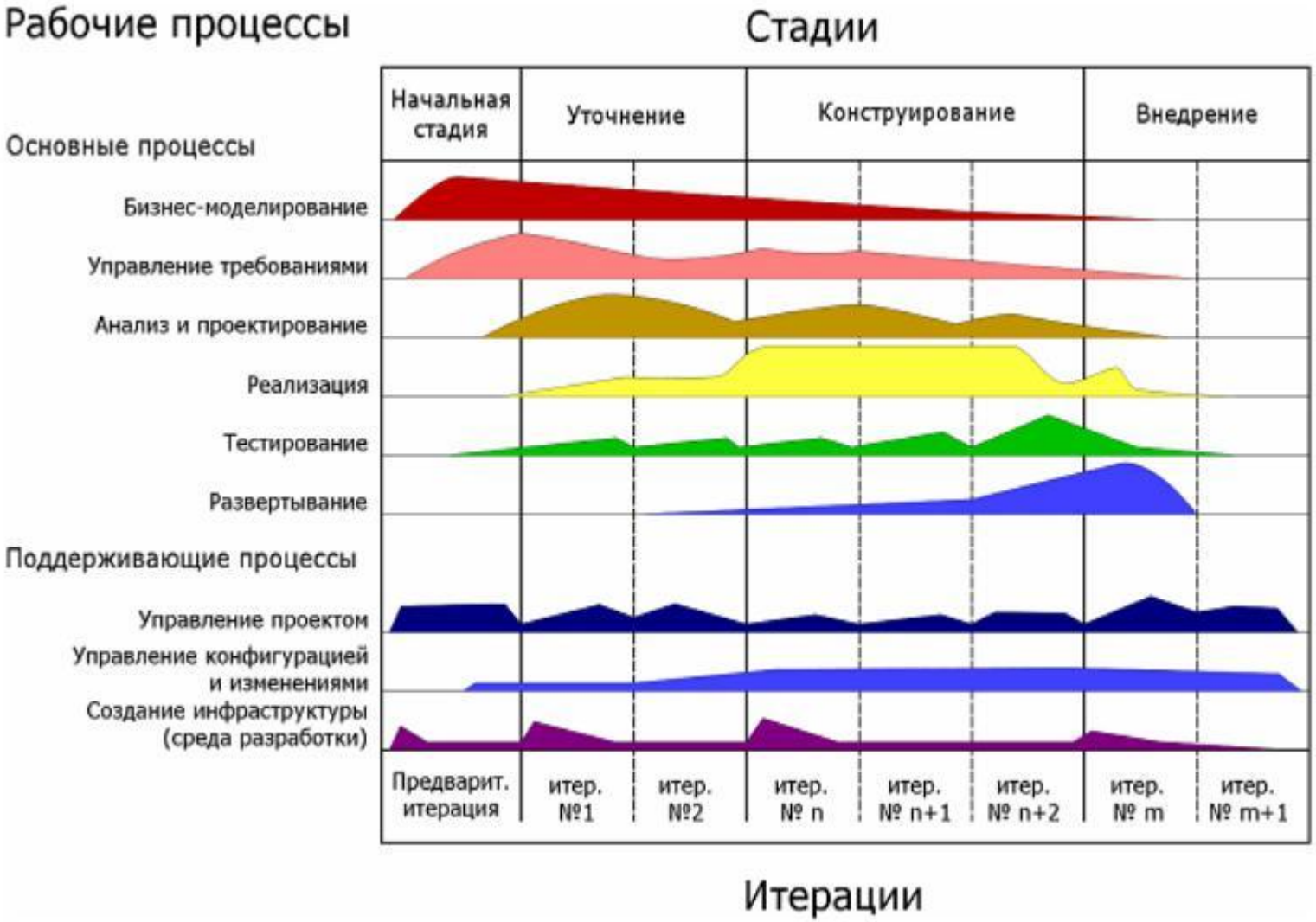
Лекция 4
Модели процесса создания программного обеспечения
4.6 **Унифицированный процесс Rational (RUP)**

- Инкрементная и эволюционная итеративная методология
- Базируется на использовании языка UML
- На всех стадиях используются программные метрики
- Процесс делится на этапы (стадии)
- **Этап** состоит из итераций
- **Итерация** – законченный цикл разпработки, вырабатывающий промежуточный продукт

Лекция 4

Модели процесса создания программного обеспечения

4.6 **Унифицированный процесс Rational (RUP)**



Лекция 4
Модели процесса создания программного обеспечения
4.6 Рабочие потоки процесса RUP

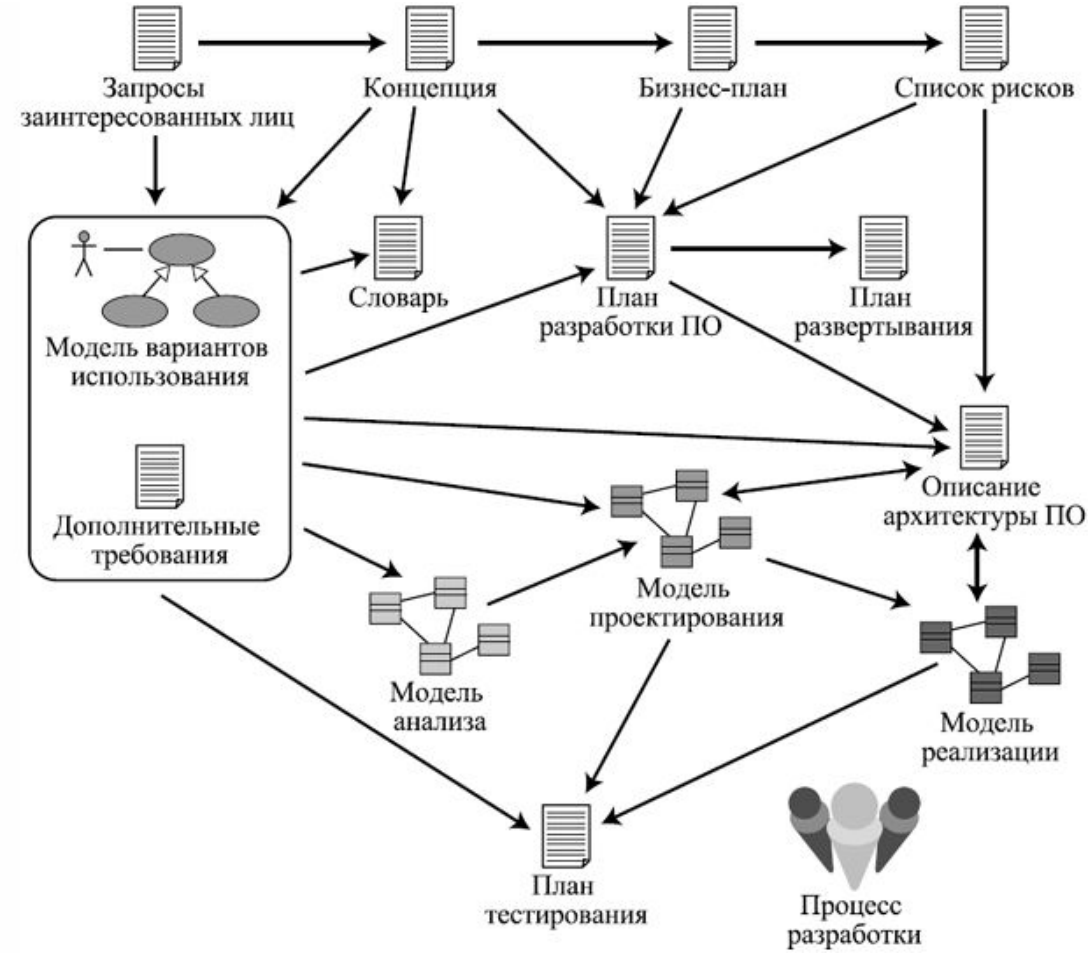
- **Бизнес-моделирование**
- **Управление требованиями**
- **Анализ и проектирование**

  Создание статистического и динамического представления системы
- **Реализация**

  Создание программного кода
- **Тестирование**

  Проверка системы в целом

Лекция 4
Модели процесса создания программного обеспечения
4.6 Артефакты RUP



Лекция 4
Модели процесса создания программного обеспечения
4.6.1 RUP. Начальная стадия (Inception)

- **Назначение**

  запуск проекта

- **Цели**

  Определение области применения

  Определение функционгальных для системы элементов (Use Case)

  Разработка предварительной архитектуры (общие черты)

  Определение общей стоимости и плана проекта

  Идентификация основных элементов риска

Лекция 4
Модели процесса создания программного обеспечения
4.6.1 RUP. Начальная стадия. Действия

- **Формулировка области применения проекта**

  Выявление требований и ограничений

- **Планирование**

  Подготовка бизнес-варианта и альтернатив развития для управления риском

  Определение персонала

  Определение проектного плана

  Определение зависимости между стоимостью, временем и функциональностью.

- **Синтез предварительной архитектуры**

  Развитие решений проектирования

  Определение используемых компонентов

Лекция 4
Модели процесса создания программного обеспечения
4.6.1 RUP. Начальная стадия. Артефакты

- Спецификация основных проектных требований (20%)
- Словарь
- Начальный бизнес-вариант
- Начальная оценка риска
- Проектный план с этапами и итерациями

Лекция 4
Модели процесса создания программного обеспечения
4.6.2 RUP. **Уточнение (Elaboration)**

- **Назначение**

  Создать архитектурный базис
- **Цели**

  Определяются оставшиеся (80%) требования

  Определяется архитектурная платформа

  Отслеживание рисков, устранение наибольших рисков

  Разработка плана итераций на этапе “Конструирование”

Лекция 4
Модели процесса создания программного обеспечения
4.6.2 RUP. **Уточнение (Elaboration).**

Действия

- Развитие спецификации
- Формирование критических элементов Use Case, задающих дальнейшие решения

Артефакты

- Модель Use Case (80%). Дополнительные (не функциональные) требования
- Описание программной архитектуры
- Действующий архитектурный макет
- Переработанный список элементов рисков и переработанный бизнес-вариант
- План разработки всего проекта

Лекция 4

Модели процесса создания программного обеспечения

4.6.3 RUP. Конструирование **(Construction).**

- **Назначение**

  Создание программного продукта с начальной функциональностью
- **Цели**

  Минимальная стоимость разработки

  Быстрое получение требуемого качества

  Быстрое получение версий
- **Действия**

  Управление ресурсами, контроль ресурсов

  Оптимизация процессов

  Полная разработка компонентов и их тестирование

  Оценивание реализации продукта и корректировка списка рисков
- **Артефакты**

  Программный продукт (альфа-, бета-версия)

  Описание текущей реализации

  Руководство пользователя

Лекция 4
Модели процесса создания программного обеспечения
4.6.4 RUP. Внедрение (Transition).

- **Назначение**

  Отдать программный продукт пользователям

  Завершить выпуск продукта

- **Действия в каждой итерации**

  Выпуск бета-версий или релизов

  Исправление найденных в процессе бета-тестирования ошибок

- **Результат**

  Законченный продукт

Лекция 4

Модели процесса создания программного обеспечения

4.7 Гибкие (**agile**) методологии.

Основные особенности

- Отказ от классических “неповоротливых” подходов
- Направленность на проекты с постоянно меняющимися требованиями
- Небольшие команды
- (!) Высокая значимость организационных и социальных составляющих процесса

2001 год – Agile Manifesto

важнее

Люди и взаимодействия	процессов и инструментов
Работающий продукт	исчерпывающей документации
Сотрудничество с заказчиком	согласование условий контракта
Готовность к изменениям	следование первоначальному плану

Лекция 4
Модели процесса создания программного обеспечения
4.7 Гибкие (agile) методологии

- Scrum
- Экстремальное программирование (XP)
- Бережливая разработка ПО (Lean Software Development)
- Agile Unified Process (AUP)
- Feature Driven Development (FDD)

Лекция 4
Модели процесса создания программного обеспечения
4.7.1 Экстремальное программирование **XP**

- Автор — Кент Бек, 1999год.
- Ориентирован на группу до 10 человек.
- Группа размещается в одном помещении
- **Процесс:**

  гибкий и динамичный

  пригоден для проектов с изменяющимися требованиями

  процесс итеративен

- **Основные действия**

  Кодирование

  Тестирование

  Выслушивание заказчика

  Проектирование

- **Динамика определяется**

Непрерывностью связи с заказчиком

Простотой – выбирается простейшее решение

Быстрой обратной связью

Профилактика проблем

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 Методы **XP**

- Игра в планирование (замена полноценного планирования)
- Небольшие версии
- Метафора
- Простой дизайн (выбираем самое простое решение
- Тестирование
- Рефакторинг
- Парное программирование
- Коллективное владение кодом
- Непрерывная интеграция
- 40-часовая рабочая неделя
- Локальный заказчика
- Стандарты кодирования

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 **XP.** Методы

1. Игра в планирование

Заказчик:

- Объем работ
- Приоритет
- Композиция версий
- Сроки выпуска версий

Разработчик:

- Временные оценки
- Последствия
- Процесс
- Подробный график работ

2. Небольшие версии

Быстрый запуск простой системы

Версия маленькая, насколько это возможно

Версия должна быть завершенной

3. Метафора

Глобальное “видение” проекта, понятное всем

Замена большой архитектуры (основная идея проекта)

4. Простой дизайн (правильный)

Выполняются все тесты

нет дублирующей логики

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 XP. Методы

- **Рефакторинг**

  изменение программы для упрощения добавления новой функциональности

  Изменение программы после добавления новой функциональности

  Программы до и после рефакторинга функционально эквивалентны
- **Парное программирование**

  Разработчики работают парами

  Первый думает о реализации

  Второй думает стратегически

  Состав пар меняется

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 XP. Методы

- **Коллективное владение**

  Код – общая собственность

  При необходимости код модифицируется немедленно, независимо от авторства
- **Непрерывная интеграция**

  Код интегрируется раз в несколько часов. Не реже 1-го раза в день

  Интеграция нового кода заканчивается после прохождения системой всех тестов

  Ответственна за интеграцию пара, которая внесла изменения

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 XP. Методы

- **40-часовая рабочая неделя**

  Сверхурочные – крайне нежелательны

  Если постоянно требуется переработка – неправильно организован проект

  Отпуск обязателен
- **Локальный заказчик**

  В состав команды – представитель заказчика

  Представитель – пользователь системы

  Представитель отвечает на вопросы разработчиков и расставляет мелкие приоритеты

Лекция 4
Модели процесса создания программного обеспечения
4.7.2 XP. Методы

- **Общие стандарты кодирования**

  Единый стандарт кодирования

  Стандарт должен способствовать коммуникациям

  Стандарт должен быть добровольно воспринят командой

Лекция 4
Модели процесса создания программного обеспечения
4.8.1 **SCRUM**.

Подход впервые описали **Хиротака Такэути[en] и Икудзиро Нонака[en]** в статье “The New Product Development Gam”. (Harvard Business Review, январь-февраль 1986)

Впервые методология **SCRUM** была представлена на общее обозрение задокументированной, четко сформированной и описанной совместно Швабером и **Джеф Сазерленд[en**] на OOPSLA’95[7] в Остине. 2001 г.«Scrum. Революционный метод управления проектами» Джеффа Сазерленда.

Слово scrum («схватка») автор позаимствовал из игры в регби. Оно «обозначает **метод командной игры**, позволяющий завладеть мячом и вести его дальше по полю, а для этого нужны **слаженность, единство намерений и четкое понимание цели.** „**Схватка“** представляет собой **идеальную модель полного взаимодействия игроков».** И это именно то, что требуется для успешной командной работы

Лекция 4
Модели процесса создания программного обеспечения
4.8.2 SCRUM. Планирование

- **Артефакты**

  Backlog - список работ, которые необходимо выполнить

  Backlog Sprint - набор требований, которые могут быть реализованы за один этап (sprint)
- **Спринт (Sprint)**

  30-тидневный (обычно) промежуток, за который выполняется реализация заданной функциональности
- **Планирование спринта**

  Происходит в начале спринта
- **SCRUM**

  Ежедневная встреча разработчиков
- **Демонстрация**

  Происходит в конце спринта

Лекция 4
Модели процесса создания программного обеспечения
4.8.3 SCRUM. Роли

- **Основные**

  Владелец продукта

  Руководитель (ScrumMaster)

  Команда (!)

- **Дополнительные**

  Пользователи

  Клиенты

  Эксперты-консультанты

Лекция 4
Модели процесса создания программного обеспечения
4.8.4 Методология SCRUM

- Заказчик определяет и периодически меняет функциональные требования
- Руководитель проекта расставляет приоритеты
- Формируются небольшие группы (1-6, реже до 9) человек для реализации небольших частей проекта
- Формируется backlog проекта
- Формируется sprint backlog для каждой группы
- Выполнение sprint происходит группой автономно.
- Руководитель не вправе влиять на sprint

Лекция 4
Модели процесса создания программного обеспечения
4.8.4 Методология SCRUM

- Каждая группа ежедневно выполняет схватки (scrum)- 10-30 мин.:

  Что сделано каждым в предыдущий день?

  Что будет сделано сегодня (в следующий день)?

  Что мешает работать или повышать производительность?

- Участвовать могут все, говорить только основные участники
- Задача руководителя группы – решать проблемы
- По окончании спринта – встреча с руководителями и заказчиками

Лекция 4
Модели процесса создания программного обеспечения
4.8.4 Методология SCRUM

Лекция 4
Технологические подходы к проектированию ПО. Итоги

	Классическая	Прототипирование	Спиральная	Инкрементная	RAD	RUP	XP	SCRUM
Стратегия	О	Э	Э	И	И	И+Э	Э	Э
Вид	Пр	Пр	Пр	Пр	Пр	Пр	Ад	Ад
Команда	1...∞	≤10	1...∞	1...∞	1...∞	1...∞	≤10	≤6
Продолжительность	Выс	Низк	Выс	Низк	Низк	Сред, Выс	Низк	Низк
Промеж.версия	-	-	+/-	+	+/-	+	+	-
ИС	-	-	-	-	+	+	+	-

Лекция 4.Задания

Предложите подходящую модель процесса создания ПО для разработки перечисленных ниже программных систем. Обоснуйте свое предложение.

1. Система управления торможением автомобиля.
2. Система поддержки процесса сопровождения программного обеспечения.
3. Университетская система учета и отчетности, которая должна заменить существующую систему.
4. Интерактивная система просмотра железнодорожных расписаний для пассажиров.
5. Поясните, почему программы, создаваемые в соответствии с эволюционной моделью разработки, трудны для сопровождения.
6. Опишите основные этапы процесса проектирования ПО, укажите выходной результат каждого этапа. С помощью диаграммы "сущность-связь" покажите возможные взаимосвязи между выходными результатами разных этапов процесса проектирования.
7. Назовите пять основных компонентов любых методов проектирования. Какие методы проектирования вы знаете? Опишите их компоненты. Оцените полноту этих методов.