Технологии разработки программного обеспечения

Виноградова Мария Валерьевна
Vinogradova.m@bmstu.ru

МГТУ им.Н.Э.Баумана
Кафедра СОИУ (ИУ5)

Контрольные мероприятия

- Семестр 1
  - 3 ДЗ
  - 2 РК
  - 7 ЛР
  - экзамен
- Семестр 2
  - Курсовая работа
- Задания, методички и варианты для всех КМ – http://iu5.bmstu.ru

Баллы (max/min)

- ДЗ = 6/4 (ДЗ-1); 10/6 (ДЗ-2); 6/4 (ДЗ-3)

  (+2 за срок и +2\+3\+2 за ответы в срок)

  => М = 22/14; Э = + (6 и 7)
- РК = 10/5 (РК-1); 10/5 (РК-2)

  (+2 за срок и +3 за ответы в срок)

  => М = 20/10; Э = + (4 и 6)
- ЛР = 4/2.5 (+1 за срок) – 7 шт.

  => М = 28/18; Э = + (7)
- Экзамен = 30/18
- Всего => 70/40 (+30)
  - 60=3, 71=4, 85=5

Зеленый коридор

- Все КМ (ЛР, ДЗ, РК) в срок
- Посещение всех лекций (очно)
- Если пропуск лекции, то ее защита

(не более 2-х пропусков)

Отчетность

- ЛР – очная защита + показ, отчет на почту
- ДЗ – очная защита + показ, отчет на почту и в бумажном виде
- РК – очно/удаленно + защита, отчет на почту и в бумажном виде

- Vinogradova.m.iu5@yandex.ru

Темы курса

- Методологии разработки ПО.
- Управление разработкой. Оценка стоимости, анализ рисков.
- CASE средства.
- RUP. Этапы и модели.
- Паттерны проектирования.
- Тестирование ПО.

Технологии разработки ПО

Технология разработки ПО – система инженерных принципов для создания экономичного ПО, которое надежно и эффективно работает на реальных компьютерах.

Технология определяется методами, средствами и процедурами:

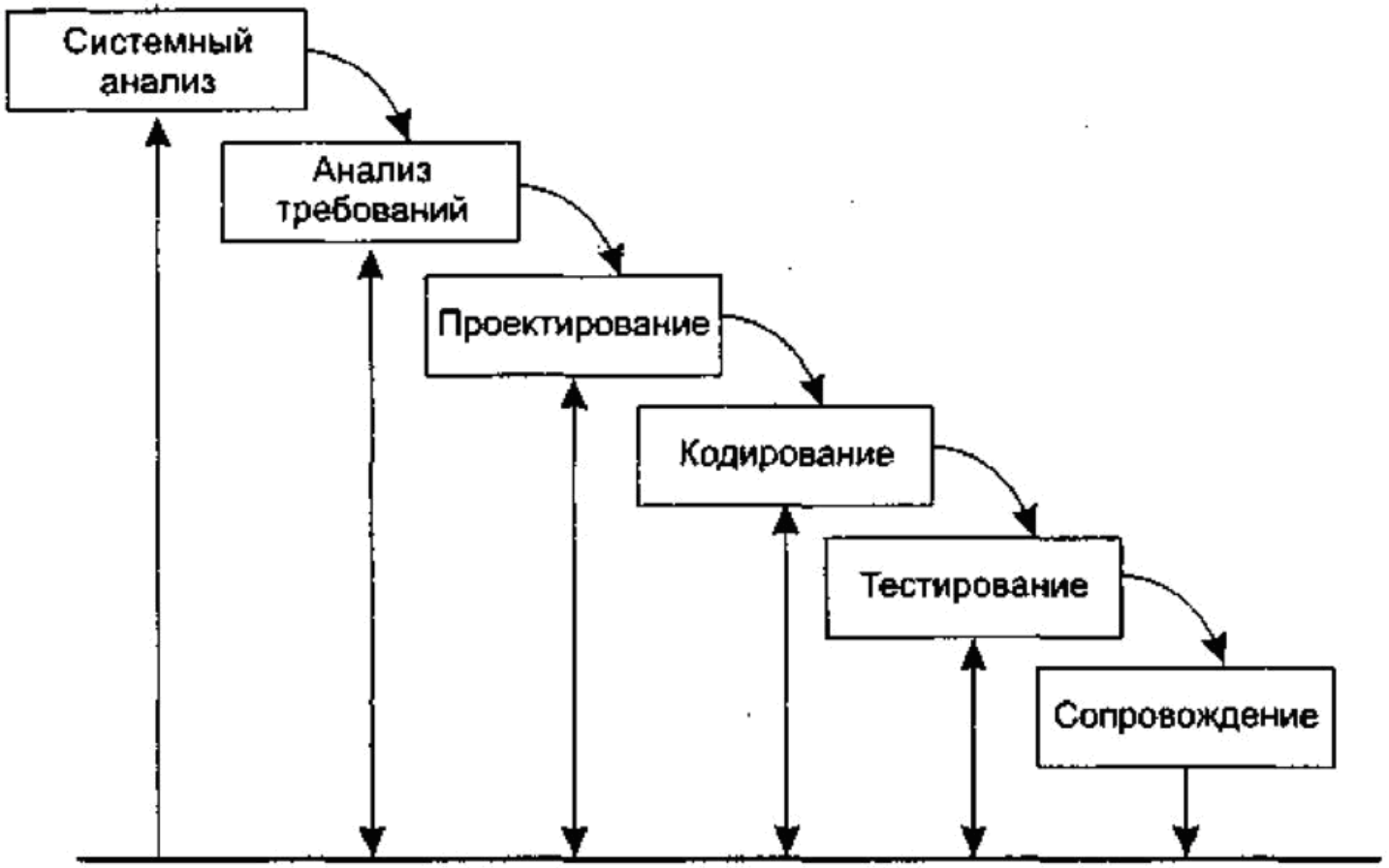
- **Методы** решают задачи разработки.
- **Средства** дают автоматическую или автоматизированную поддержку методов.
- **Процедуры** связывают методы и средства организационно.

Парадигмы (модели) разработки ПО – это последовательность шагов, использующих методы, средства и процедуры для разработки ПО.

Классический жизненный цикл

Каскадная (водопадная) модель;

Особенность - переход на следующий этап <u>после</u> завершения предыдущего.



Этапы ЖЦ каскадной модели

- Системный анализ
  - требования к системе в целом;
  - роли элементов в компьютерной системе, их взаимодействие;
  - распределение требований по элементам системы;
  - интерфейс ПО с внешней средой (аппаратное обеспечение, люди, БД);
  - оценка объема проектных работ, трудозатрат, рисков;
  - рабочие задачи и план-график работ.
- Анализ требований

  По каждому элементу ПО:
  - детализация требований;
  - функции, характеристики, интерфейс;
  - завершение планирование проекта.

  => спецификации анализа

Этапы ЖЦ каскадной модели

- Проектирование
  - архитектура ПО;
  - модульная структуры;
  - алгоритмы;
  - структуры данных;
  - входной /выходной интерфейс (формы данных);
  - оценка качество ПО.

  => множество проектных представлений
- Кодирование

  => составить текст (код) программы

Этапы ЖЦ каскадной модели

- Тестирование
  - выявление ошибок в функциях, логике, форме реализации ПО (через исполнение ПО)
- Сопровождение
  - внесение изменений в эксплуатируемое ПО для
    - удаления ошибок;
    - адаптации к изменениям внешней среды для ПО;
    - усовершенствования ПО по требованиям заказчика.

Особенности классического ЖЦ

Особенность - переход на следующий этап после завершения предыдущего.

Плюсы:

- Дает план и временной график по всем этапам;
- Упорядочивает ход конструирования.

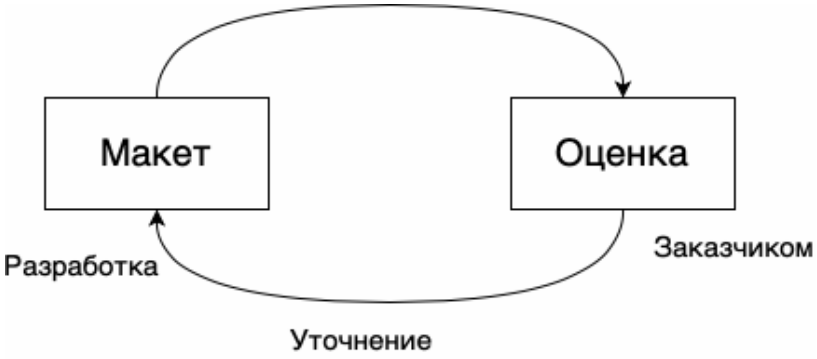
Минусы:

- Реальный проекты часто отклоняются от стандартной последовательности;
- Цикл основан на точных исходных требованиях, что нереально;
- Результаты доступны заказчику только в конце.

Макетирование (прототипирование)

Макетирование – многократный повтор, если

- заказчик не может сформулировать подробные требования по вводу/обработке/выводу данных;
- разработчик не может определить возможности ПО под ОС, эффективность алгоритма, форму диалога.



Этапы макетирования

1. Сбор и уточнение требований (исходных) – (разработчик + заказчик)
2. Быстрое проектирование – (разработчик)
   - наглядная демонстрация;
   - неэффективный алгоритм;
   - не та ОС, язык программирования и т.д.
3. Построение макета – (разработчик)
4. Оценка макета – (заказчик)
5. Если все в порядке, то к пункту 6, иначе уточнение требований и к пункту 2.
6. Разработка ПО.

Особенности макетирования

- Позволяет снять неопределенность в требованиях заказчика.

- Формы моделей:
  - бумажный макет (рис. человеко-машинный диалога);
  - работающий макет (реализация части функций ПО);
  - существующее ПО (которое должно быть улучшено).

- Достоинства:
  - Возможность определения полных требований к ПО.
- Недостатки:
  - Заказчик/разработчик могут принять макет за продукт.

Стратегии разработки ПО

- **Однократный** проход – *водопадная* модель

  (все исходные требования есть, 1 цикл конструирования, промежуточное ПО не используется)

- **Инкрементная** стратегия – **инкрементная** модель, **RAD**

  (все исходные требования есть, разбиение функций на версии, N циклов, последовательная реализация версий) – наращивание ПО.

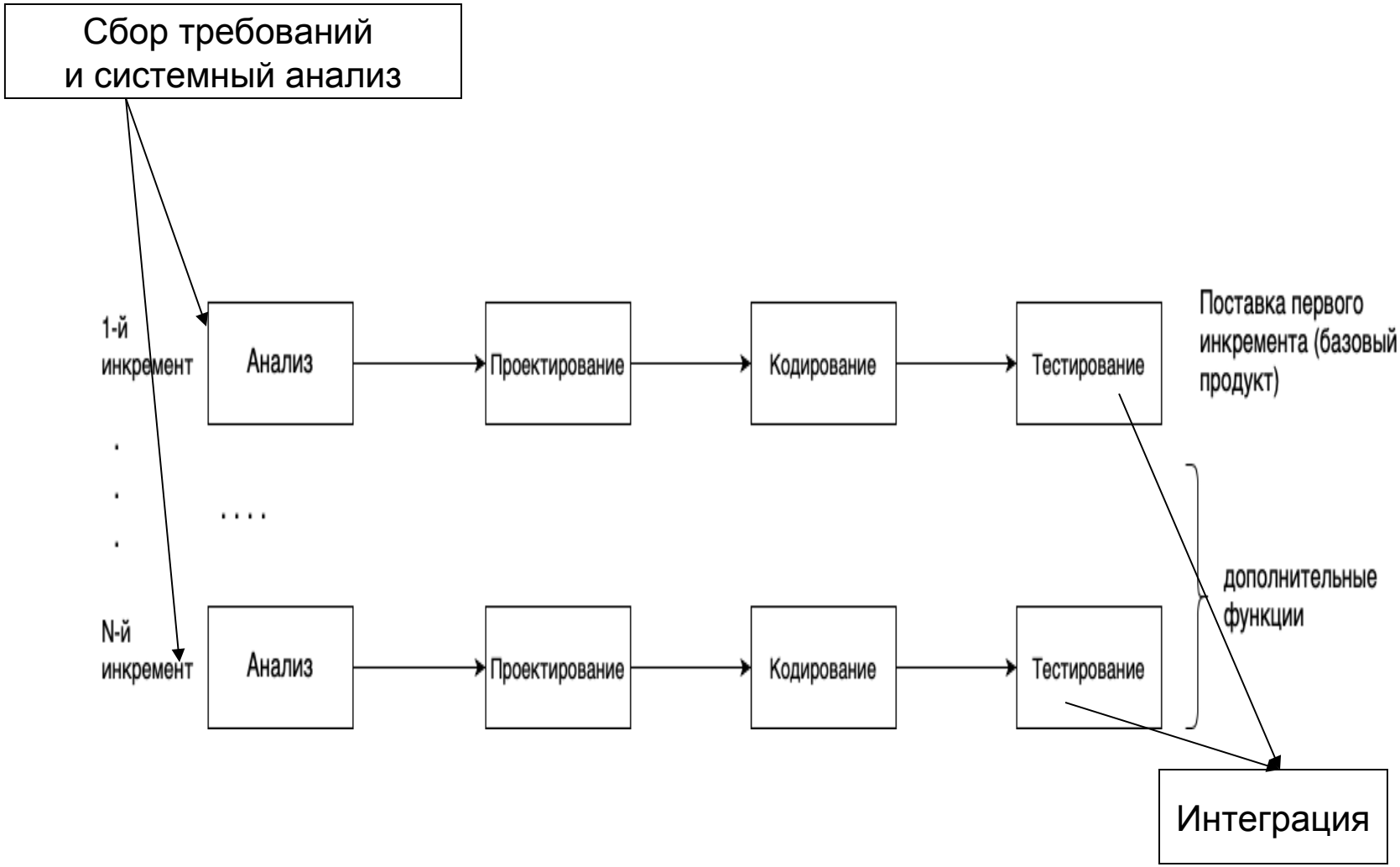
- **Эволюционная** стратегия – **спиральная** модель, **компонентно-ориентированная**

  (не все исходные требования определены, разработка версий – N циклов, промежуточное ПО, уточнение требований) – последовательная разработка с уточнением требований.

Инкрементная модель

- Объединяет последовательности и итерации
- Содержит этапы классического жизненного цикла
- Все исходные требования известны
- Разбиение функций системы на версии
- По каждой версии ее конструирование и запуск
- Каждый инкремент – это работающий продукт

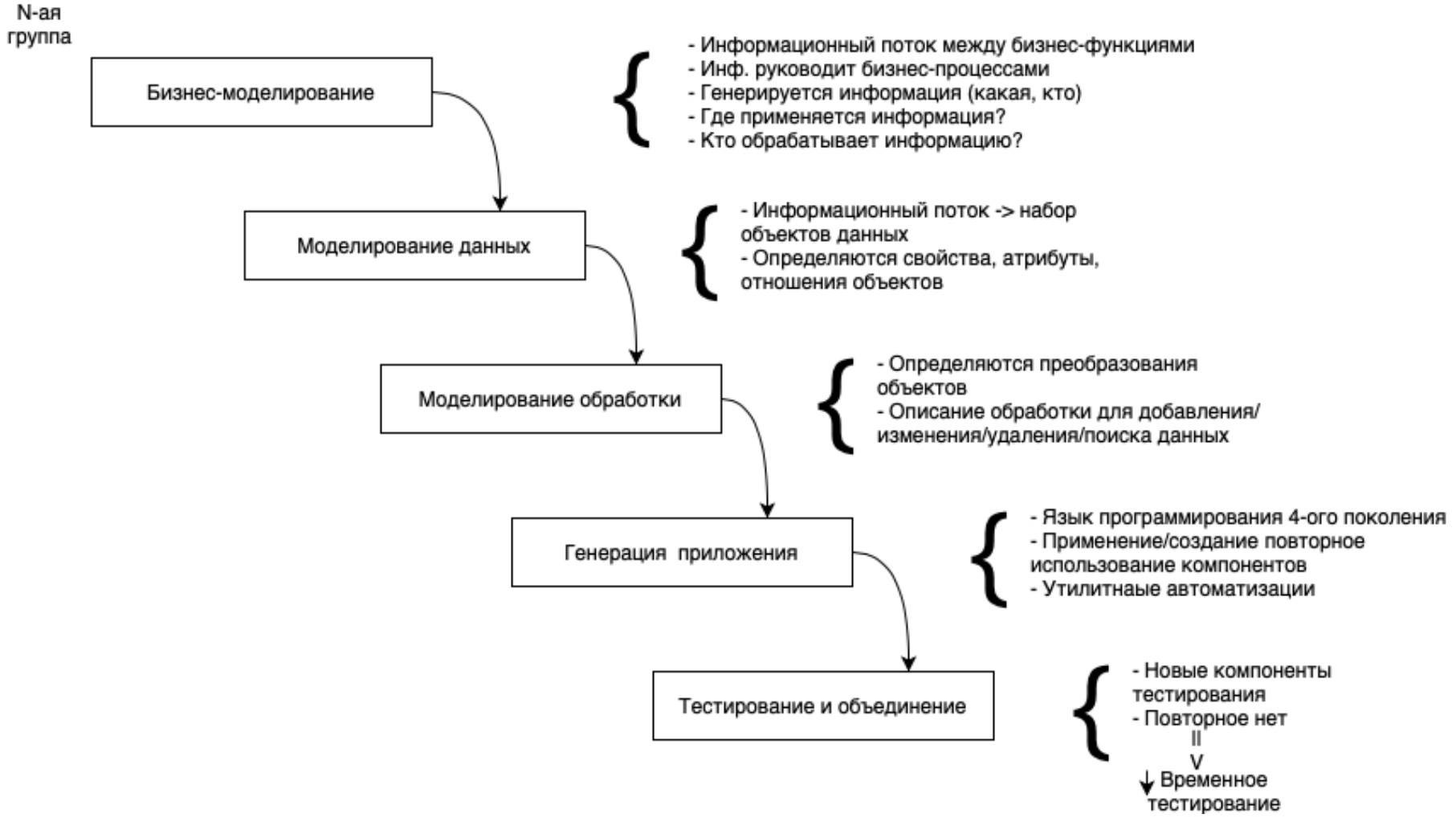
Этапы инкрементной модели



Модель быстрой разработки приложений RAD (Rapid Application Development)

- Особенности:
  - все исходные требования известны;
  - проектная область ограничена;
  - экстремально короткий цикл (высокоскоростная адаптация линейной последовательности модели);
  - компонентно-ориентированная разработка.
- Этапы:
  - Разбиение исходных требований на главные функции;
  - Каждая функция реализуется своей группой за 60-90 дней;
  - Интеграция главных функций.

Этапы RAD


N-ая группа
Бизнес-моделирование
- Информационный поток между бизнес-функциями
- Инф. руководит бизнес-процессами
- Генерируется информация (какая, кто)
- Где применяется информация?
- Кто обрабатывает информацию?
Моделирование данных
- Информационный поток -> набор объектов данных
- Определяются свойства, атрибуты, отношения объектов
Моделирование обработки
- Определяются преобразования объектов
- Описание обработки для добавления/ изменения/удаления/поиска данных
Генерация приложения
- Язык программирования 4-ого поколения
- Применение/создание повторное использование компонентов
- Утилитнаые автоматизации
Тестирование и объединение
- Новые компоненты тестирования
- Повторное нет
II
V
Временное тестирование

Этапы RAD

- Бизнес моделирование
  - Бизнес-функции
  - Информационный поток между бизнес-функциями
  - Информация, руководящая бизнес-процессами
  - Генерируется информация (какая, кто)
  - Где применяется информация?
  - Кто обрабатывает информацию?
- Моделирование данных
  - Информационный поток (набор объектов данных)
  - Определяются свойства, атрибуты, отношения объектов

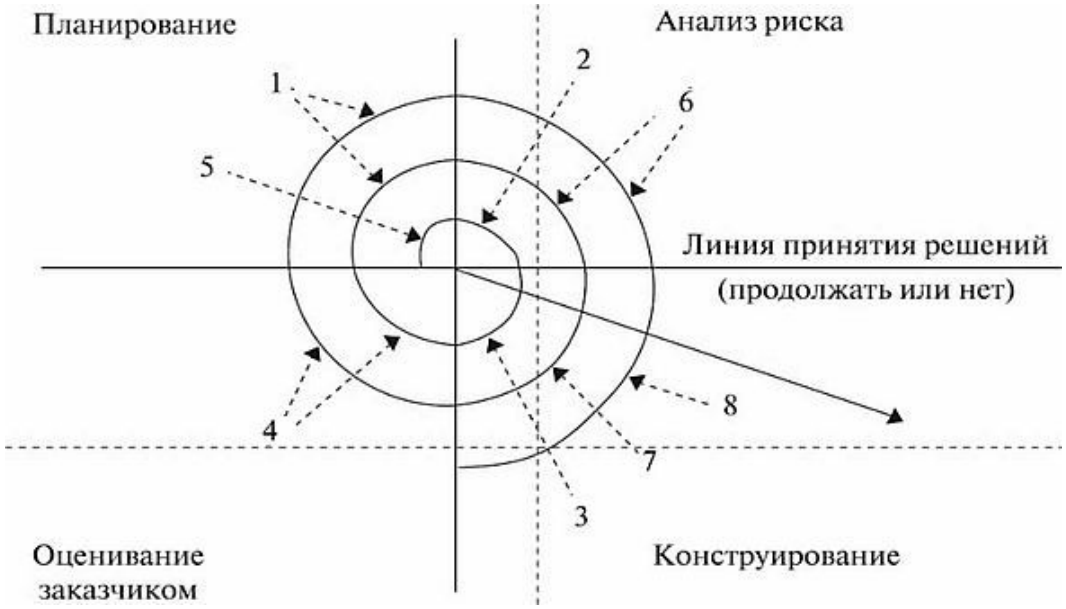
Этапы RAD - 2

- Моделирование обработки
  - Определяются преобразования объектов (алгоритмы)
  - Описание обработки для обавления/изменения/удаления/поиска данных
- Генерация приложения
  - Язык программирования 4 и 5 -х поколений (среда)
  - Применение/создание повторное используемых компонентов
  - Утилиты автоматизации
- Тестирование и объединение
  - Новые компоненты тестируются
  - Повторные нет (уменьшается время тестирования)

Ограничения RAD

- Для больших проектов необходимы большие человеческие ресурсы (много групп);
- RAD применима, если возможна декомпозиция на отдельные модули и производительность не является критической величиной;
- RAD не применима в условиях высоких технических рисков (новых технологиях)

Спиральная модель



- Б. Боэм, 1988
- Объединяет классический ЖЦ, макетирование, анализ рисков

Планирование – цели, варианты, ограничения
Анализ риска – анализ вариантов; распознавание/выбор риска
Конструирование – макетирование и/или разработка
Оценивание – оценка заказчиком текущего результата

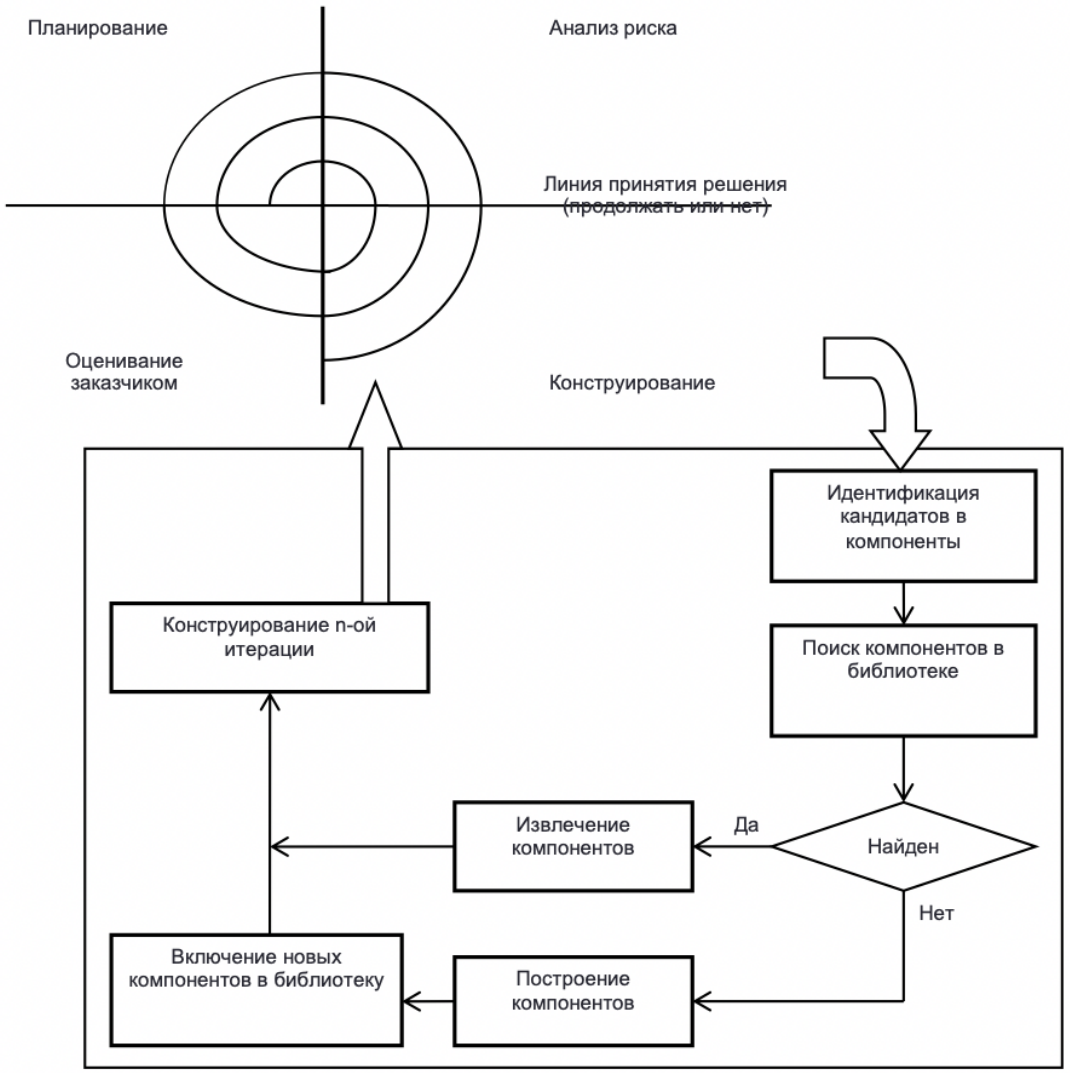
Этапы спиральной модели

- Сбор исходных требований
- Начальный анализ риска (неопределенные требования)
- Начальный макет
- Оценка и уточнение требований заказчиком (новые предложения)
- Уточнение требований
- Анализ риска с учетом реакции заказчика
- Макет, моделирование или промежуточное ПО
- Сконструированная система

Особенности спиральной модели

- Плюсы:
  - Реально (по эволюции) отражает разработку ПО;
  - Явный учет риска на каждом витке;
  - Включение системного подхода в итерационный подход;
  - Моделирование для снижения риска и повышения качества/эффективности ПО.
- Минусы:
  - Повышенные требования к заказчику;
  - Трудности контроля и управления временем разработки.

Компонентно-ориентированная модель



Особенности компонентно-ориентированной модели

- Особенности:
  - Повторное использование компонентов;
  - Развитие эволюционной модели.
- Плюсы:
  - Снижение на 30% времени разработки;
  - Снижение на 70% стоимости программной разработки;
  - Повышение на 50% производительности разработки.

Процессы разработки

- **Прогнозирующие** (тяжеловесные) – predictive / heavyweight
  - прогноз работ
  - порядок разработки
  - документирование
  - **Применимы при фиксированных требованиях и многих разработчиках разной квалификации**

- **Адаптивные** (подвижные, облегченные) - agile / lightweight
  - меньше документов
  - ориентированы на человека (разработчиков)
  - адаптируют изменения требований
  - **Применимы при частых изменениях требований,**
  - **малой группе высококвалифицированных разработчиков**
  - **и грамотных заказчиках, участвующих в разработке**

Основополагающие принципы Agile-манифеста

С 1990-х гг. Манифест – 2001 г.

1. Наивысшим приоритетом для нас является удовлетворение потребностей заказчика, благодаря регулярной и ранней поставке ценного программного обеспечения.

2. Изменение требований приветствуется, даже на поздних стадиях разработки. Agile-процессы позволяют использовать изменения для обеспечения заказчику конкурентного преимущества.

3. Работающий продукт следует выпускать как можно чаще, с периодичностью от пары недель до пары месяцев.

4. На протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе.

Основополагающие принципы Agile-манифеста

5. Над проектом должны работать мотивированные профессионалы. Чтобы работа была сделана, создайте условия, обеспечьте поддержку и полностью доверьтесь им.

6. Непосредственное общение является наиболее практичным и эффективным способом обмена информацией как с самой командой, так и внутри команды.

7. Работающий продукт — основной показатель прогресса.

8. Инвесторы, разработчики и пользователи должны иметь возможность поддерживать постоянный ритм бесконечно. Agile помогает наладить такой устойчивый процесс разработки.

Основополагающие принципы Agile-манифеста

9. Постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта.

10. Простота искусства минимизации лишней работы — крайне необходима.

11. Самые лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд.

12. Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы.

Приоритеты Agile

- Более важно:
  - Личности и их взаимодействия, ЧЕМ процессы и инструменты.
  - Работающее ПО, ЧЕМ полная документация.
  - Сотрудничество с заказчиками, ЧЕМ контрольные обязательства.
  - Реакция на изменения, ЧЕМ следование плану.

Экстремальное программирование (Extreme Programming, XP - процесс)

- К. Бек, 1999

- Игра планирования (planning game)

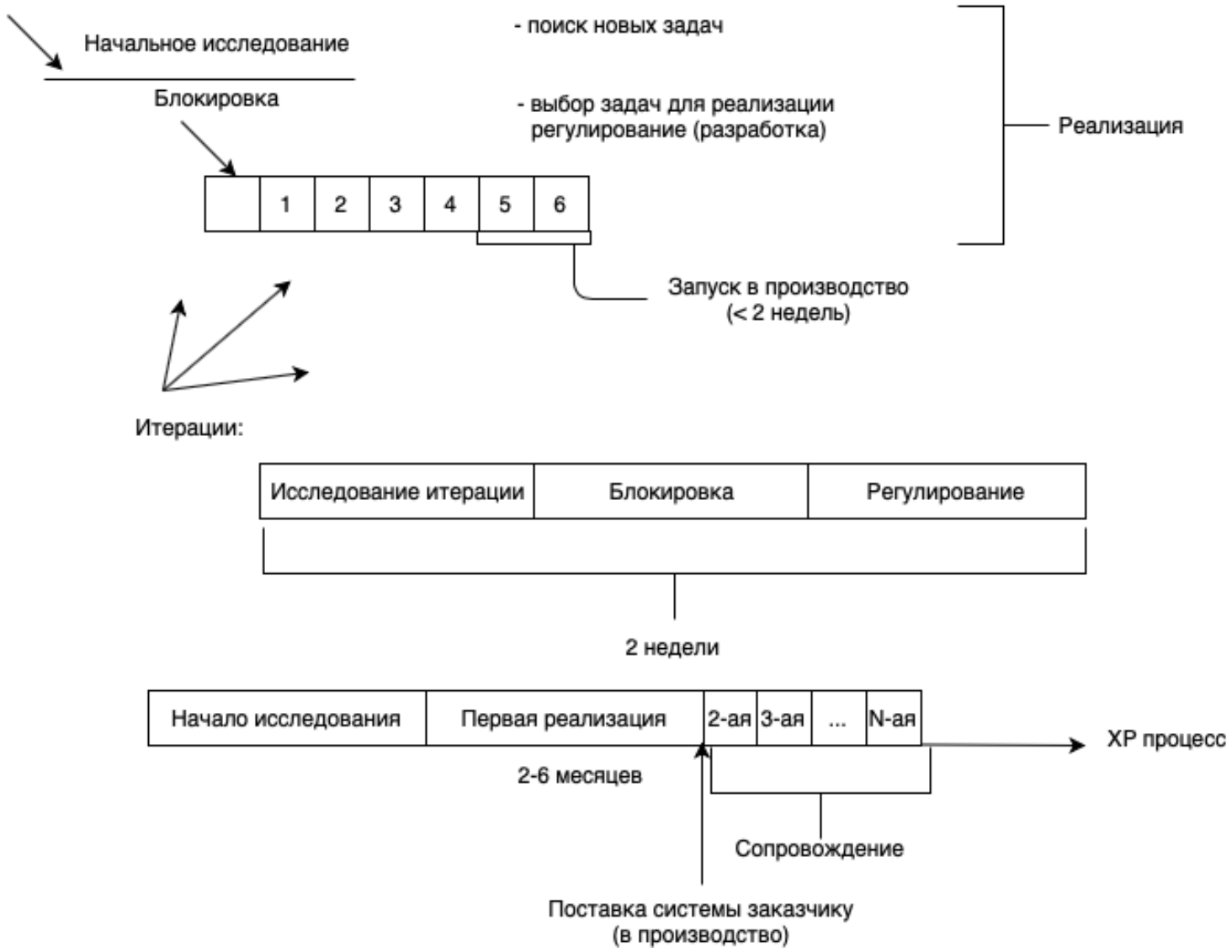
Шаги XP - процесса

1. Игра планирования (planning game)
2. Частая смена версий (small releases)
   - Запуск в производство новых версий каждые 2 недели
3. Метафора (Metaphor)
   - Простое, глобальное, общедоступное представление проекта.
4. Простое проектирование
   - Максимальная простота проектирования;
   - Минимум документации проектирования;
   - При остановке изменений требований – «нафталиновая документация» на 5-10 страниц.
5. Тестирование (Testing)
   - «Тестируй, потом кодируй»
   - Планирование тестирования (параллельно с анализом требований);
   - Тесты модулей непрерывно;
   - Тесты функций.
6. Реорганизация (Refactoring)
   - Постоянное проектирование с сохранением функциональности (удаление дублирований, добавление гибкости, упрощение)

Шаги XP - процесса

7. Парное программирование (Pair programming)
   - 2 программиста на 1 ПК;
   - Затраты повышаются на 15%;
   - Время уменьшается на 40-45%;
   - Повышается качество, снижаются затраты на сопровождение.
8. Коллективное владение кодом (Collective ownership)
   - Каждый разработчик может изменять любой код в любой момент.
9. Непрерывная интеграция (Continuous integration)
   - По каждой задаче (N раз в день);
   - Непрерывное регрессионное тестирование (повтор тестов).
10. 40-часовая неделя (40-hour week);
11. Локальный заказчик (On-site customer)
    - В группе;
    - Может отвечать на вопросы.
12. Стандарты кодирования (Coding standards)
    - Правила для одинакового кода во всех частях ПО.

Структура XP - процесса

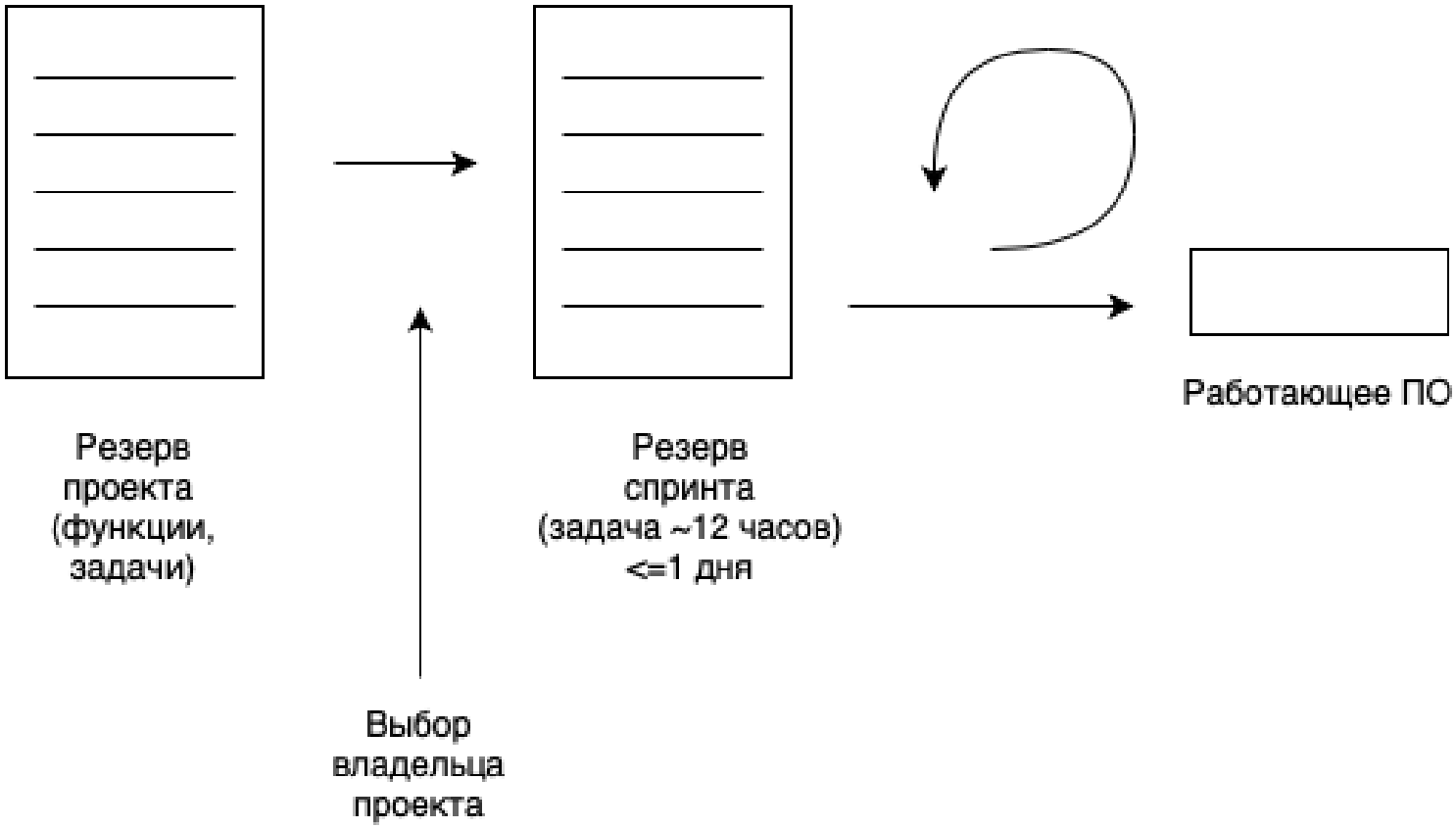


Особенности XP - процесса

- Управление XP проектом
  - через метрики (3-4 штуки), видимые всем.
  - основная метрика - «скорость проекта» – количество историй, реализуемых в итерации
- Плюсы:
  - Ориентирована на малые/средние группы (до 10 человек) при неопределенности/частом изменении требований;
  - Уменьшение стоимости внесения изменений.
- Минусы:
  - Обладает социальным воздействием;
  - Все методы необходимо исполнять совместно.

Scrum (схватка)

- Agile технология
- Выпуск (2-3 месяца)



Роли в Scrum

- Основные роли (свиньи, целиком в проекте):
  - Скрам-мастер – руководит процессом;
  - Владелец проекта;
  - Команда разработчиков 7+/-2 человека, разные специализации (кодировщик, тестировщик, архитектор и т.п.)
- Дополнительные роли (куры):
  - Пользователи;
  - Клиенты;
  - Управленцы;
  - Эксперты.

Термины Scrum

- ***Скрам*** – рассматриваемая методология управления проектами.

- ***Выпуск*** (release) – временной отрезок, в ходе которого над продуктом ведется работа. Фиксирован по времени. Имеет список задач, подлежащих решению в этот выпуск. Часто завершается выпуском новой версии продукта. Состоит из спринтов.

- ***Сприн***т (итерация) – временной отрезок, в ходе которого над продуктом ведется работа. Фиксирован по времени. Имеет список задач, подлежащих решению в этот спринт.

Термины Scrum

- ***Пожелание заказчика*** (user story**)** – функциональность, которую заказчик хочет реализовать в проекте.
- ***Резерв продукта (Product Backlog)*** – список пожеланий заказчика, которые необходимо реализовать в проекте, отсортирован по важности.
- ***Резерв выпуска(Release Backlog)*** – список пожеланий заказчика, которые необходимо реализовать в текущем выпуске.
- ***Резерв спринта(Sprint Backlog)*** – список пожеланий заказчика, которые необходимо реализовать в текущем спринте.
- ***Очки истории*** (Story Points) – абстрактные очки сложности пожеланий заказчика.
- ***График сгорания задач –*** график, показывающий оптимальный темп решения задач и реальный.

Этапы Scrum

1. Планирование
   - команда и владелец выбирают задачи,
   - команда обсуждает технические вопросы
2. Ежедневная встреча (15 минут, говорят свиньи)
   - результаты/планы/проблемы
   - скрам над скрамом приводит N команд
3. Демонстрация инкремента
   - Подводят итоги
   - время ограничено

Планирование спринта (Sprint Planning Meeting)

- Происходит в начале новой итерации Спринта.
  - Из резерва проекта выбираются задачи, обязательства по выполнению которых за спринт принимает на себя команда;
  - На основе выбранных задач создается резерв спринта. Каждая задача оценивается в идеальных человеко-часах;
  - Решение задачи не должно занимать более 12 часов или одного дня. При необходимости задача разбивается на подзадачи;
  - Обсуждается и определяется, каким образом будет реализован этот объём работ;
- Продолжительность совещания ограничена сверху 4-8 часами в зависимости от продолжительности итерации, опыта команды
- (первая часть совещания) Участвует владелец проекта и скрам команда: выбирают задачи из резерва продукта;
- (вторая часть совещания) Участвует только команда: обсуждают технические детали реализации, наполняют резерв спринта.

Ежедневное совещание (Daily Scrum meeting)

- начинается точно вовремя;
- длится не более 15 минут;
- проводится в одном и том же месте в течение спринта.

- В течение совещания каждый член команды отвечает на 3 вопроса:
  - Что сделано с момента предыдущего ежедневного совещания?
  - Что будет сделано с момента текущего совещания до следующего?
  - Какие проблемы мешают достижению целей спринта? (Над решением этих проблем работает скрам мастер. Обычно это решение проходит за рамками ежедневного совещания и в составе лиц, непосредственно затронутых данным препятствием.)

Обзор итогов спринта (Sprint review meeting)

- Проводится после завершения спринта.

- Команда демонстрирует инкремент функциональности продукта всем заинтересованным лицам.
- Привлекается максимальное количество зрителей.
- Все члены команды участвуют в демонстрации (один человек на демонстрацию или каждый показывает, что сделал за спринт).
- Нельзя демонстрировать незавершенную функциональность.

- Ограничена 4-мя часами в зависимости от продолжительности итерации и инкремента продукта.

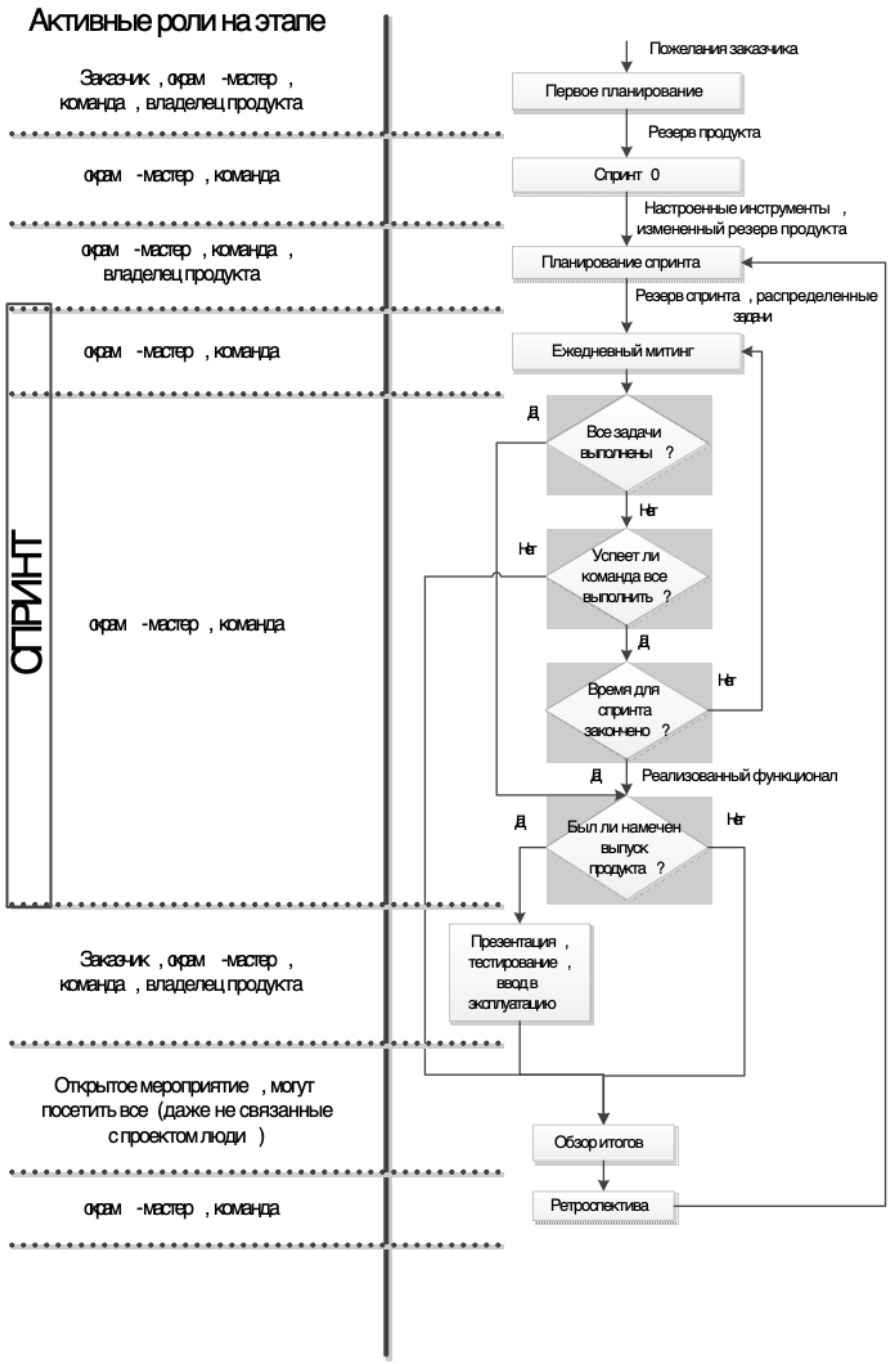
Ретроспективное совещание (Retrospective meeting)

- Проводится после завершения спринта.

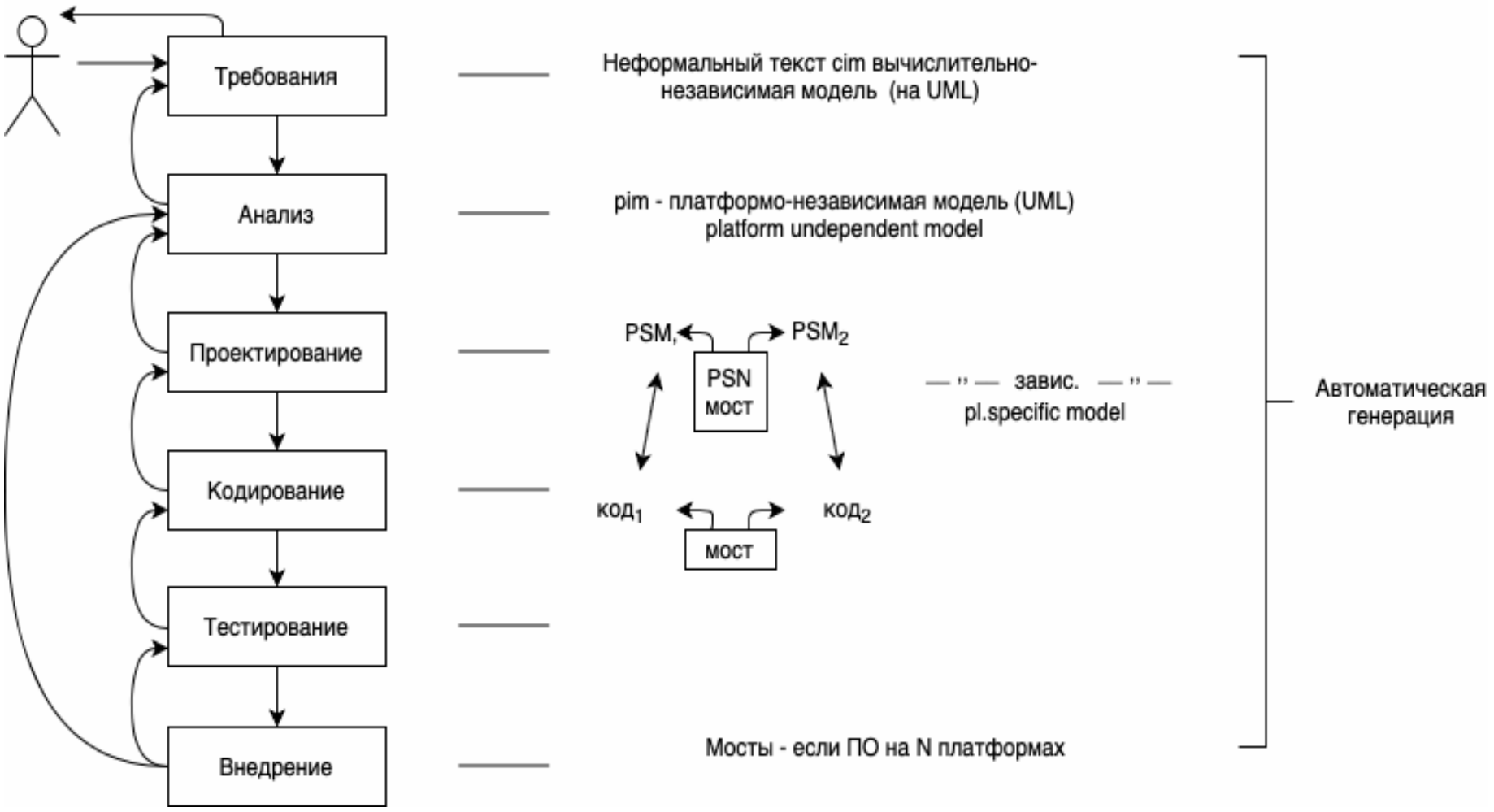
- Члены команды высказывают своё мнение о прошедшем спринте.

- Отвечают на два основных вопроса:
- Что было сделано хорошо в прошедшем спринте?
- Что надо улучшить в следующем?
- Выполняют улучшение процесса разработки (решают вопросы и фиксируют удачные решения).

- Ограничено 2-мя часами.


Активные роли на этапе
Заказчик , скрам -мастер , команда , владелец продукта
скрам -мастер , команда
скрам -мастер , команда , владелец продукта
скрам -мастер , команда
СПРИНТ
скрам -мастер , команда
Заказчик , скрам -мастер , команда , владелец продукта
Открытое мероприятие , могут посетить все (даже не связанные с проектом люди)
скрам -мастер , команда
Пожелания заказчика
Первое планирование
Резерв продукта
Спринт 0
Настроенные инструменты , измененный резерв продукта
Планирование спринта
Резерв спринта , распределенные задачи
Ежедневный митинг
Все задачи выполнены ?
Да
Нет
Успеет ли команда все выполнить ?
Нет
Да
Время для спринта закончено ?
Нет
Да
Реализованный функционал
Был ли намечен выпуск продукта ?
Да
Нет
Презентация , тестирование , ввод в эксплуатацию
Обзор итогов
Ретроспектива

Model Driven Architecture (MDA)



Особенности MDA

- MDA – последовательность преобразований моделей (спецификаций) систем.

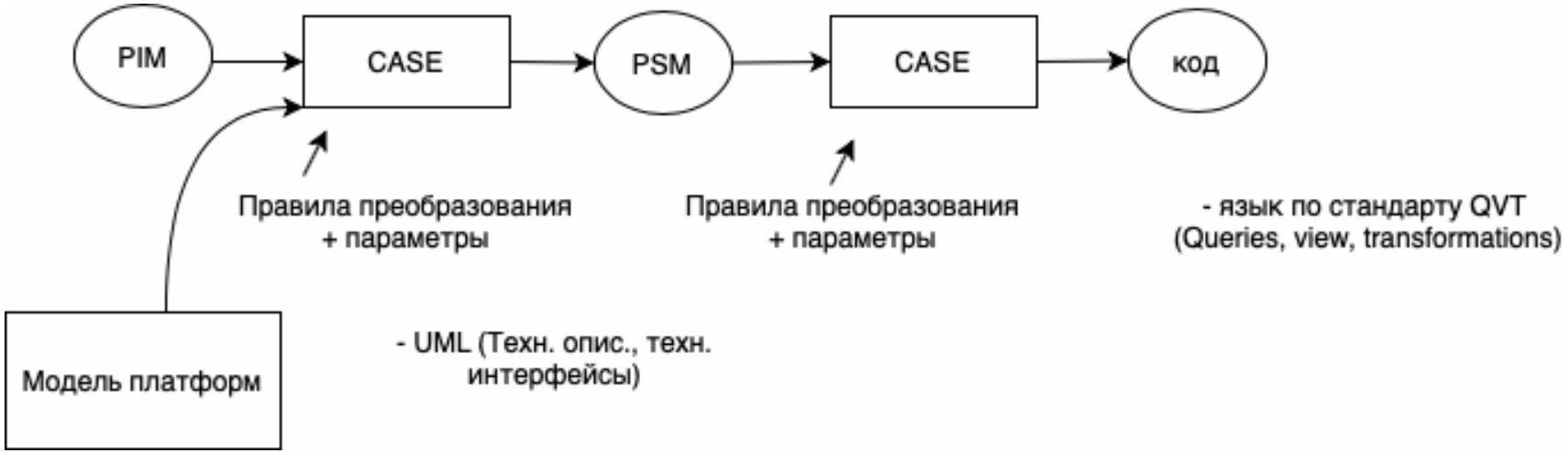
- Стандарт от OMG.
- Формальная разработка систем.
- «Выполняемая спецификация».
- Для автоматизации необходим CASE.
- Не все преобразования выполняются автоматически.

Языки MDA

- OMG не определяет язык PIM (может быть UML + OCL; профиль UML).
- Компонентно-ориентированная для конкретных предметных областей

- UML (PIM) -> UML (PSM) через XMI (XML для UML)

- QVT (Query/View/Transformation) – стандартный язык описания преобразования.



Модели качества процессов разработки ПО

- Качество процесса определяется стандартами.
- Самые авторитетные:
  - ISO 9001:2000 – процессы разработки
  - ISO/IEC 15504 – процессы программной разработки, с использованием CMM
  - CMM (Capability Maturity Model) – модель зрелости процесса конструирования (университет Карнеги-Меллон)

CMM (Capability Maturity Model)

- Определяет критерии оценки зрелости компании – разработки ПО
- Дает Сертификат зрелости;
- Дает рекомендации повышения зрелости.

  – Зрелость – наличие формальных процедур разработки и управления проектами.

Уровни зрелости CMM

- Начальный:
  - Процесс разработки/управления не формализован, случаен;
  - Нет плана и контроля;
  - Успех случаен, зависит от личности.
- Повторяемый:
  - Процесс планируется и контролируется;
  - Повтор достигнутых успехов.
- Определенный:
  - Все элементы процесса определены, стандартизированы, задокументированы;
  - Один стандарт в компании;
  - Качество ПО не зависит от личности.
- Управляемый:
  - Количественные показатели качества ПО и процесса;
  - Повышение точности планирования и контроля качества.
- Оптимизирующий:
  - Плановое и последовательное улучшение и повышение качества процесса создания и сопровождения ПО.

Уровни зрелости CMM и ОКП

- Каждый уровень состоит из области ключевых процессов (ОКП).
- ОКП N-ого уровня включает все ОКП нижних уровней.
- Для сертификата необходимо наличие всех ОКП уровня.
- Определяют по опросникам
- (Пример опросов для ОКП-5: предотвращение дефектов, управление изменениями технологии, управление изменениями процесса)

Спасибо за внимание!