Основы программной инженерии

Жизненный цикл ПО

2012

Определения

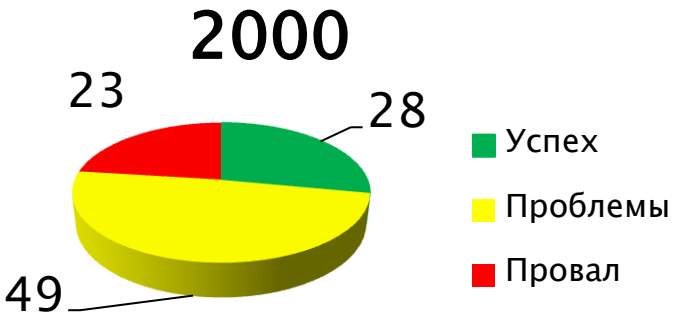
- Программное обеспечение
- Проектирование ПО
- Фаза проектирования ПО
- Жизненный цикл ПО
- Программный продукт

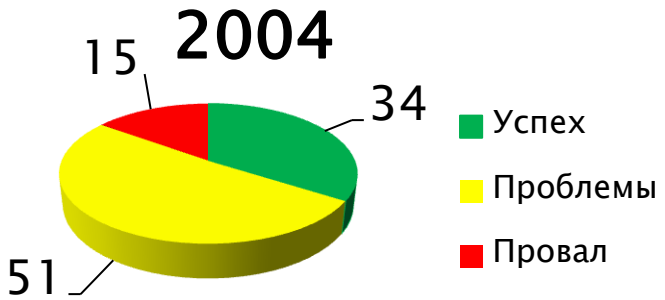
Основные фазы жизненного цикла ПО

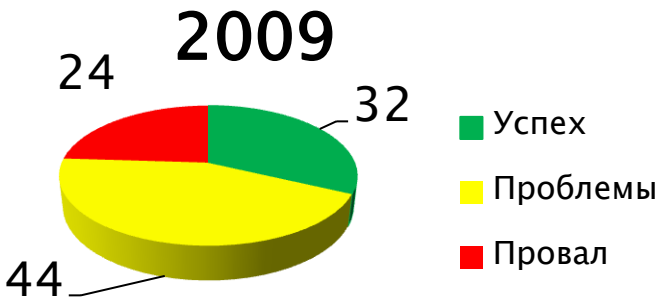
Критерии успешности проекта

- Качество
  - Реализованы все возможности
  - Эти возможности реализованы с надлежащим качеством
- Время
  - Проект реализован вовремя
  - Этапы проекта реализованы вовремя
- Бюджет
  - Проект уложился в планируемый бюджет

Статистика успешности проектов по разработке ПО







Источник: The Standish Group International, Chaos reports

Успешность программного проекта

- Программная индустрия существенно отличается от других областей производства:
  - Очень высокая сложность системы
  - Менее предсказуем результат
  - Хуже поддается планированию
  - До сих пор в большей степени творчество, чем ремесло

Что влияет на успешность программного проекта?

- Решаемая задача
- Заказчик
- Со стороны разработчика
  - Команда разработки
  - Инфраструктура
  - ***Выбранная методология проектирования ПО***

Методологии проектирования ПО

- Методологии определяются:
  - Составом и последовательностью работ
  - Ролью участников проекта
  - Составом и шаблонами документов
  - Организацией и управлением требованиями
  - Порядком контроля и проверки качества
  - Способами взаимодействия участников
  - …

Известные методологии проектирования ПО

- Agile software development
- Agile Unified Process (AUP)
- Behavior Driven Development (BDD)
- Big Design Up Front (BDUF)
- Constructionist design methodology (CDM)
- Design-driven development (D3)
- Design Driven Testing (DDT)
- Domain-Driven Design (DDD)
- Dynamic Systems Development Method (DSDM)
- Evolutionary Model
- Extreme Programming (XP)
- Feature Driven Development
- Iterative and incremental development
- Kaizen
- Kanban
- Lean software development
- Microsoft Solutions Framework (MSF)
- Model-driven architecture (MDA)
- Open Unified Process
- Rapid application development (RAD)
- Rational Unified Process (RUP)
- Scrum
- Software Craftsmanship
- Spiral model
- Structured Systems Analysis and Design Method (SSADM)
- Team Software Process (TSP)
- Test-driven development (TDD)
- Unified Process (UP)
- V-Model
- Waterfall model
- Wheel and spoke model

Что выбрать?

Характеристики методологий проектирования

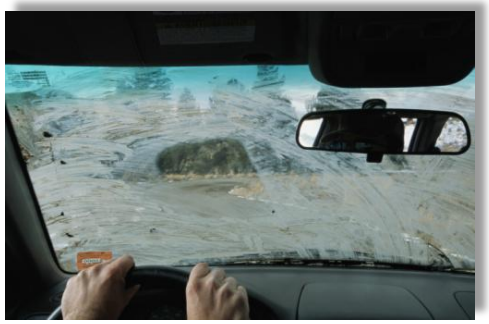
- Стратегия конструирования
- Адаптивность процесса
- Этапы и связи между ними
- Формулировка требований

Стратегии конструирования ПО

- Однократные
  - Определены все требования
  - Один цикл конструирования
  - Промежуточных версий нет
- Инкрементные
  - Иногда - инкрементно-итеративные
  - Определены все требования
  - Множество циклов конструирования
  - Промежуточные версии могут распространяться
- Эволюционные
  - Иногда - эволюционно-итеративные
  - Определены не все требования
  - Множество циклов конструирования
  - Промежуточные версии могут распространяются

Адаптивность процесса к окружению

- Тяжеловесные (прогнозирующие)
  - Фиксированные требования
  - Большая команда
  - Разная квалификация разработчиков
- Адаптивные (облегченные)
  - Постоянно меняющиеся требования
  - Маленькая команда
  - Высококвалифицированные разработчики



Рассматриваемые методологии

- Классическая (водопадная) модель
  - Общепринятая линейная модель
  - Классическая итерационная
  - Каскадная модель
  - Строгая каскадная модель
- Прототипирование (макетирование)
- Инкрементная модель
- Быстрая разработка приложений (RAD)
- Спиральная модель
- Rational Unified Process (RUP)
- Microsoft Solution Framework (MSF)
- Экстремальное программирование (XP)
- Scrum

Рассматриваемые методологии

	Стратегия	Адаптивность	Полнота
Классическая	Однокр.	Прогн.	+
Прототипирование	Эвол.	Прогн.	–
Спиральная	Эвол.	Прогн.	+
Инкрементная	Инкр.	Прогн.	+
RAD	Инкр.	Прогн.	+
RUP	Инкр. / Эвол.	Прогн.	+
MSF	Однокр. / Эвол.	Прогн./Адапт	+
XP	Эвол.	Адапт.	+
SCRUM	Эвол.	Адапт.	+

Как выбрать методологию проектирования?

- Выбор зависит от:
  - Решаемых задач
  - Сроков реализации
  - Команды разработчиков:
    - Размер команды разработчиков
    - Опыт команды разработчиков
    - Сработанность команды разработчиков
    - Местонахождение разработчиков
    - Механизмы взаимодействия в команде разработчиков
  - Заказчика:
    - Требований заказчика
    - Механизмов взаимодействия с заказчиком
  - и т.д.

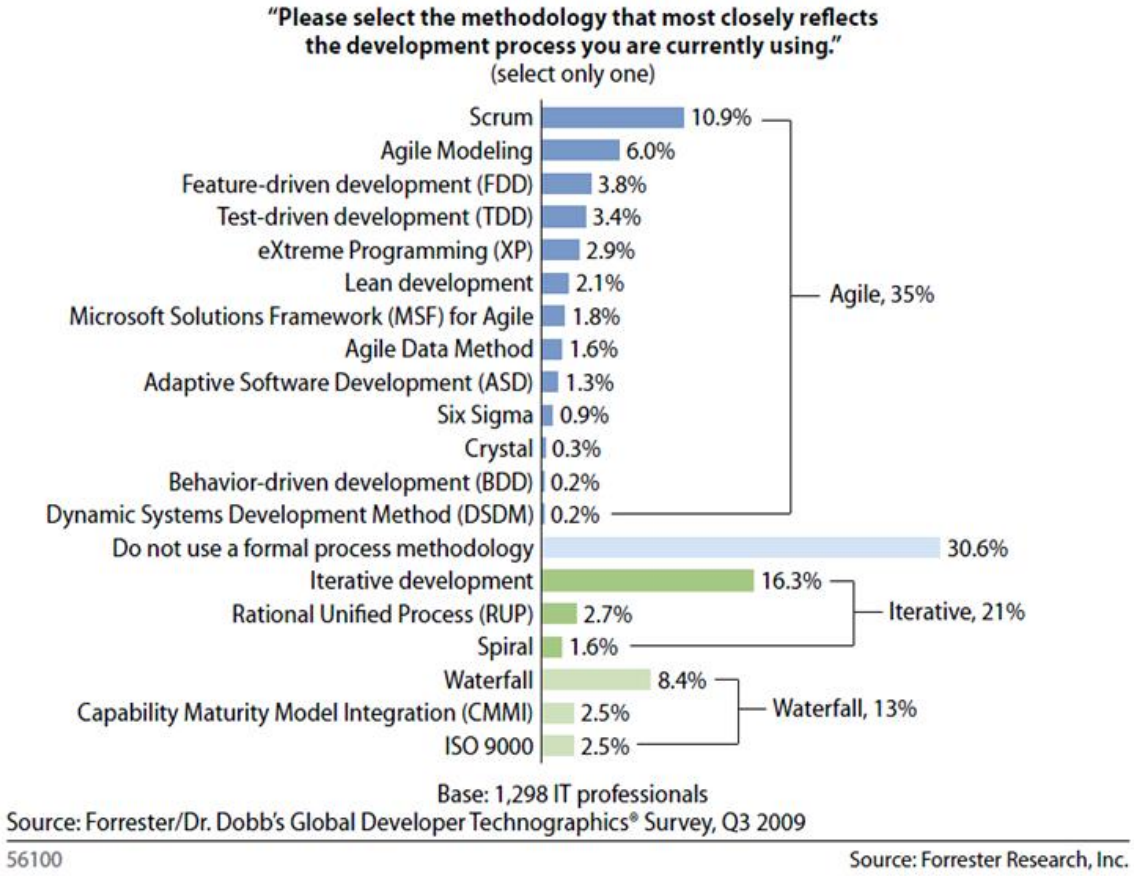
Чем отличаются различные методологии проектирования?

- Этапы
  - Список этапов
  - Последовательность этапов
  - Связи между этапами
  - Состав этапов
  - Объемы (длительность) этапов
- Требования
  - Формулировка требований
  - Корректировка требований
- Команда
  - Размер
  - Роли участников проекта
  - Способами взаимодействия участников

Чем отличаются различные методологии проектирования?

- Артефакты
  - Состав и содержание
  - Время получения артефактов (например, версий)
  - Объем и состав документации
- Заказчик
  - Квалификация заказчика
  - Степень участия заказчика
- Порядок контроля и проверки качества

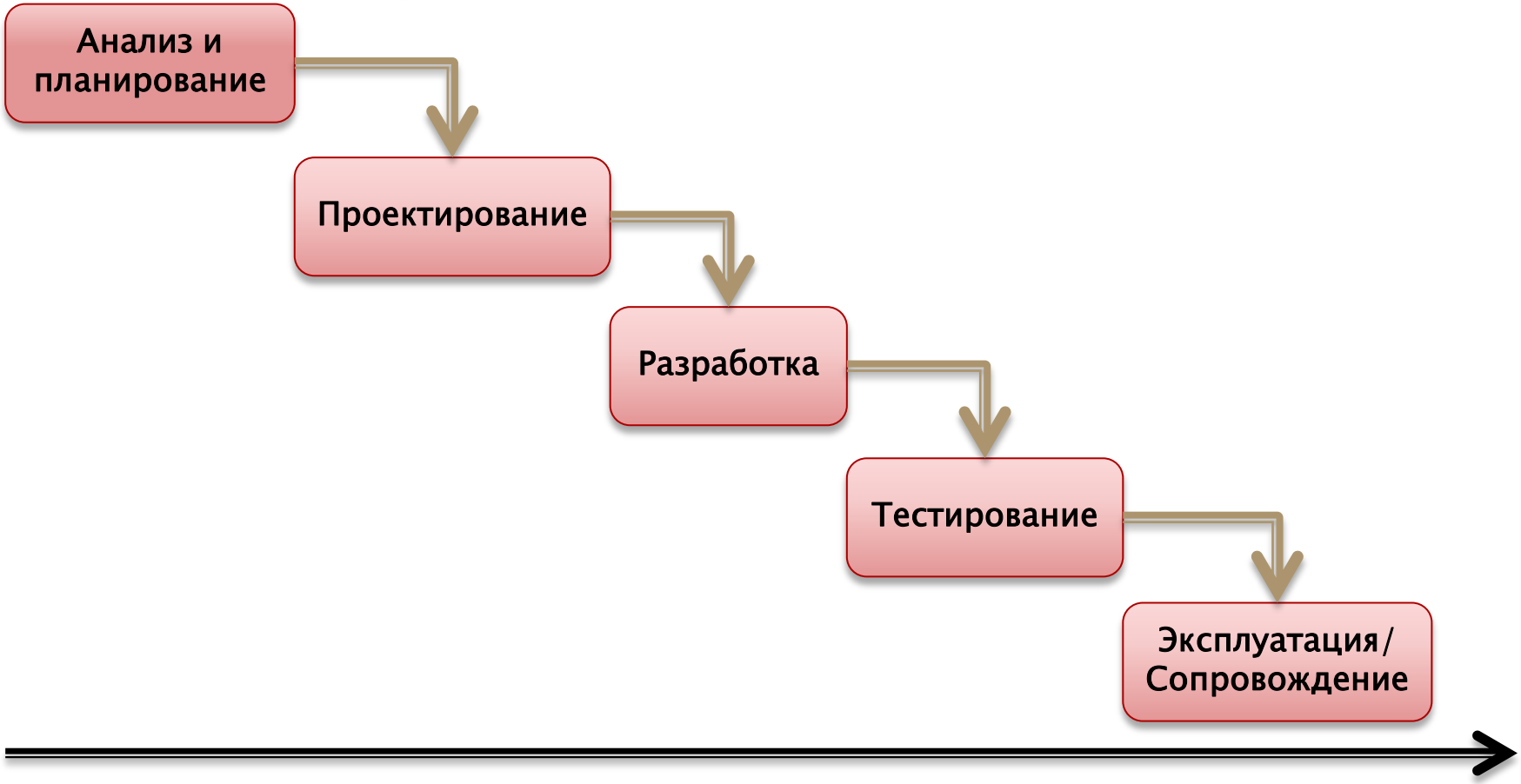
Статистика использования методологий



Классическая модель проектирования ПО

- Предложена в 1960-х годах, впервые описана 1970 г., В. Ройсом
- Водопадный (однократный) подход
- Относится к прогнозирующим методологиям
- Предполагает полное наличие всех требований на момент старта проекта
- Требования не могут меняться в процессе проектирования
- Программный продукт появляется по окончании проектирования
- Промежуточные версии не предусмотрены

Классическая модель проектирования ПО



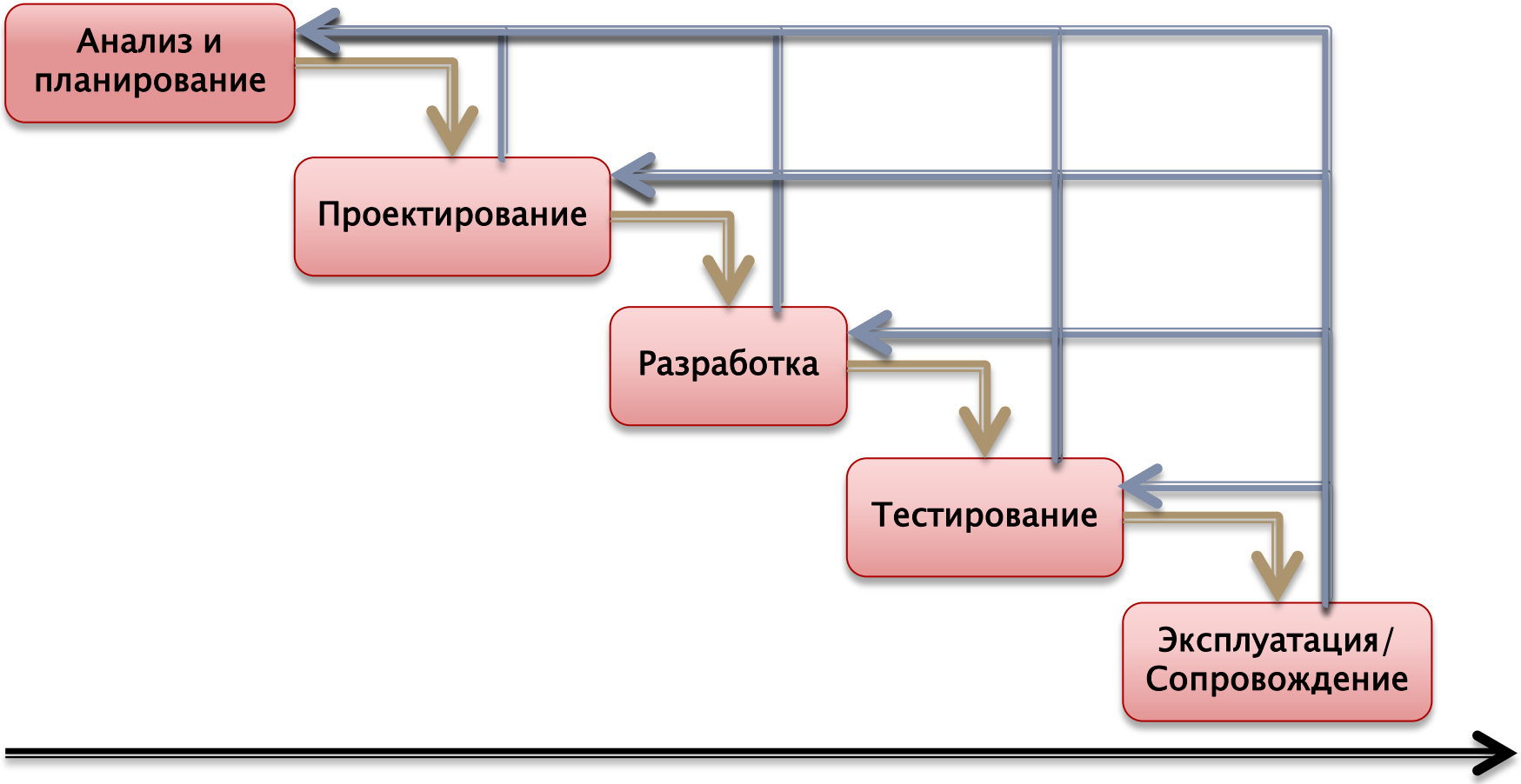
Классическая модель проектирования ПО

- Анализ и планирование
  - Сбор требований
  - Анализ требований
  - Планирование проекта
- Проектирование
  - Разработка архитектуры
  - Разработка моделей данных
  - Разработка алгоритмов
- Реализация
  - Кодирование
  - Отладка
- Тестирование/верификация
- Сопровождение
  - Внедрение
  - Эксплуатация
  - Внесение изменений

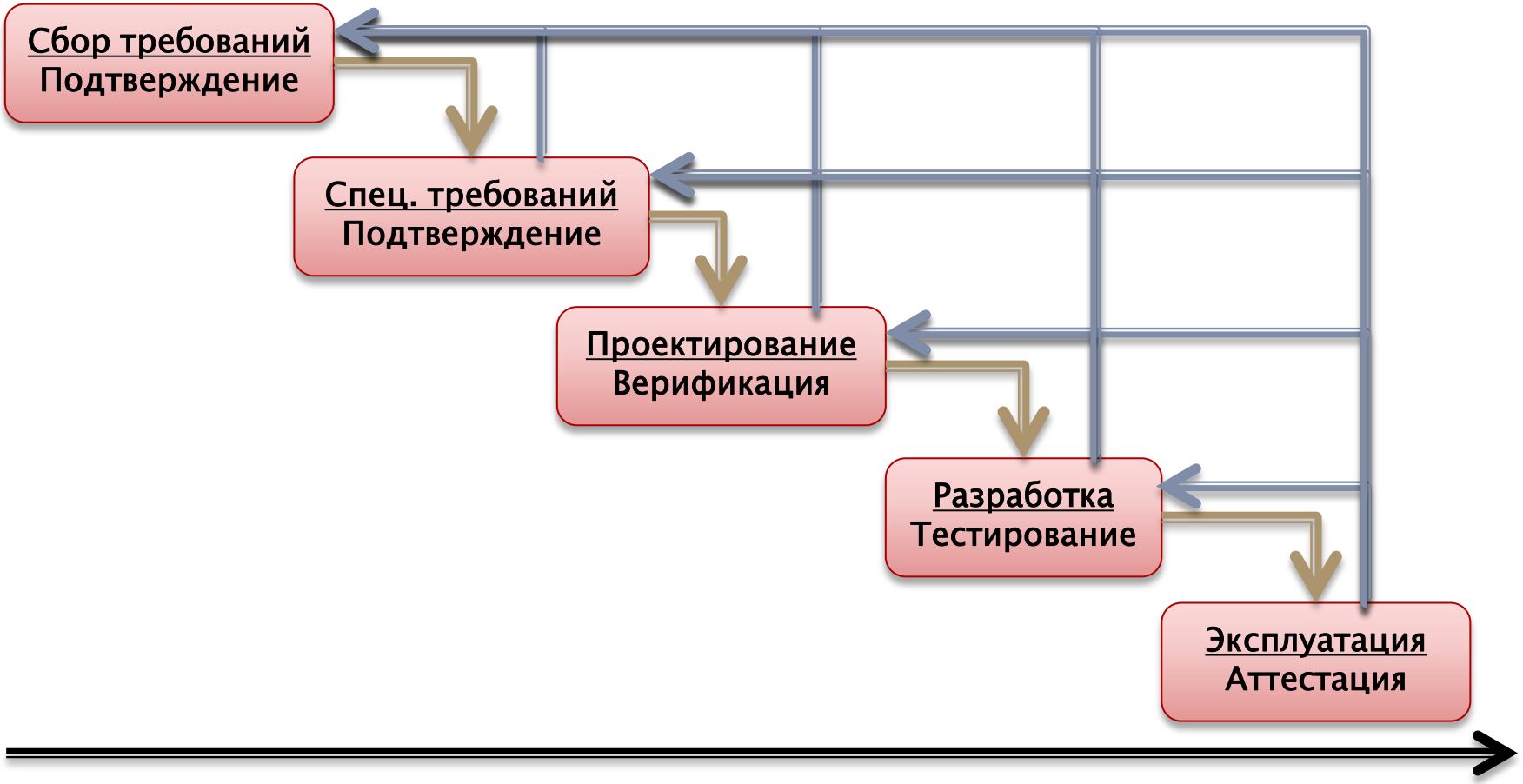
Классическая модель проектирования ПО

- Имеется несколько модификаций
  - Общепринятая линейная модель
  - Классическая итерационная
    - Предложена В. Ройсом, 1970 г.
    - Обратная связь после каждого этапа
  - Каскадная модель
    - Завершение каждого этапа проверкой
  - Строгая каскадная модель
    - Минимизация возвратов к пройденным этапам

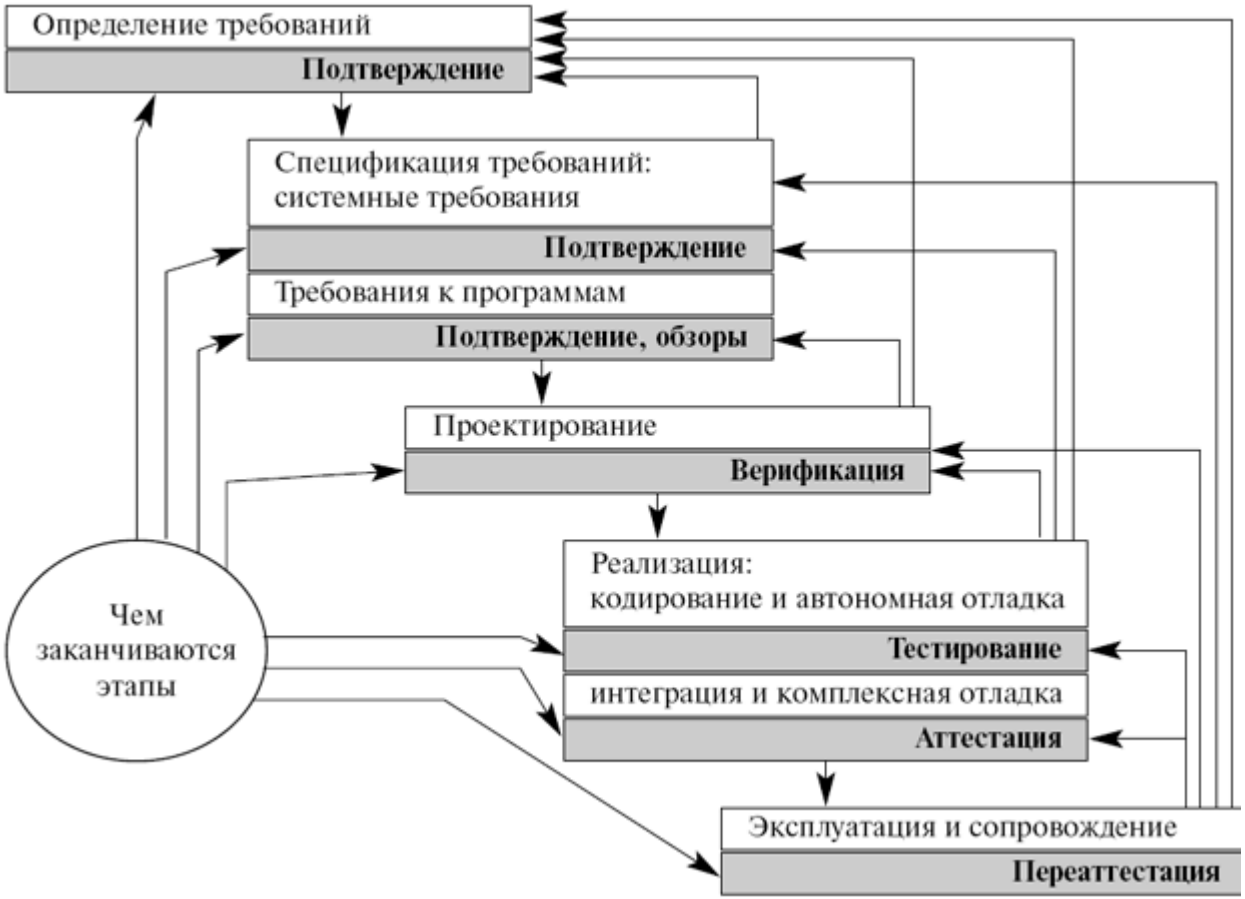
Классическая итерационная модель ППО



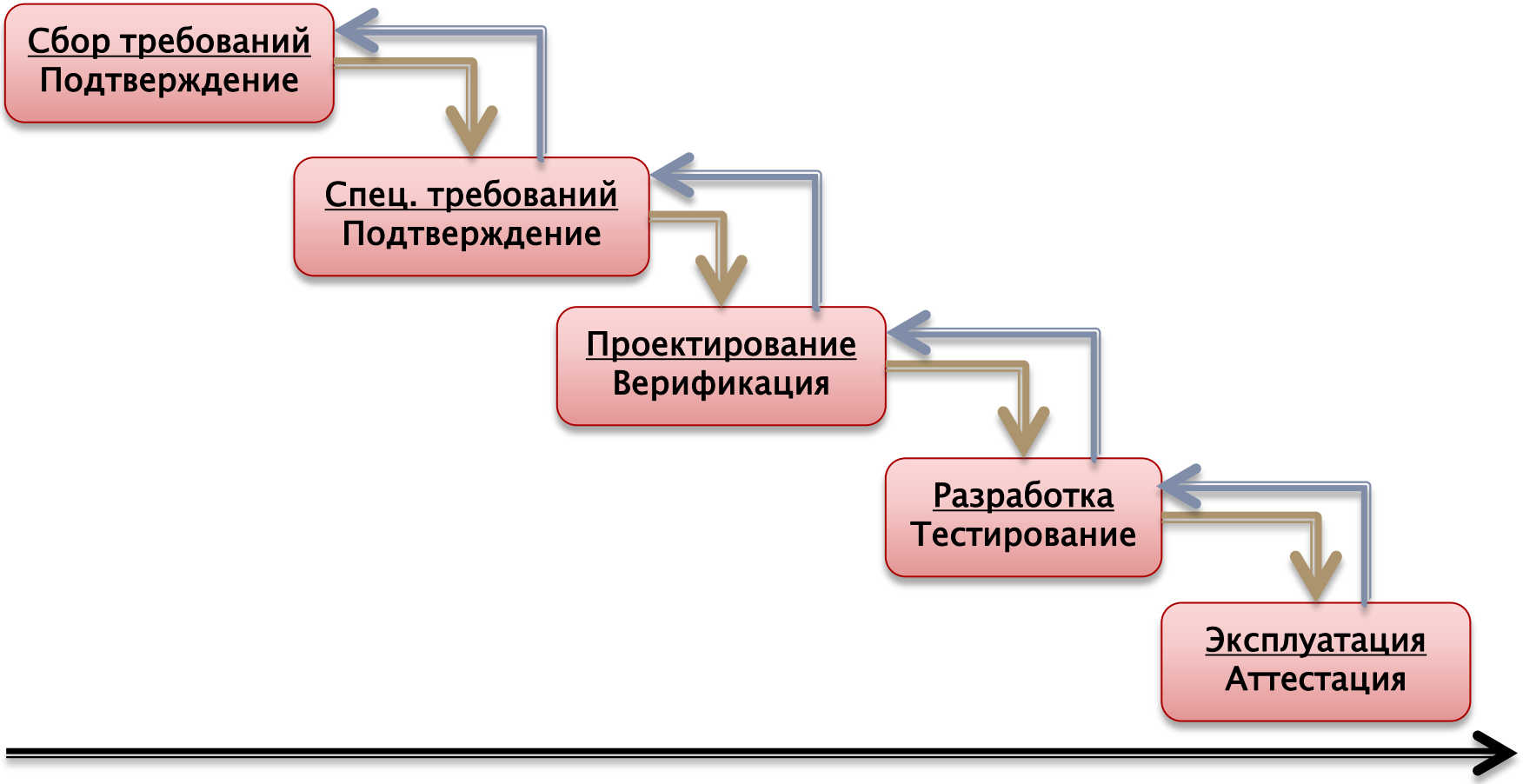
Каскадная модель



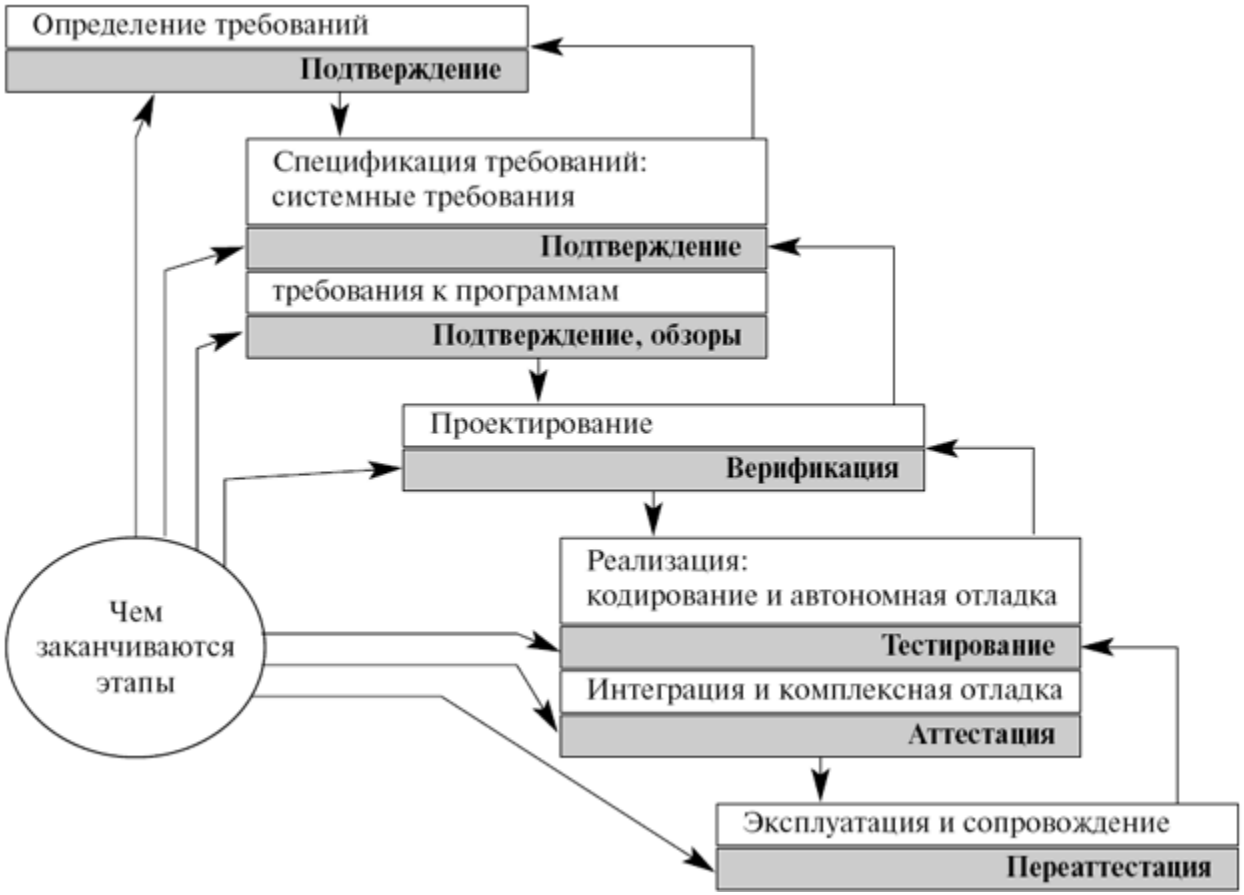
Каскадная модель



Строгая каскадная модель



Строгая каскадная модель ППО

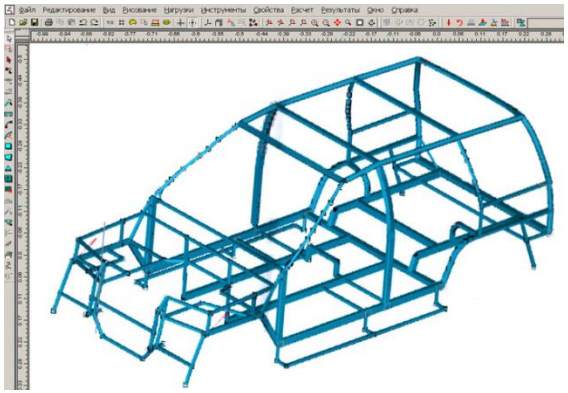


Классическая модель проектирования ПО

- Достоинства:
  - Имеется план и график по всем этапам конструирования
  - Ход конструирования – упорядочен
  - Имеется богатый опыт использования
- Недостатки:
  - Не всегда соответствует реальным проектам (отсутствует гибкость)
  - Часто всех требований на начальном этапе нет
  - Результат доступен только в конце

Прототипирование (макетирование)

- Применятся, когда имеются не все требования
- Позволяет быстро увидеть некоторые свойства продукта
  - Удобство
  - Внешний вид
  - Применимость
- Часто применятся при проектировании
  - Информационных систем
  - Программных продуктов с ГПИ
- Используются средства быстрой разработки приложений



Прототипирование

1. Сбор и уточнение требований
2. Быстрое проектирование
3. Построение макета
4. Оценка макета заказчиком
   - Заказчик не удовлетворен
     - Уточнение требований
     - Переход к п. 2
   - Заказчик удовлетворен
     - Переход к п. 5
5. Конструирование продукта

Прототипирование

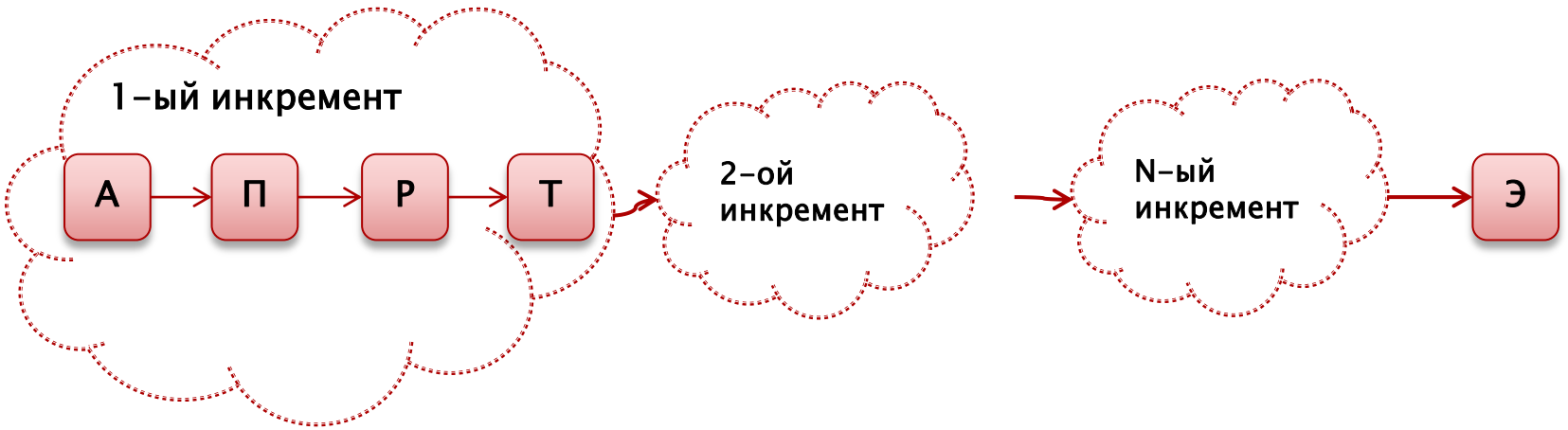
Прототипирование

- Достоинства:
  - Обеспечивает определение полных требований к ПО
- Недостатки:
  - По сути не является полным ЖЦ
  - Заказчик может принять макет за продукт
  - Разработчик может принять макет за продукт

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

Инкрементная модель

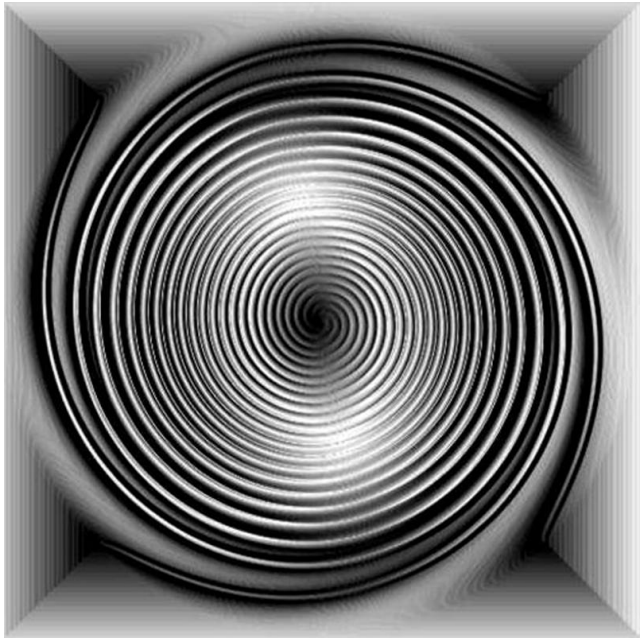


Инкрементная модель

- Достоинства:
  - Имеется план и график по всем этапам конструирования
  - Промежуточные версии доступны заказчику
- Недостатки:
  - Часто всех требований на начальном этапе нет
  - Не всегда можно заранее спланировать содержание версий
  - Отсутствует гибкость

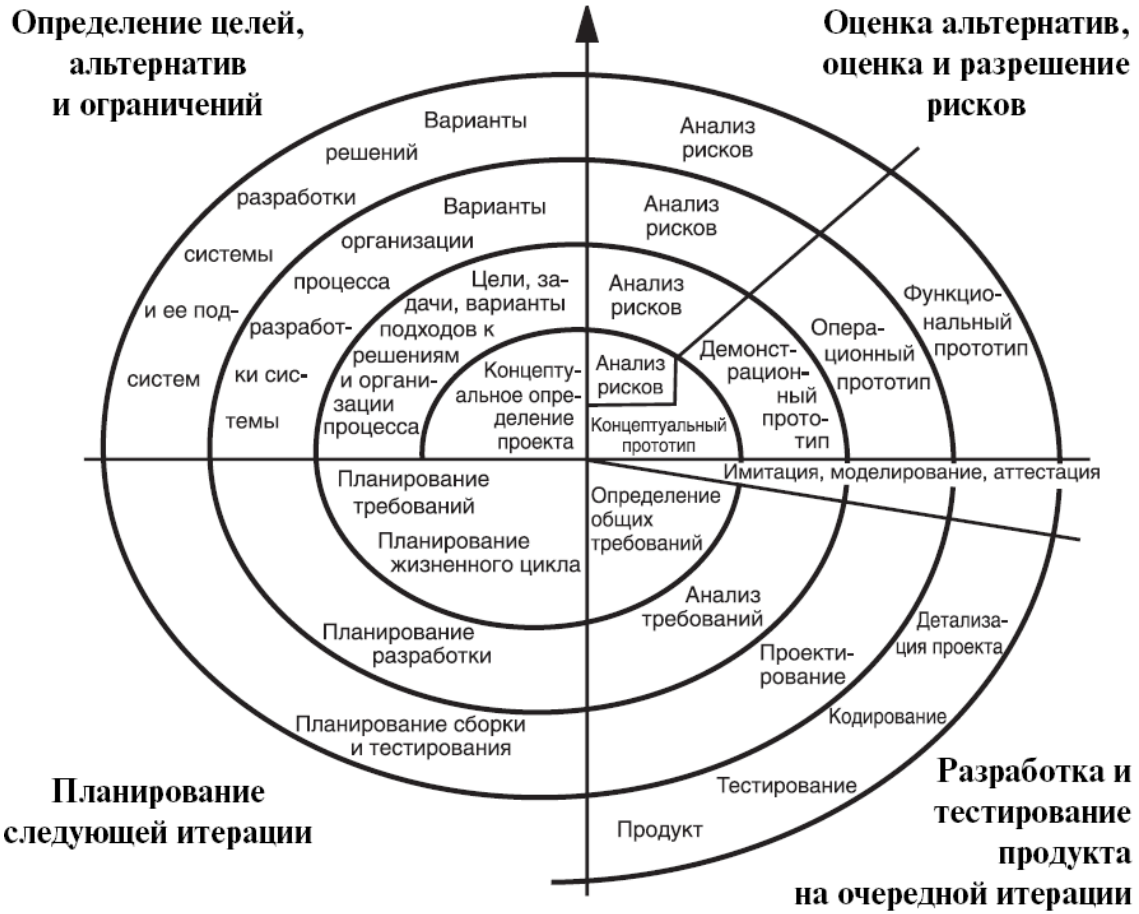
Спиральная модель

- Предложена Б. Боемом, 1988г
- Базируется:
  - На классическом ЖЦ
  - На макетировании
- Дополнена анализом рисков
- Основные компоненты
  - Планирование
  - Анализ
  - Конструирование
  - Оценивание



Спиральная модель

Инструментальная спиральная модель



Спиральная модель ППО

- Достоинства:
  - Адекватно отражает эволюционный характер проектирования
  - Позволяет явно учитывать риски на каждом витке эволюции
  - Использует моделирование
- Недостатки:
  - Высокие требования к заказчику
  - Трудность контроля времени разработки и управления им

Быстрая разработка приложений

Быстрая разработка приложений (RAD)

RAD = Rapid Application Development

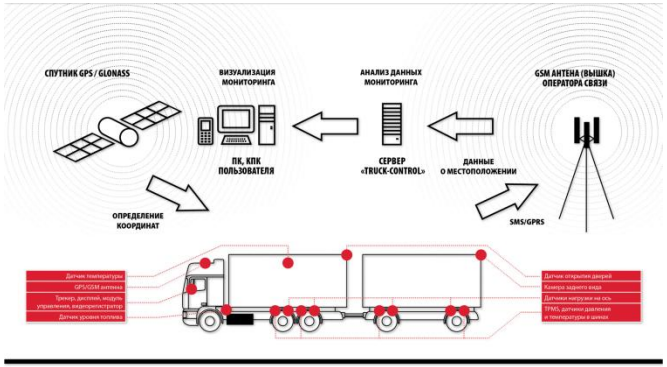
- Инкрементная стратегия конструирования
- Использование компонентно-ориентированного конструирования
- Обеспечение очень короткого цикла разработки (60-90 дней)
- Ориентирована в основном на разработку ИС

RAD. Основные этапы

- Бизнес-моделирование
- Моделирование данных
- Моделирование обработки
- Генерация приложения
- Тестирование и объединение

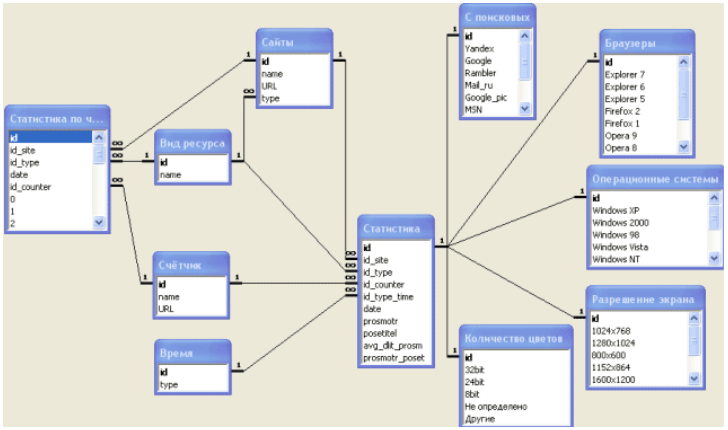
RAD. Бизнес-моделирование

- Моделируется информационный поток между бизнес-функциями
- Определяется:
  - Какая информация создается
  - Кто ее создает
  - Кто ее обрабатывает
  - Где информация применяется



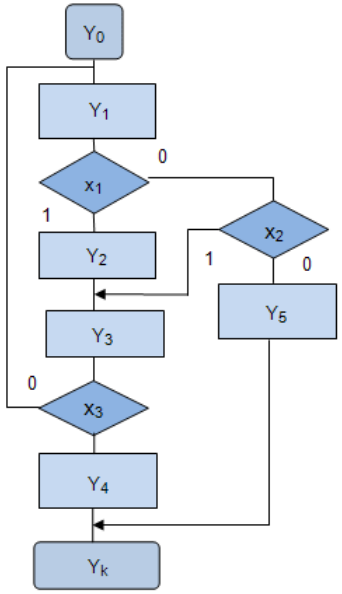
RAD. Моделирование данных

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



RAD. Моделирование обработки

- Определение преобразований объектов данных
- Создаются описания для
  - добавления объектов данных
  - модификации объектов данных
  - удаления объектов данных
  - поиска объектов данных



RAD. Генерация приложения

- Использование ЯП 4-го поколения
- Использование готовы компонентов
- Создание повторно используемых компонентов
- Использования средств автоматизации

RAD. Тестирование и объединение

- Тестирование упрощается из-за повторного использования компонентов
  - Они не требуют автономного тестирования
- Используется интеграционное тестирование

RAD. Ограничения

- Область применения – проектирование информационных систем
- Производительность не является критичной
  - Неприменимо для задач реального времени
- Можно привлечь достаточно разработчиков
- Отсутствуют технические риски

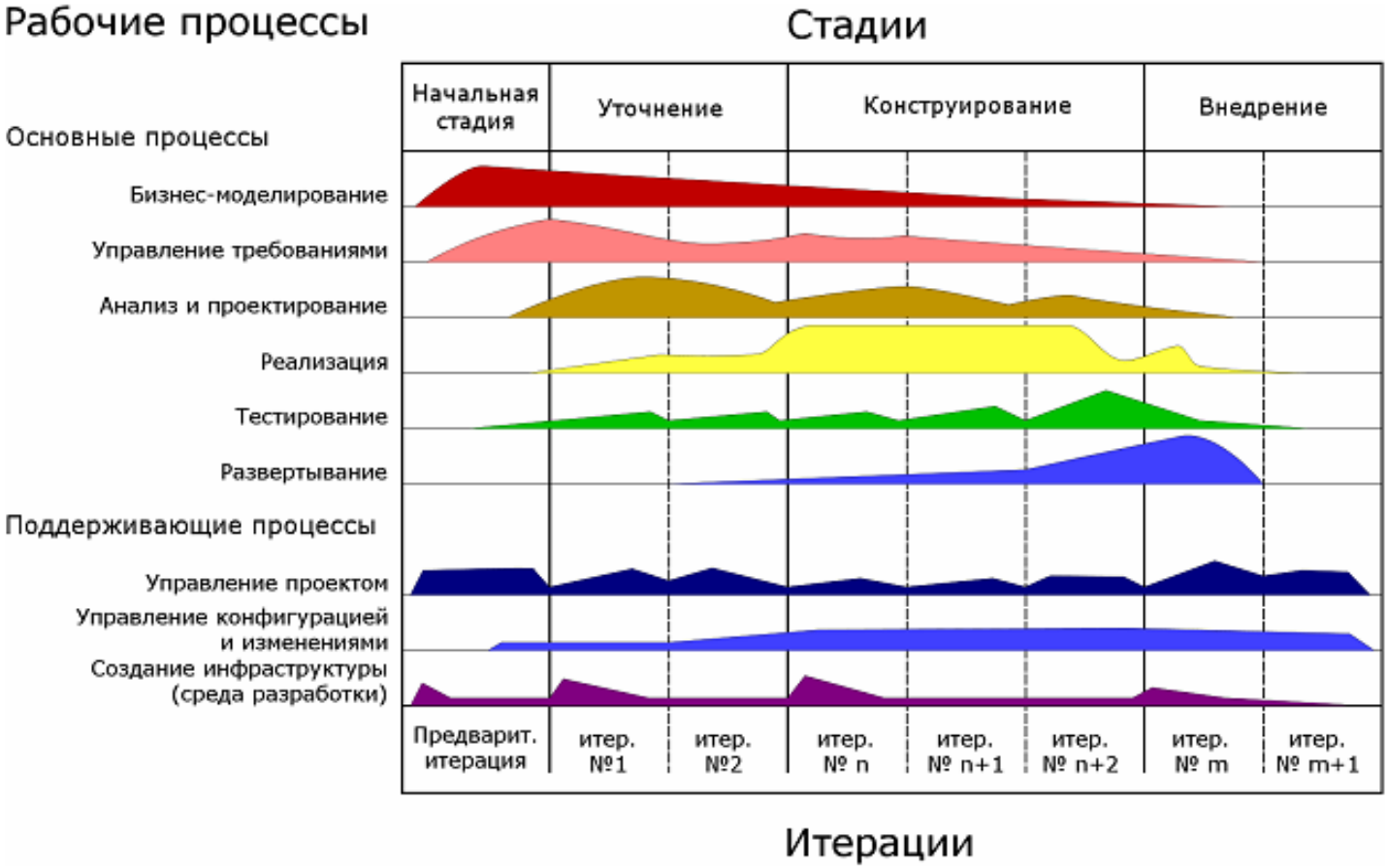
Rational Unified Process

- Авторы:
  - А. Якобсон
  - Г. Буч
  - Дж. Рембо
- Продвигается IBM Rational
- Начало разработки - 1995 г.
- Первая версия RUP - 1998 г.
- Наиболее глубоко проработанная методология

Rational Unified Process

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

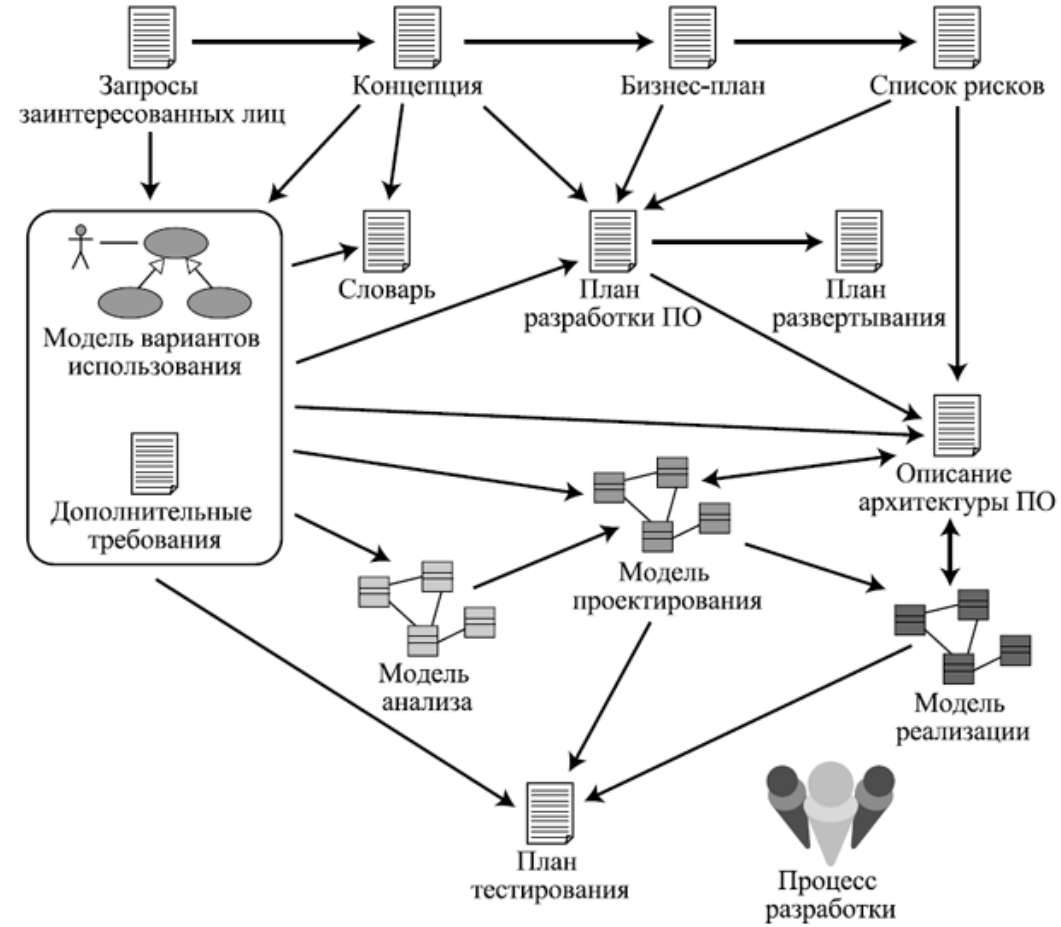
Rational Unified Process



RUP. Рабочие потоки процесса

- Бизнес-моделирование
- Управление требованиями
- Анализ и проектирование
  - Создание статического и динамического представления системы
- Реализация
  - Создание программного кода
- Тестирование
  - Проверка системы в целом

RUP. Артефакты



RUP. Начальная стадия. (Inception)

- Назначение
  - Запуск проекта
- Цели
  - Определение области применения
  - Определение элементов Use Case, критических для системы
  - Определение общих черт архитектуры
  - Определение общей стоимости и плана проекта
  - Идентификация основных элементов риска

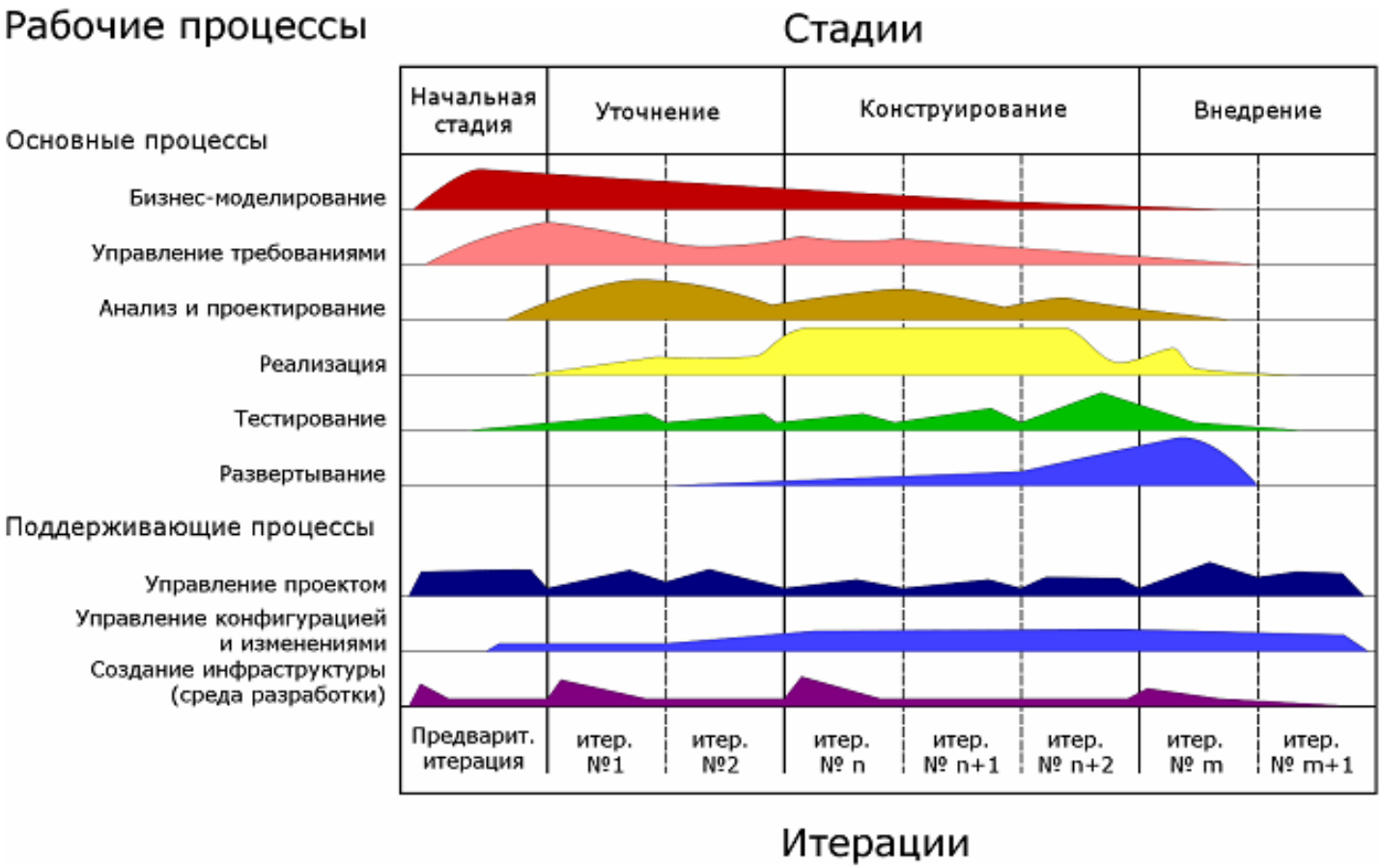
RUP. Начальная стадия. Действия

- Формулировка области применения проекта
  - Выявление требований и ограничений
- Планирование
  - Подготовка основного плана развития и альтернатив развития для управления риском
  - Определение персонала
  - Определение проектного плана
  - Определение зависимостей между стоимостью, планированием и полезностью
- Синтез предварительной архитектуры
  - Развитие решений проектирования
  - Определения используемых компонентов (разработка, покупка, повторное использование)

RUP. Начальная стадия. Артефакты

- Спецификация основных проектных требований
- Начальная модель Use Case (20%)
- Начальный словарь проекта
- Начальный план развития
- Начальная оценка риска
- Проектный план с этапами и итерациями

Rational Unified Process



RUP. Уточнение (Elaboration)

- Назначение
  - Создать архитектурный базис
- Цели
  - Определение оставшихся требований
    - Функциональные требования выражаются с помощью Use Case
  - Определение архитектурной платформы системы
  - Отслеживание рисков, устранение наибольших рисков
  - Разработка плана итераций этапа «Конструирование»

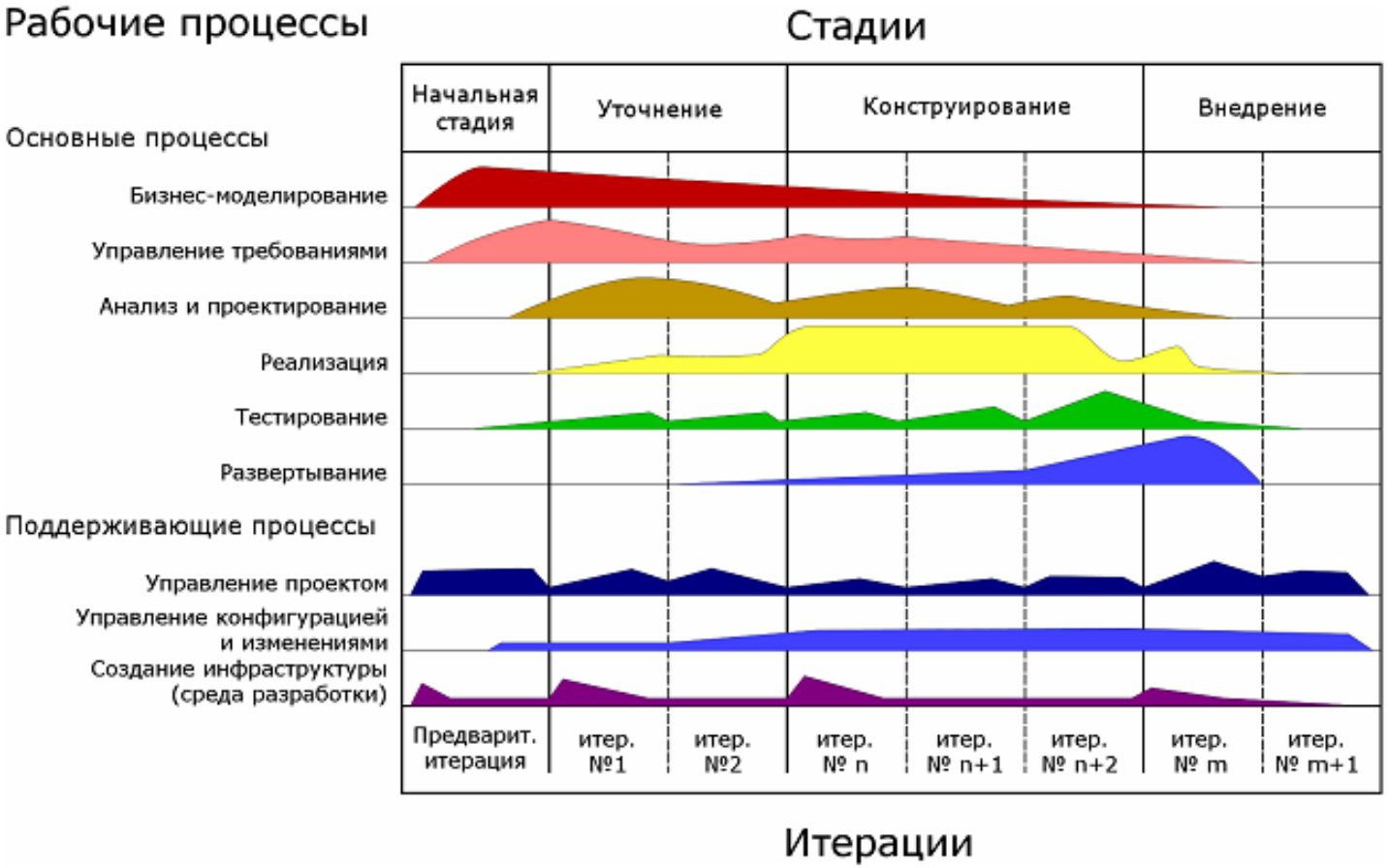
RUP. Уточнение. Действия

- Развитие спецификации
- Формирование критических элементов Use Case, задающих дальнейшие решения
- Развитие архитектуры, выделение ее компонентов

RUP. Уточнение. Артефакты

- Модель Use Case (80%)
- Дополнительные (том числе нефункциональные) требования
- Описание программной архитектуры
- Действующий архитектурный макет
- Переработанный список элементов рисков и основной план развития
- План разработки всего проекта, включающий все итерации и критерий развития для каждой итерации

Rational Unified Process



RUP. Конструирование. (Construction)

- Назначение
  - Создание программного продукта с начальной функциональностью
- Цели
  - Минимизация стоимости разработки
  - Быстрое получение требуемого качества
  - Быстрое получение версий

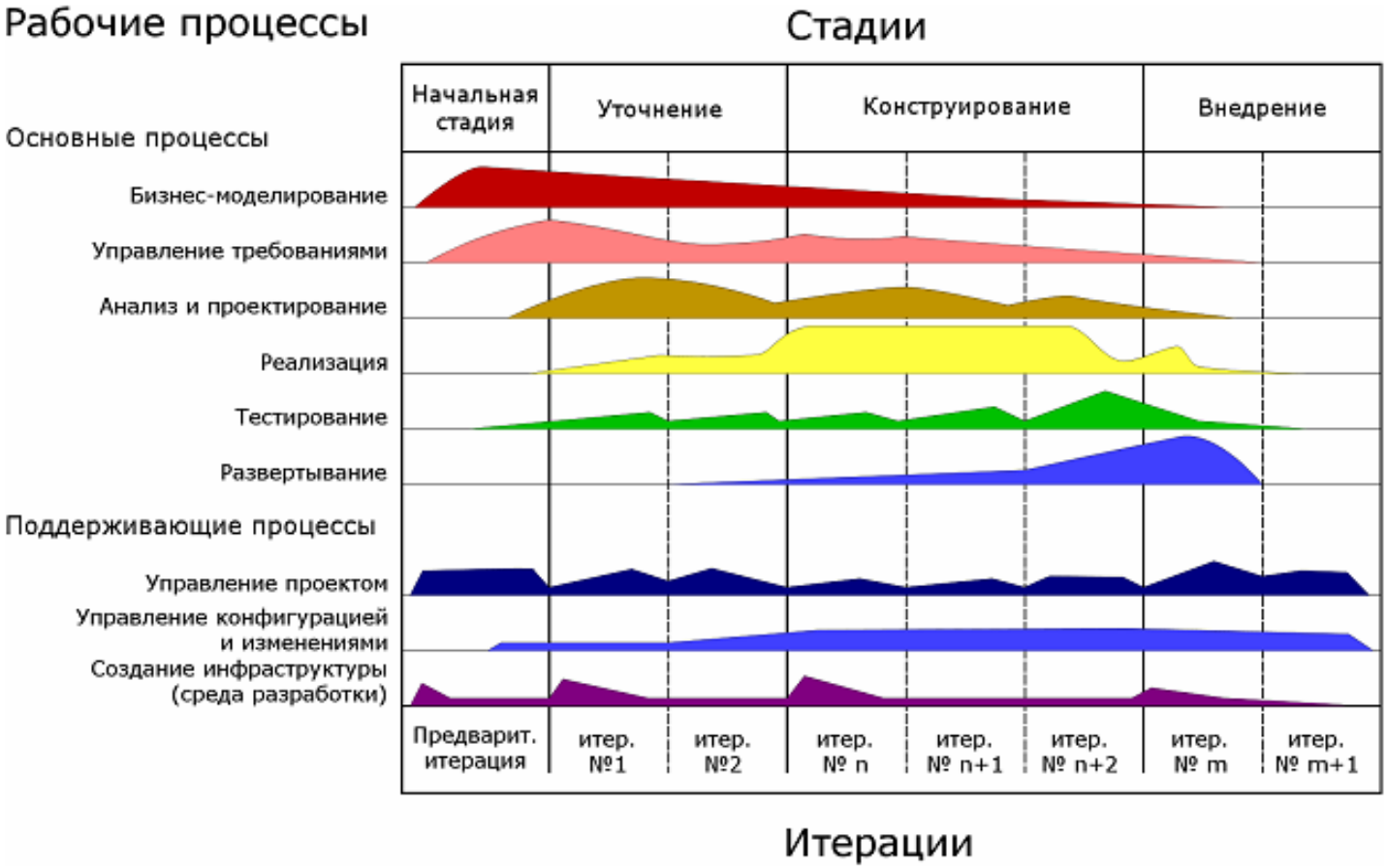
RUP. Конструирование. Действия

- Управление ресурсами, контроль ресурсов
- Оптимизация процессов
- Полная разработка компонентов и их тестирование
- Оценивание реализаций продукта

RUP. Конструирование. Артефакты

- Программный продукт, пригодный для отчуждения от разработчиков (альфа-, бета-версия и т.п.)
- Описание текущей реализации
- Руководство пользователя

Rational Unified Process



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

RUP. Выводы

- Наиболее продуманная методология
- Подходит для больших и очень больших проектов (реже средних)
- Требует высокой квалификации участников

Гибкие (agile) методологии

- Основные особенности
  - Отказ от классических «неповоротливых» подходов
  - Направленность на проекты с постоянно меняющимися требованиями
  - Небольшие команды
  - (!)Высокая значимость не только технических составляющих процесса, но и организационных, социальных и т.п.

Манифест гибкой разработки ПО (Agile Manifesto), 2001 год

Люди и взаимодействие		процессов и инструментов
Работающий продукт	**ВАЖНЕЕ**	исчерпывающей документации
Сотрудничество с заказчиком		согласования условий контракта
Готовность к изменениям		следования первоначальному плану

Известные Agile-методологии

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

Экстремальное программирование

- Экстремальное программирование (XP)
- Автор – Кент Бек, 1999 г
- Ориентирован на группы до 10 чел.
- Группа размещается в одном помещении
- Процесс:
  - гибкий и динамичный
  - наиболее пригоден для проектов с изменяющимися требованиями
  - итеративный

Экстремальное программирование

- Основные действия:
  - Кодирование
  - Тестирование
  - Выслушивание заказчика
  - Проектирование
- Динамика определяется:
  - Непрерывностью связи с заказчиком
  - Простотой – выбирается простейшее решение
  - Быстрой обратной связью
  - Профилактика проблем

Методы XP

1. Planning game (Игра в планирование)
2. Small releases (Небольшие версии)
3. Metaphor (Метафора)
4. Simple design (Простой дизайн)
5. Testing (Тестирование)
6. Refactoring (Рефакторинг)
7. Pair programming (Парное программирование)
8. Collective ownership (Коллективное владение)
9. Continuous integration (Непрерывная интеграция)
10. 40-hour week (40-часовая неделя)
11. On-site customer (Локальный заказчик)
12. Coding standards (Стандарты кодирования)

XP. Игра в планирование

- Заказчик:
  - Объем работ
  - Приоритет
  - Композиция версий
  - Сроки выпуска версий
- Разработчик:
  - Временные оценки
  - Последствия
  - Процесс
  - Подробный график работ

XP. Небольшие версии

- Быстрый запуск простой системы
- Версия маленькая, насколько это возможно
- Версия должна быть завершенной
- Обычно выпуск версии 1 раз в 2 недели

XP. Метафора

- Глобальное «видение» проекта, понятное всем
- «Замена» большой архитектуры
- В данном контексте – основная идея проекта, сведения об архитектуре

XP. Простой дизайн

Правильный дизайн:

- Выполняются все тесты
- Нет дублирующей логики
- Выражаются все важные идеи
- Минимальное число классов и методов

Добавляется в дизайн то, что нужно, только тогда, когда нужно

XP. Тестирование

- Для любой возможности существуют автоматические тесты
- Модульные тесты – разработчики
- Функциональные тесты – заказчики
- Репозиторий тестов постоянно увеличивается
- Код разрабатывается вместе с тестами (или после тестов)

XP. Рефакторинг

- Изменение программы для упрощения добавления новой возможности
- Изменение программы после добавления новой функциональности
- Программы до и после рефакторинга функционально эквивалентны

XP. Парное программирование

- Разработчики работают парами
- Партнер с клавиатурой и мышью думает о реализации
- Партнер без клавиатуры и мыши думает стратегически:
  - Сработает ли данный подход?
  - Какие еще тесты нужно разработать?
- Состав пар меняется динамически
- Эффективность ПП очень высокая

XP. Коллективное владение

- Код – общая собственность
- При необходимости код модифицируется немедленно, независимо от авторства кода

XP. Непрерывная интеграция

- Код интегрируется раз в несколько часов. Не реже 1-го раза в день
- Интеграция нового кода заканчивается после прохождения системой всех тестов
- Ответственны за интеграцию пара, которая внесла изменения

XP. 40-часовая неделя

- Сверхурочные – крайне нежелательны
- Если постоянно требуется переработка – неправильно организован проект
- Отпуск – обязателен

XP. Локальный заказчик

- В составе команды – представитель заказчика
- Представитель – пользователь системы
- Представитель отвечает на вопросы разработчиков и расставляет мелкие приоритеты

XP. Стандарты кодирования

- Единый стандарт кодирования
- Стандарт должен способствовать коммуникациям
- Стандарт должен быть добровольно воспринят командой

Методология SCRUM

- Предложена в 1995, Оксфорд
- Scrum – схватка
- Управление хаосом
- Итерационный процесс
- Применима к любым этапам и особенностям разработки (в основном – разработка и сопровождение)
- Хорошо стыкуется с использованием объектно-ориентированного подхода

SCRUM. Артефакты

- Backlog
  - Список работ, которые необходимы выполнить
- Backlog sprint
  - набор требований, которые могут быть реализованы за один этап (спринт)

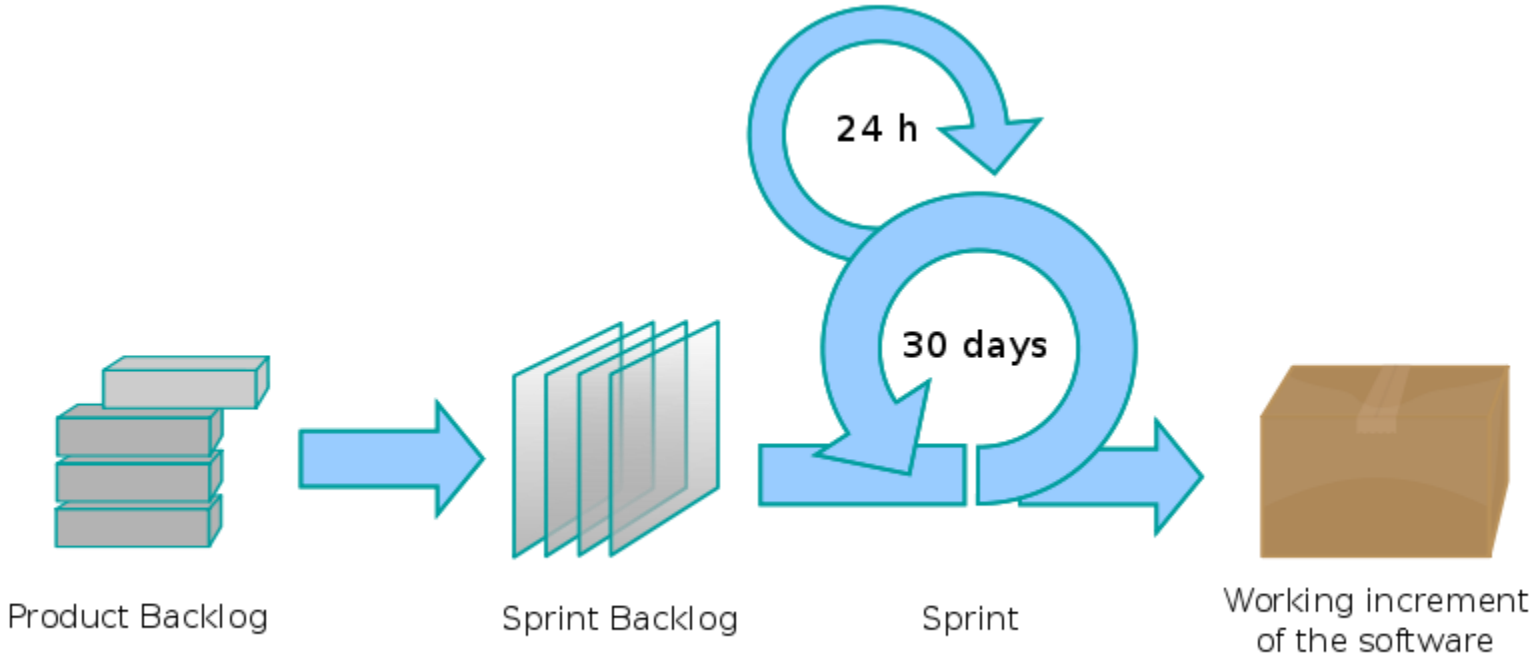
SCRUM. Планирование

- 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

SCRUM. Роли

- Основные
  - Владелец продукта
  - Руководитель (*ScrumMaster*)
  - Команда (!)
- Остальные
  - Пользователи
  - Клиенты
  - Эксперты-консультанты

Методология SCRUM



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

Методология SCRUM

- Каждая группа ежедневно выполняет схватки (scrum) (10-30 мин):
  - Что сделано каждым в предыдущий день?
  - Что будет сделано каждым 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

Технологические подходы к проектированию ПО. Итоги

	Классическая	Прототирование	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(9)
Продолжительность	Выс	Низк	Выс	Низк	Низк	Сред, Выс	Низк	Низк
Промеж. версии	-	-	+/-	+	+/-	+	+	+
ИС	-	-	-	-	+	+	+	+