Технология и методы программирования

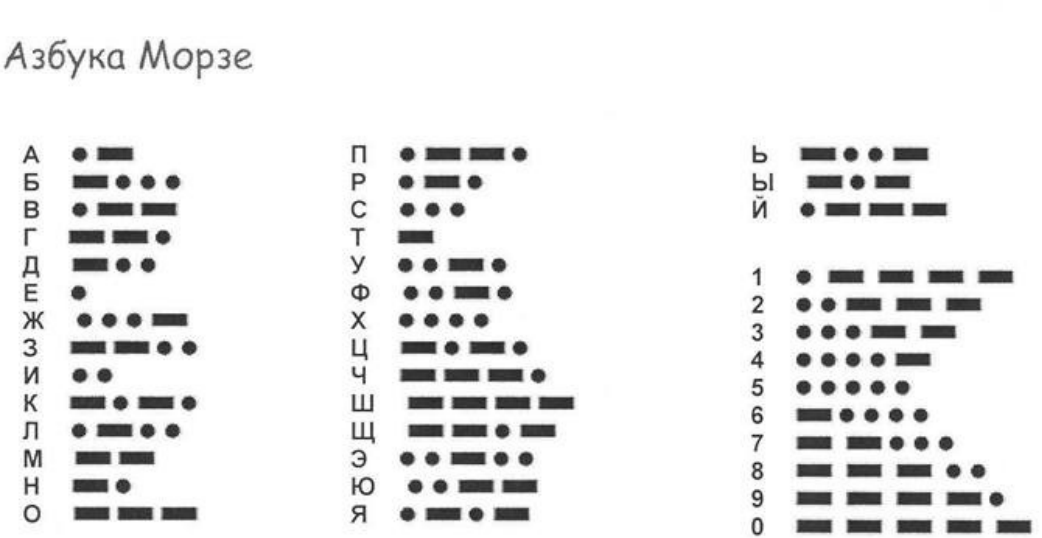
Лекция 1. Процесс разработки ПО, методологии, модели.

whoami

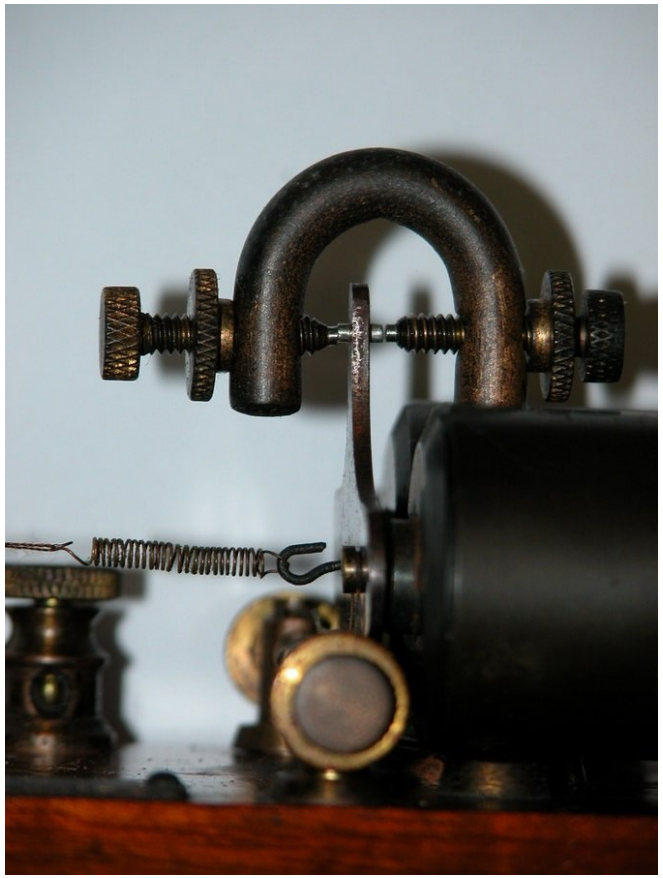
Tg: @dzruyk
Vk: lonely.ruyk

История возникновения компуктеров. Телеграф

Сэмюэл Морзе в 1840 году запатентовал электромагнитный телеграф. Большой заслугой изобретателя стало создание телеграфного кода



История возникновения компуктеров
Реле



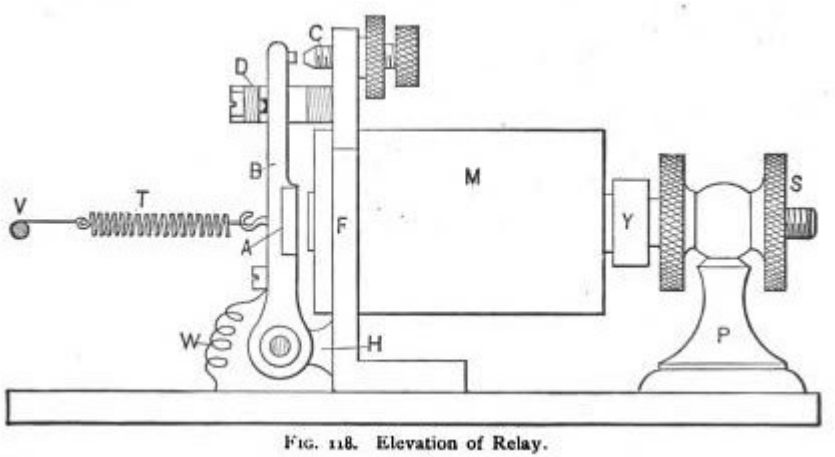
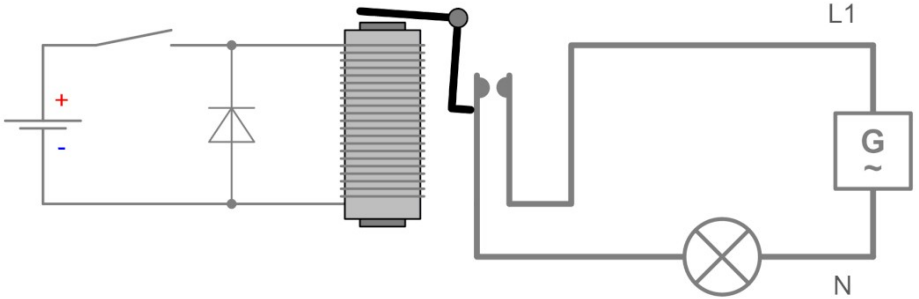
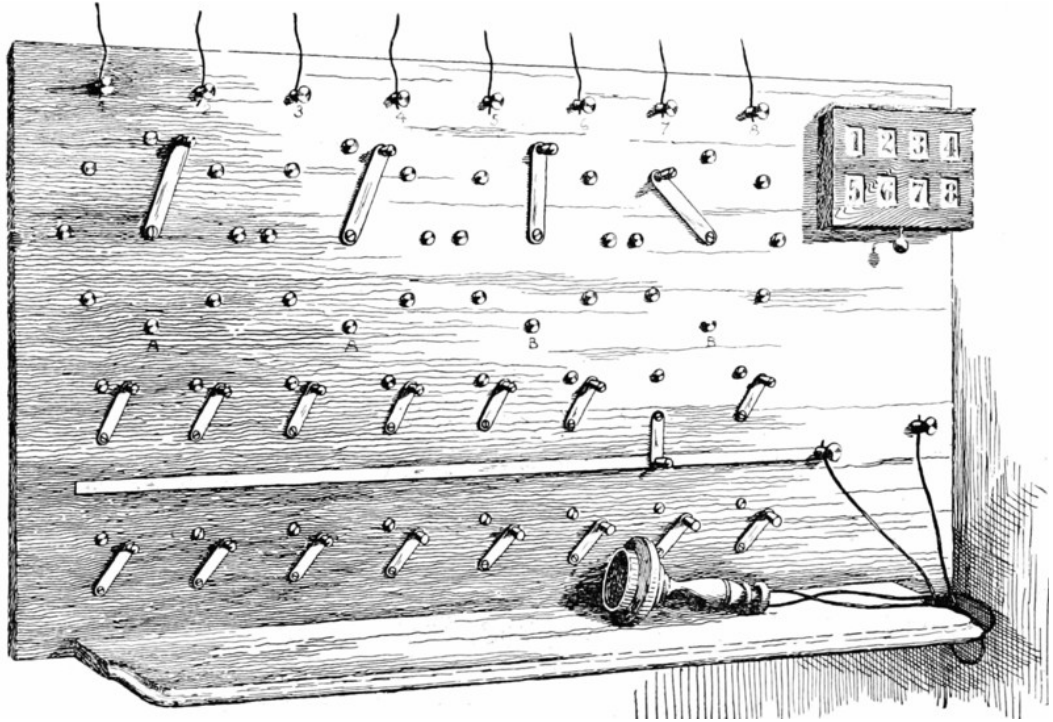


Fig. 118. Elevation of Relay.

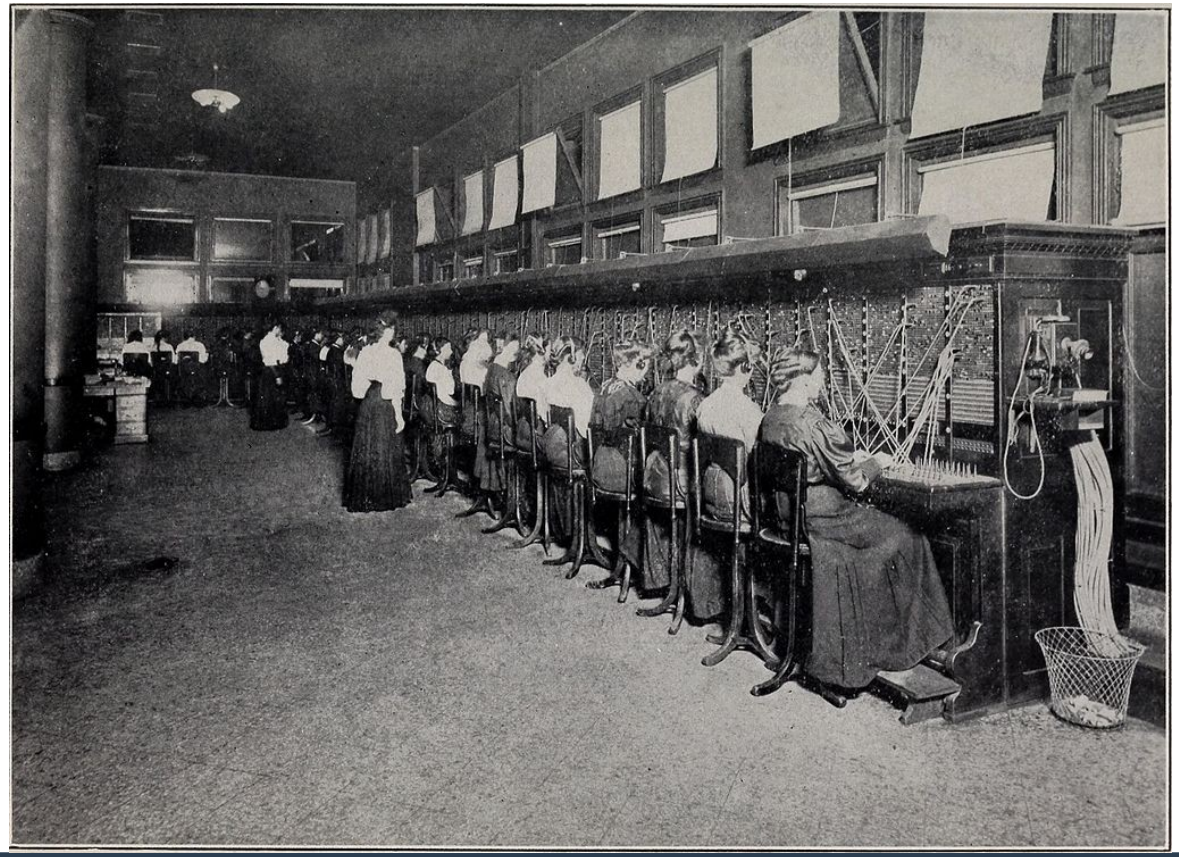


История возникновения компуктеров. Телефон

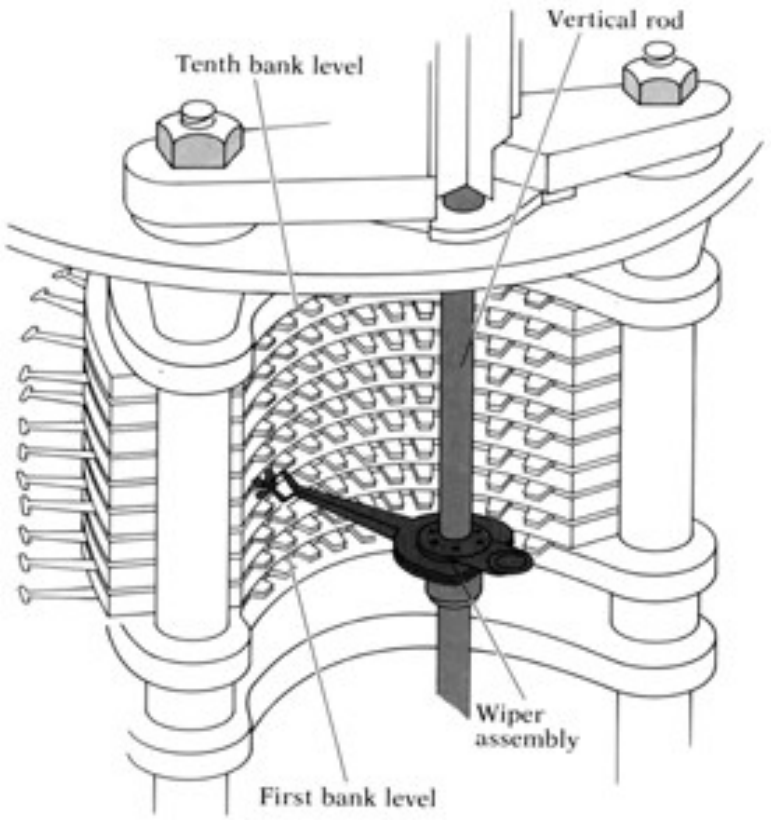
История возникновения компуктеров. Телефонные коммутаторы

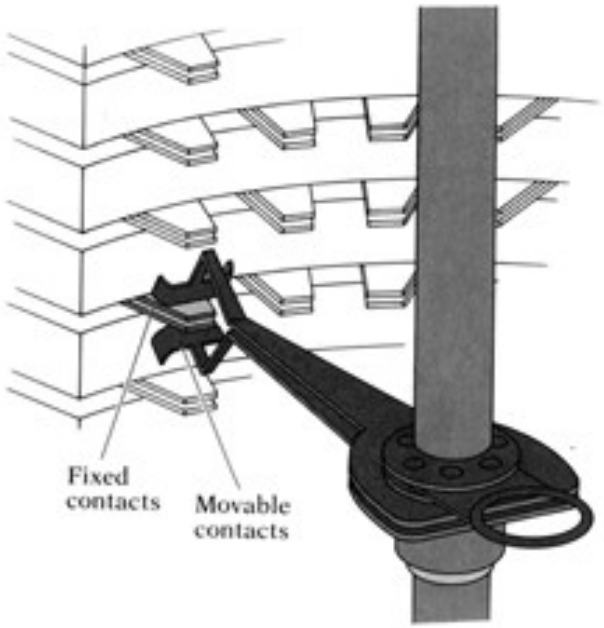


История возникновения компуктеров. Телефонные коммутаторы



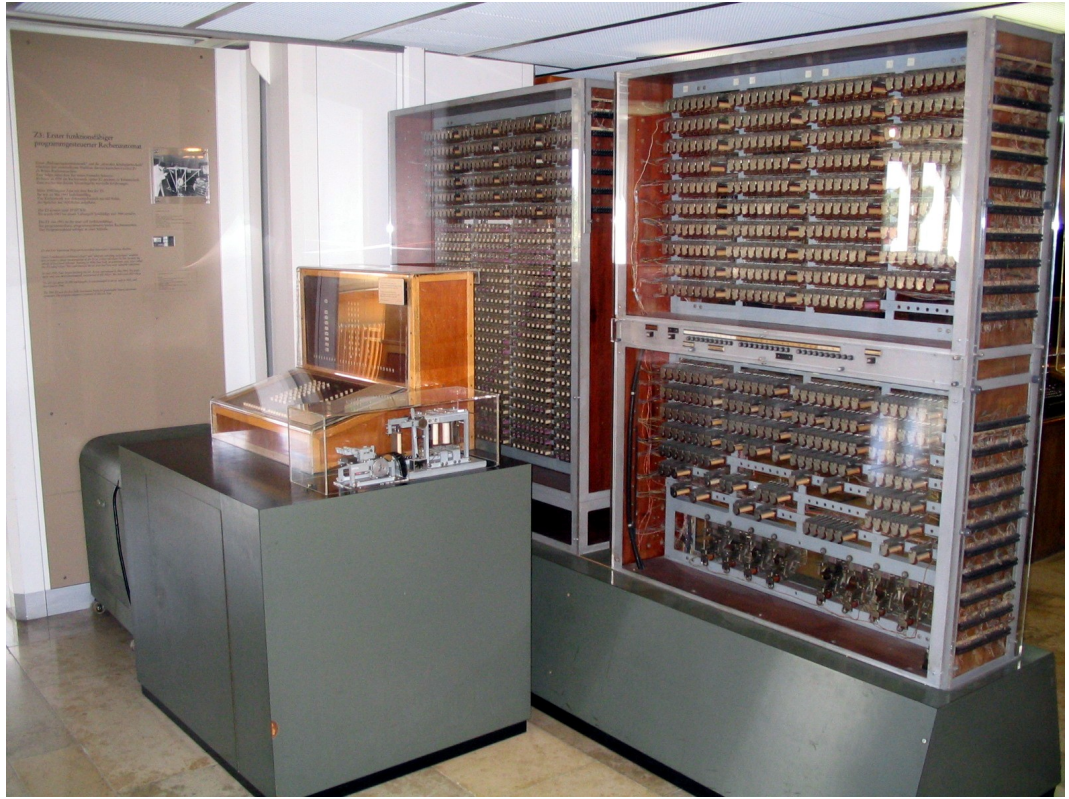
История возникновения компуктеров. Автоматические телефонные коммутаторы



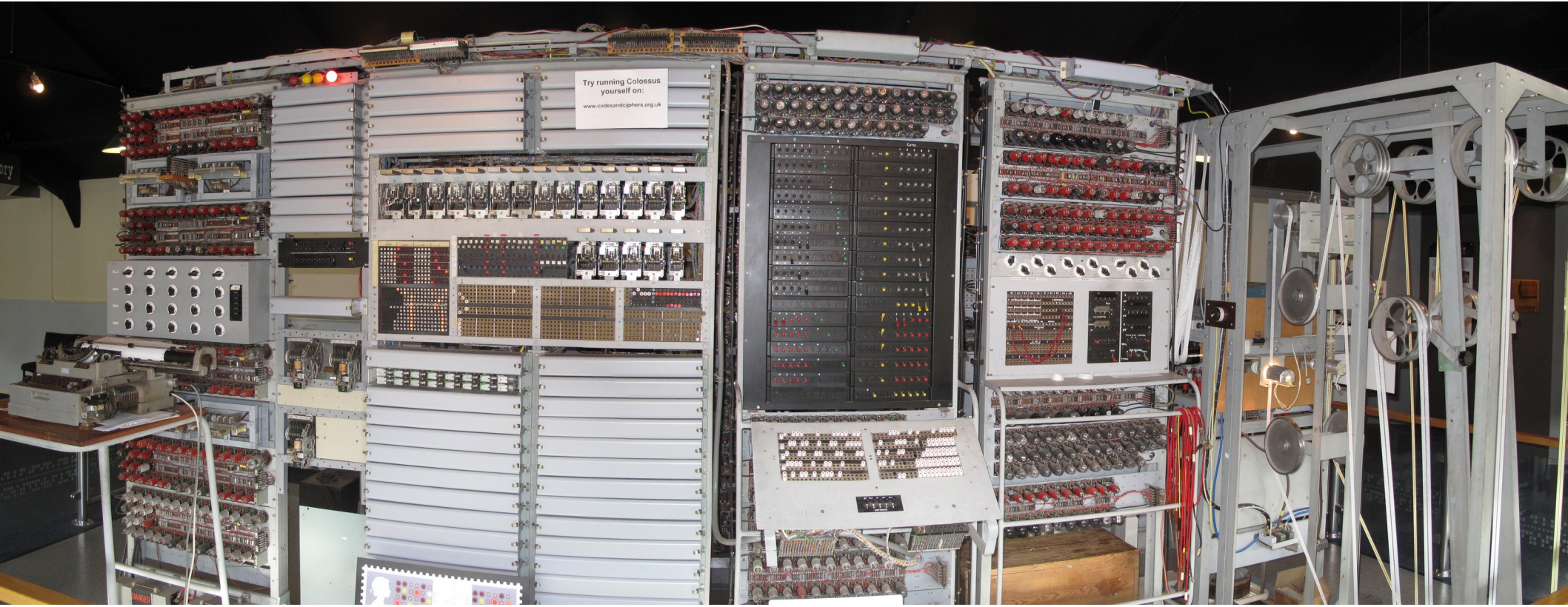


The movable contacts in a step-by-step switch can connect to any of a 100 different pairs of fixed contacts, each leading to a different line.

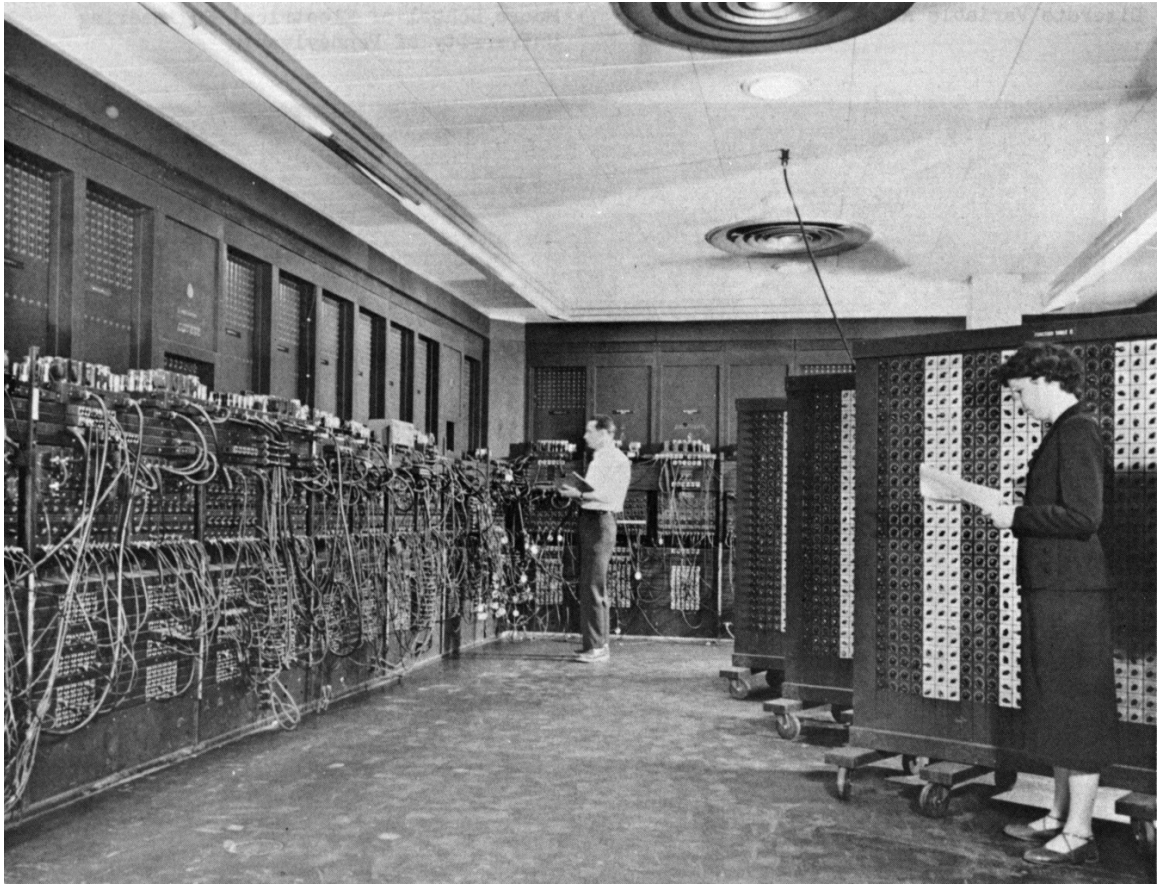
Первые компьютеры. Z3



Первые компьютеры. Colossus



Первые компьютеры. ENIAC



Компьютеры. 1950-е, где-то в Лос-Анджелесе

(с)тырено с Paul E. Ceruzzi: A history of modern computing 2 ed

Компьютеры. 10 лет спустя

Итоги нашего (короткого?) исторического экскурса

Технологии, о которых говорилось выше, проходили через похожие стадии развития.

Общие паттерны:

- Упрощение «опыта пользователя» за счёт усложнения внутреннего устройства.
- Период бурного роста связан со стандартизацией и унификацией.
- Злые машины отбирают работу у специально обученных тётенек и дяденек :-(

Жизненный цикл технологий

- НИОКР (научно исследовательские и опытно-конструкторские работы a.k.a. R&D)
- Период интенсивного развития
- Период зрелости
- Спад

Задача по управлению сложностью

Одна из самых важных задач при разработке ПО (и любой другой инженерной деятельности) — задача по управлению сложностью.

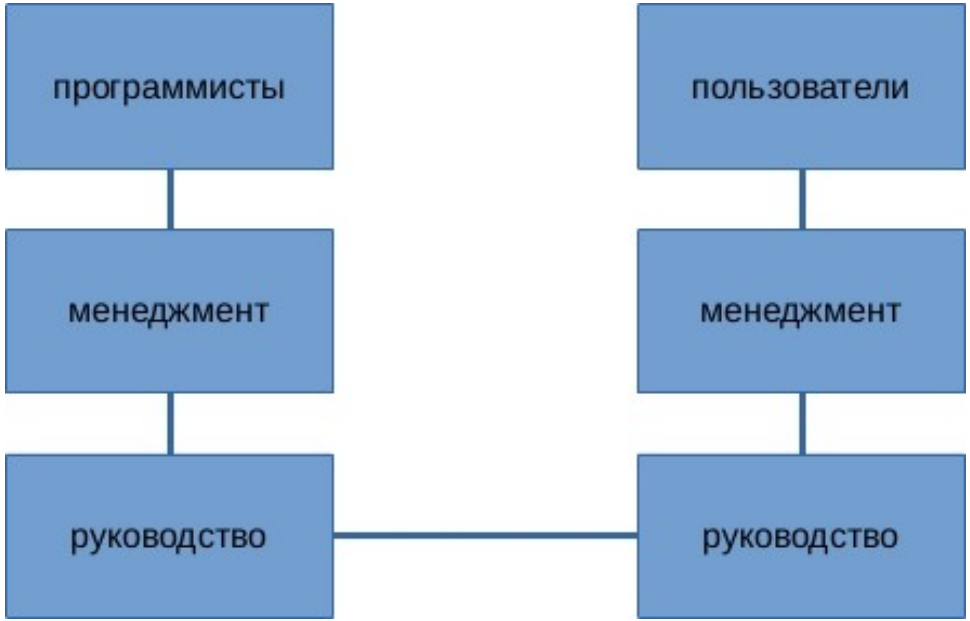
Излишняя сложность приводит к появлению систем, которые дороже, менее надёжны и качественны. И на разработку таких систем тратится (как правило) больше времени

Как можно управлять сложностью?

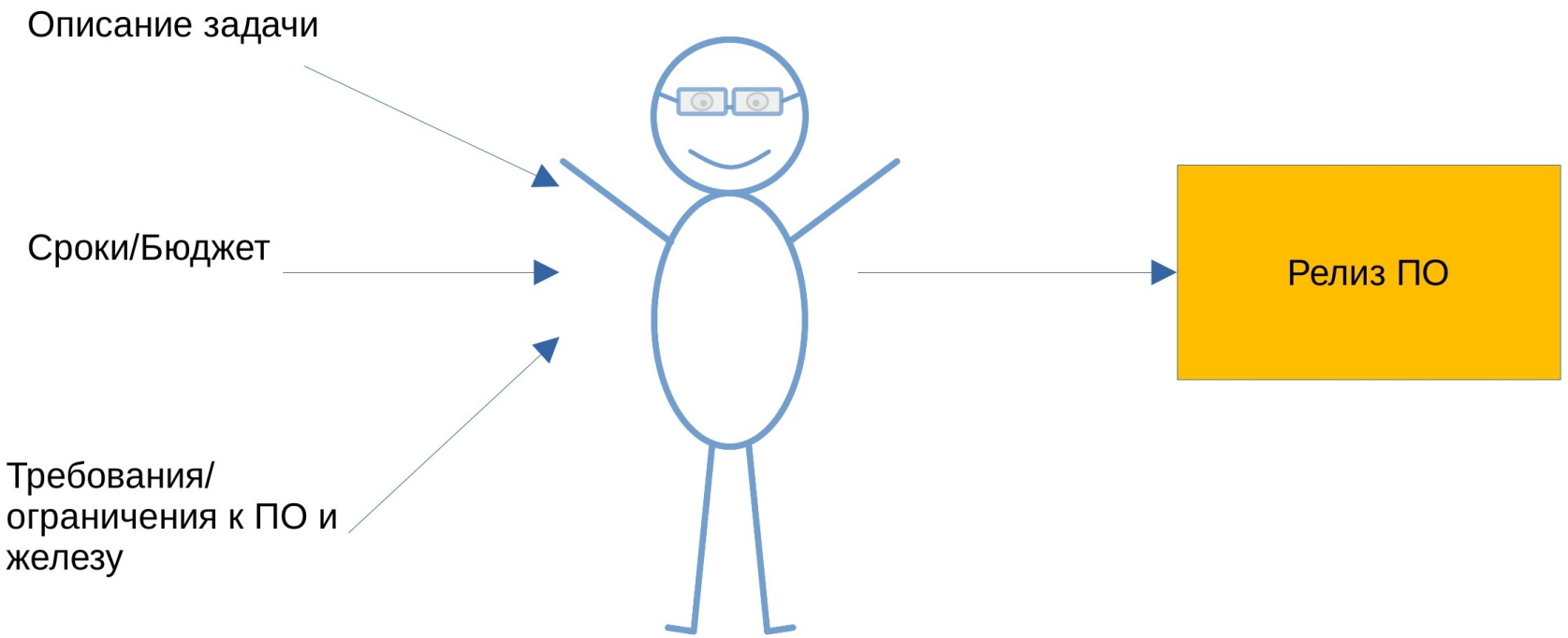
- Абстракция — упрощаем реальный объект некой моделью.
- Иерархичность — разбиваем систему на отдельные модули (рекурсивно разбиваем на подмодули до того уровня, на котором его легко будет понять).
- Модульность — отдельные модули должны иметь фиксированную функциональность и чёткий набор интерфейсов для работы с ними.
- Регулярность — Принцип требует соблюдения единообразия при проектировании отдельных модулей в системе.

Процесс разработки ПО

Общая схема. Субъекты

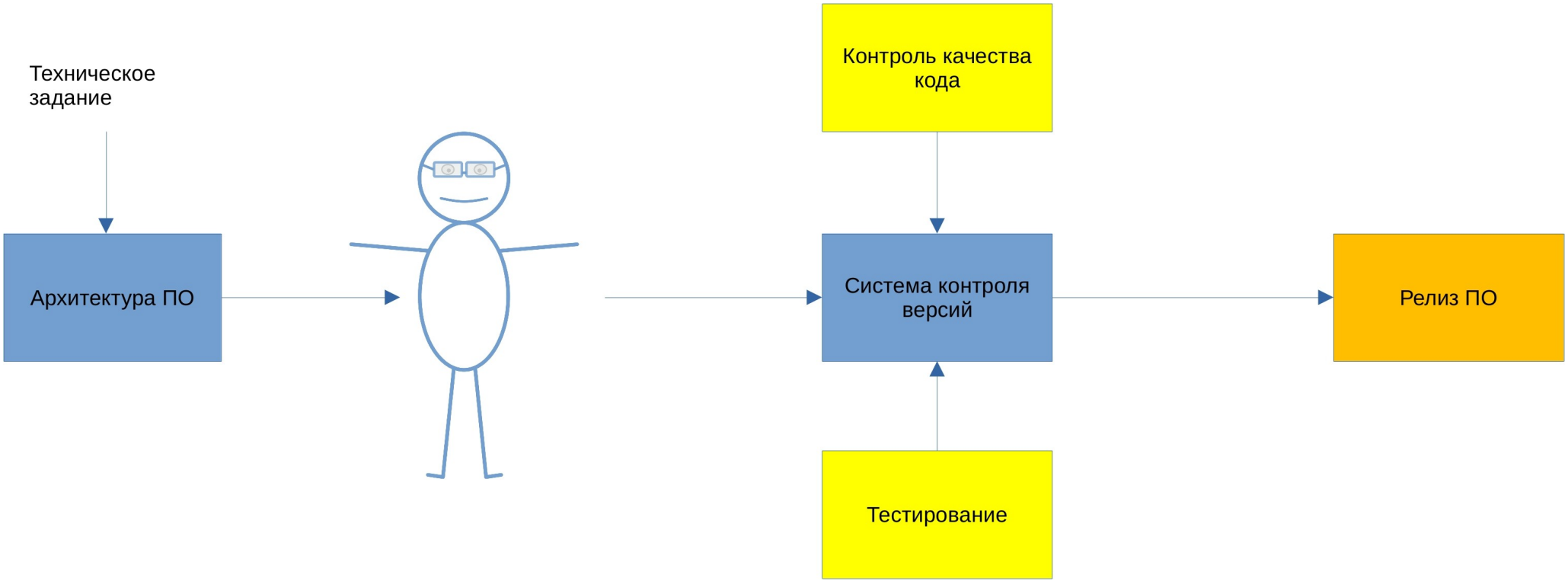


Общая схема. Процесс разработки. v1

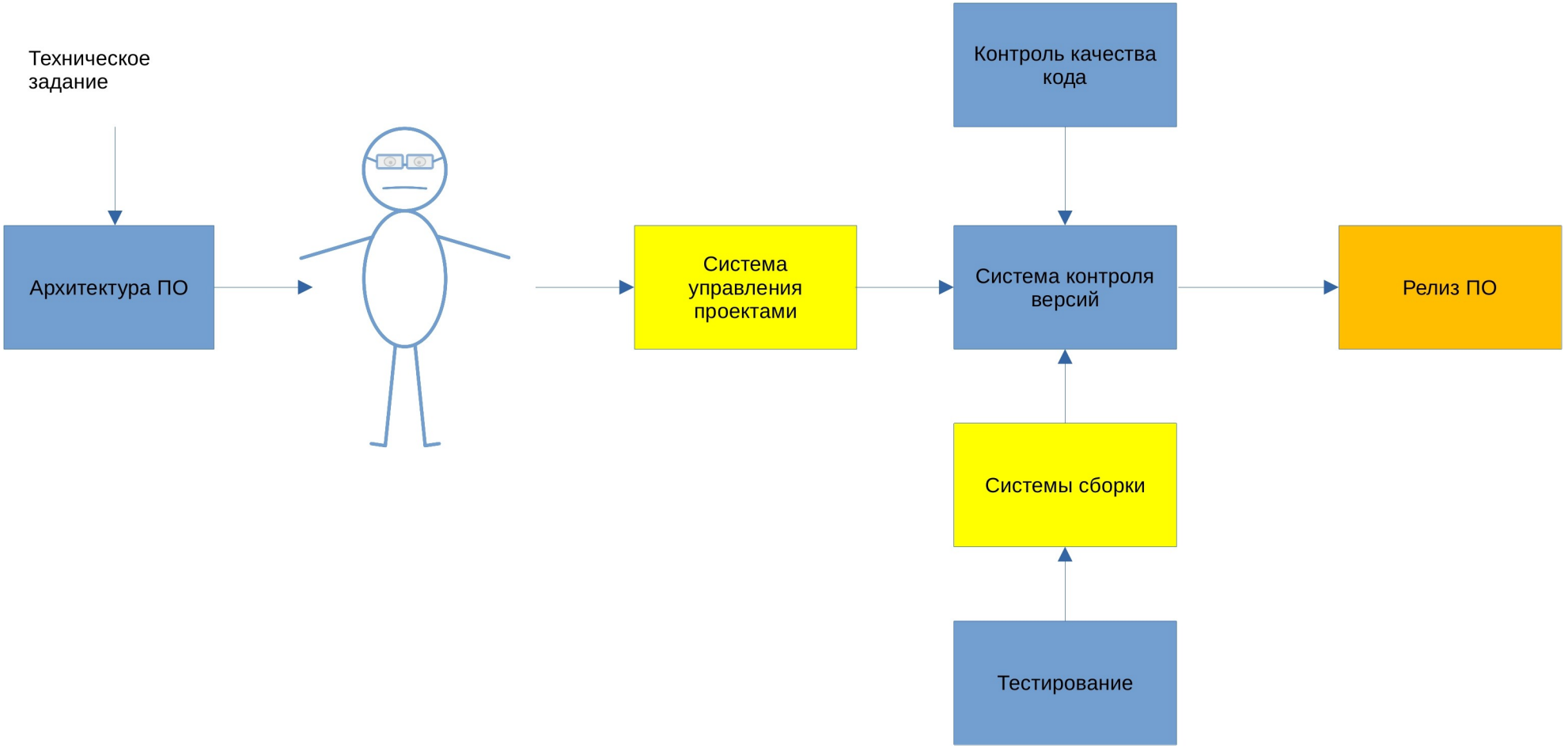


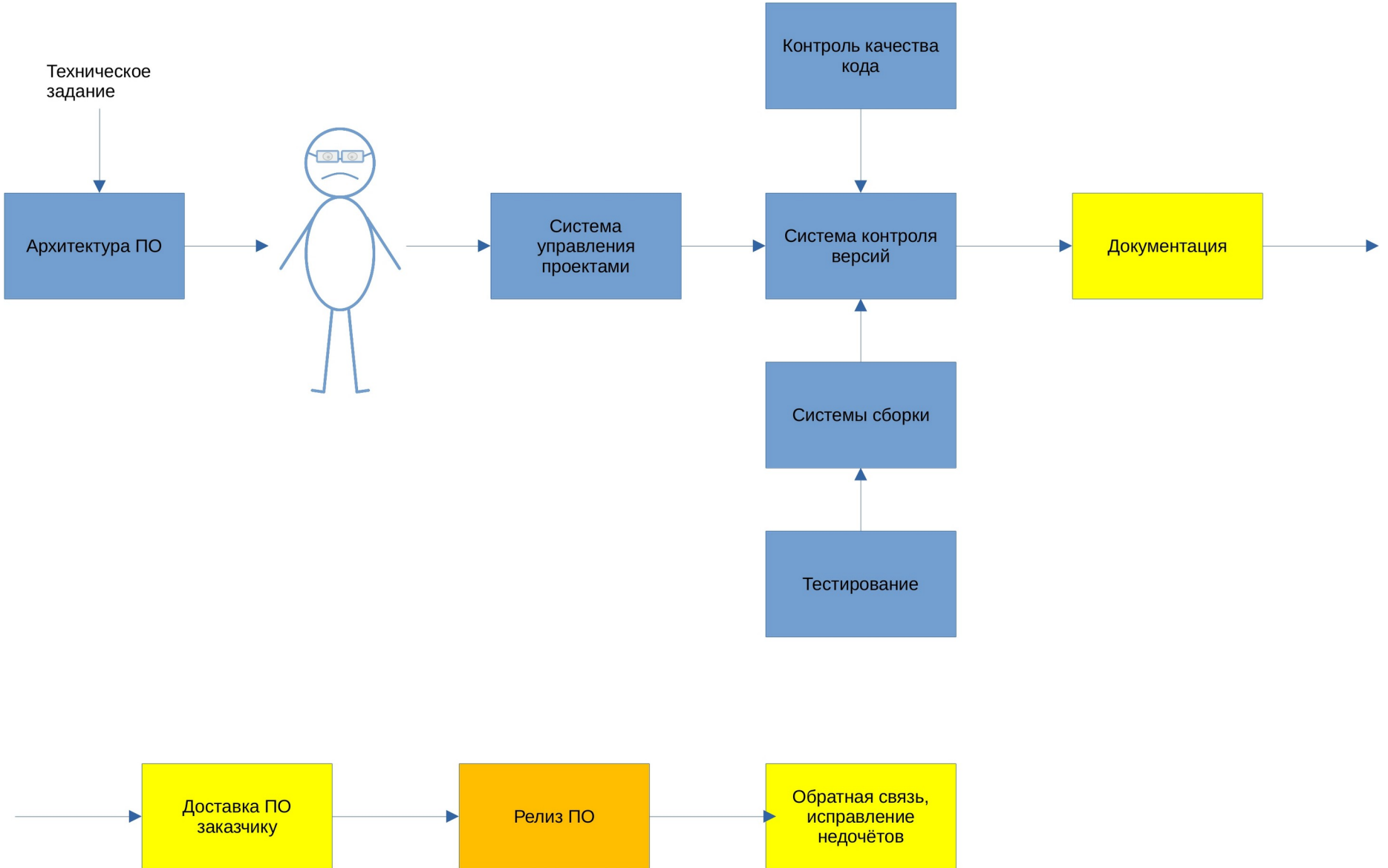
Общая схема. Процесс разработки. v2

Общая схема. Процесс разработки. v3



Общая схема. Процесс разработки. v4




Техническое задание
Архитектура ПО
Система управления проектами
Контроль качества кода
Система контроля версий
Документация
Системы сборки
Тестирование
Доставка ПО заказчику
Релиз ПО
Обратная связь, исправление недочётов

Риски при разработке ПО

1)Неверное/Неточное ТЗ
2)Бюджет
3)Время
4)Кадровый риск
5)Отсутствие системы контроля
6)...

Bus factor (автобусное число)

Итоги

Разработка ПО – непростой процесс, который приходится неким образом формализовать.

Формализация нужна для снижения рисков.

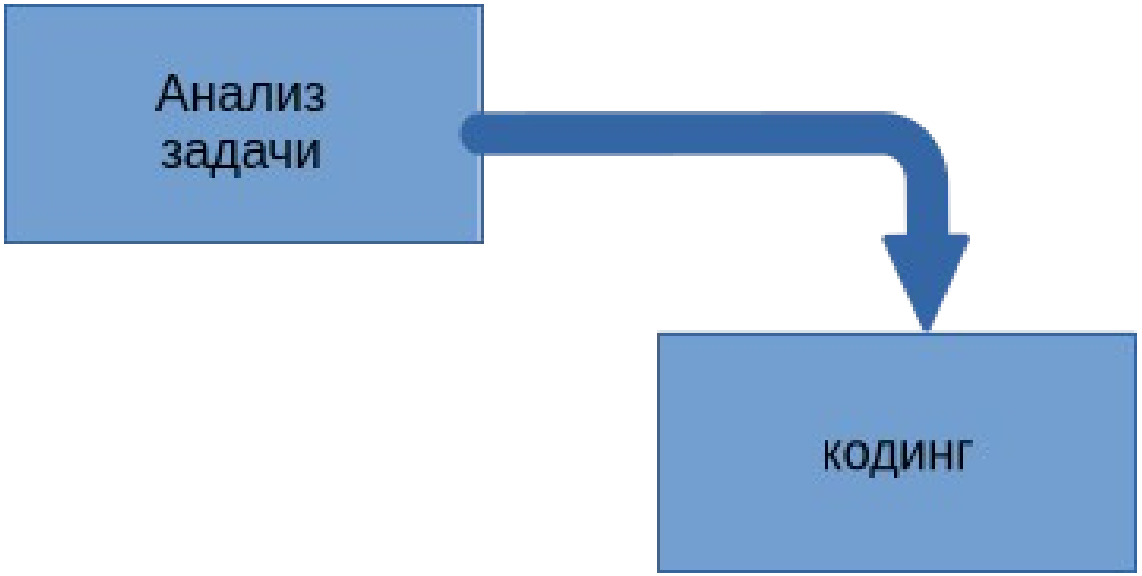
Как формализовать?

- Использовать стандарты разработки ПО
- Использовать некие стандартные модели и методологии* разработки

*Методология — система подходов, методов к решению некой проблемы

Модели и методологии разработки ПО

Тривиальная схема разработки ПО



Роли в разработке

- Менеджер проекта
- Заказчик
- Системный архитектор
- UX-дизайнер
- Разработчик
- Тестировщик
- DevOps инженер

Жизненный цикл ПО

- Планирование и анализ требований от заказчика
- Определение требований
- Проектирование
- Прототипирование(?)
- Кодирование
- Тестирование/отладка
- установка/развёртывание
- Поддержка
- «Смерть» ПО

Кризис разработки ПО

Ближе к концу 1960-х обострился ряд проблем с разработкой ПО:

- Очень часто разработка проектов значительно превышала первоначально заложенные сроки и бюджеты.
- ПО низкого качества, которое не удовлетворяет требованиям
- Код сложен для сопровождения
- Программы не доходят до выпуска

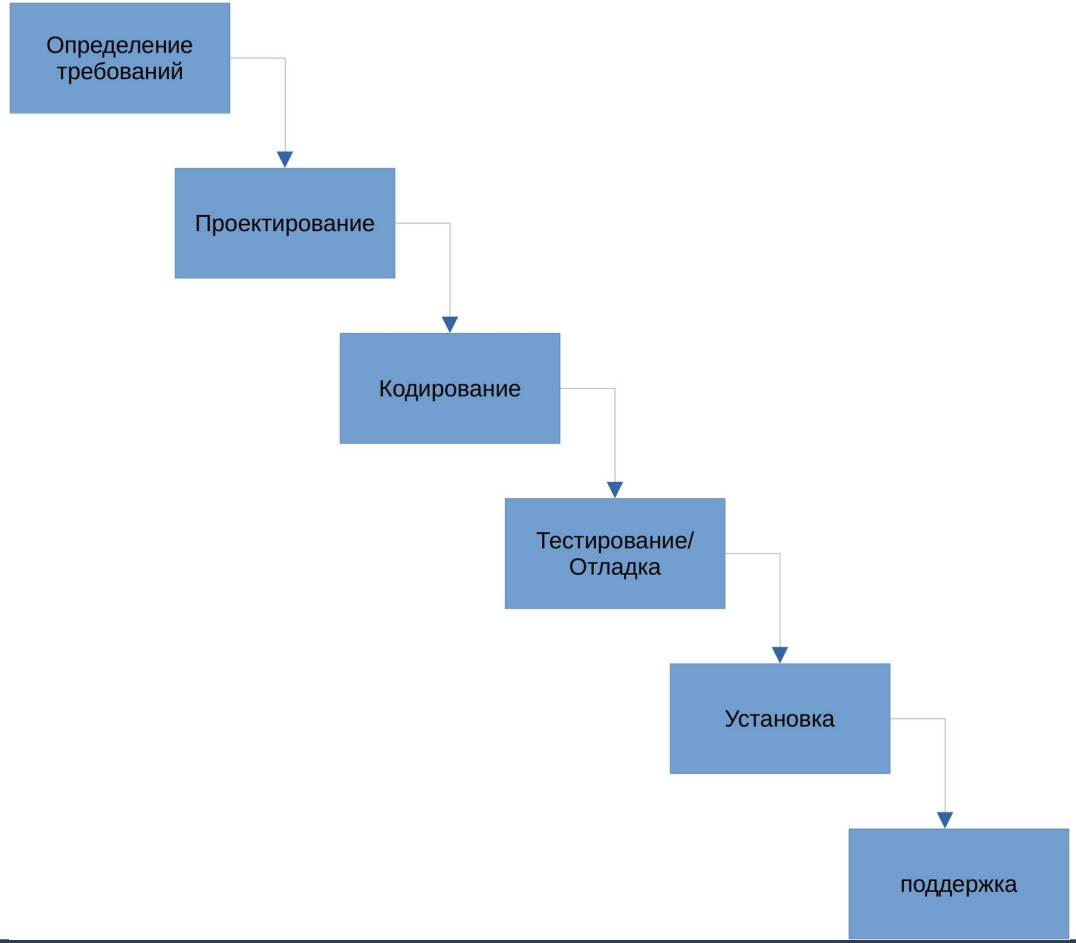
Причина: рост сложности проектов

Определения

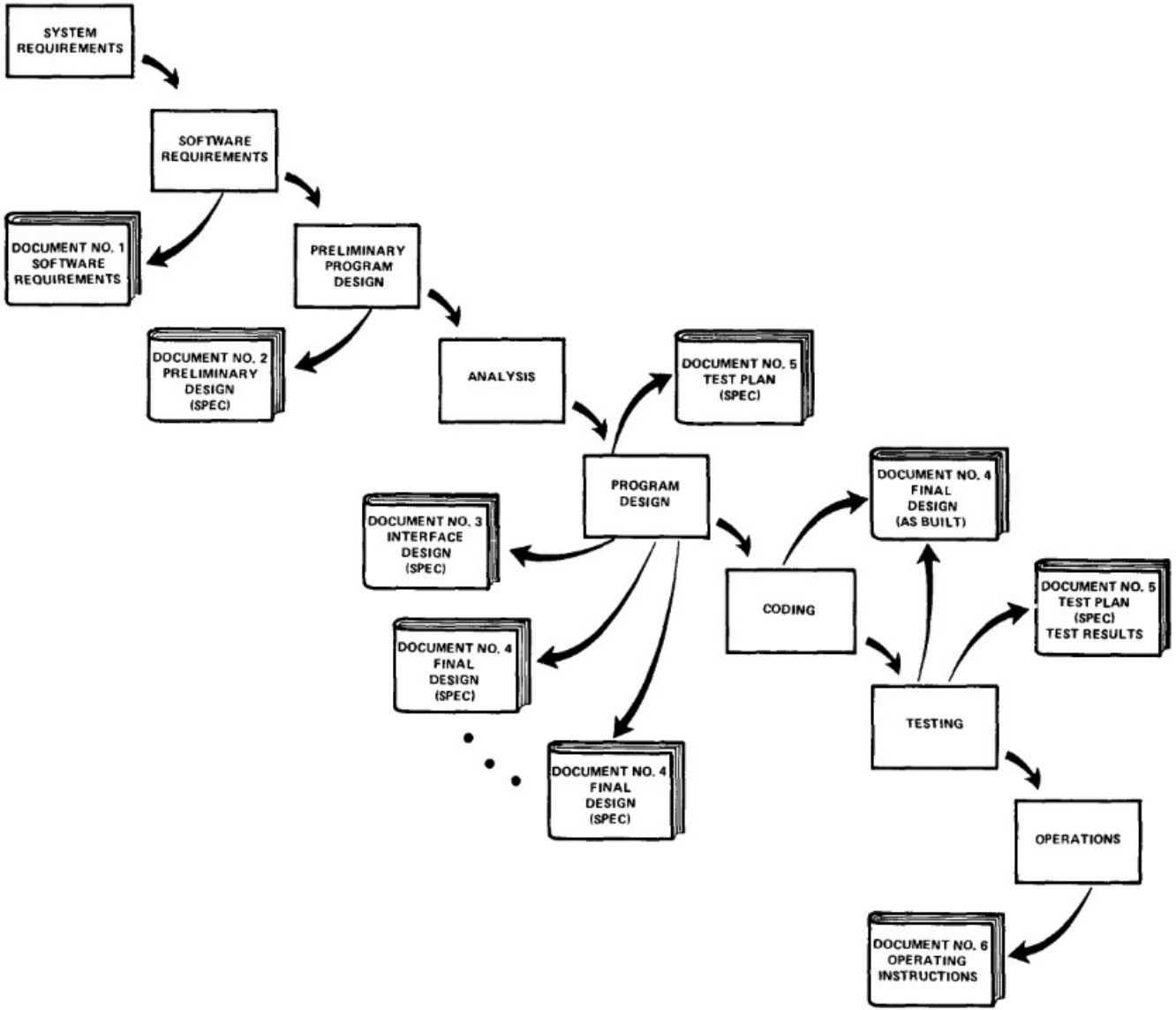
Модель разработки ПО — это общее описание того, через какие стадии ПО проходит во время создания и что происходит на каждой из них.

Методология разработки ПО— определённый набор методов и практик, повышающих эффективность разработки ПО.

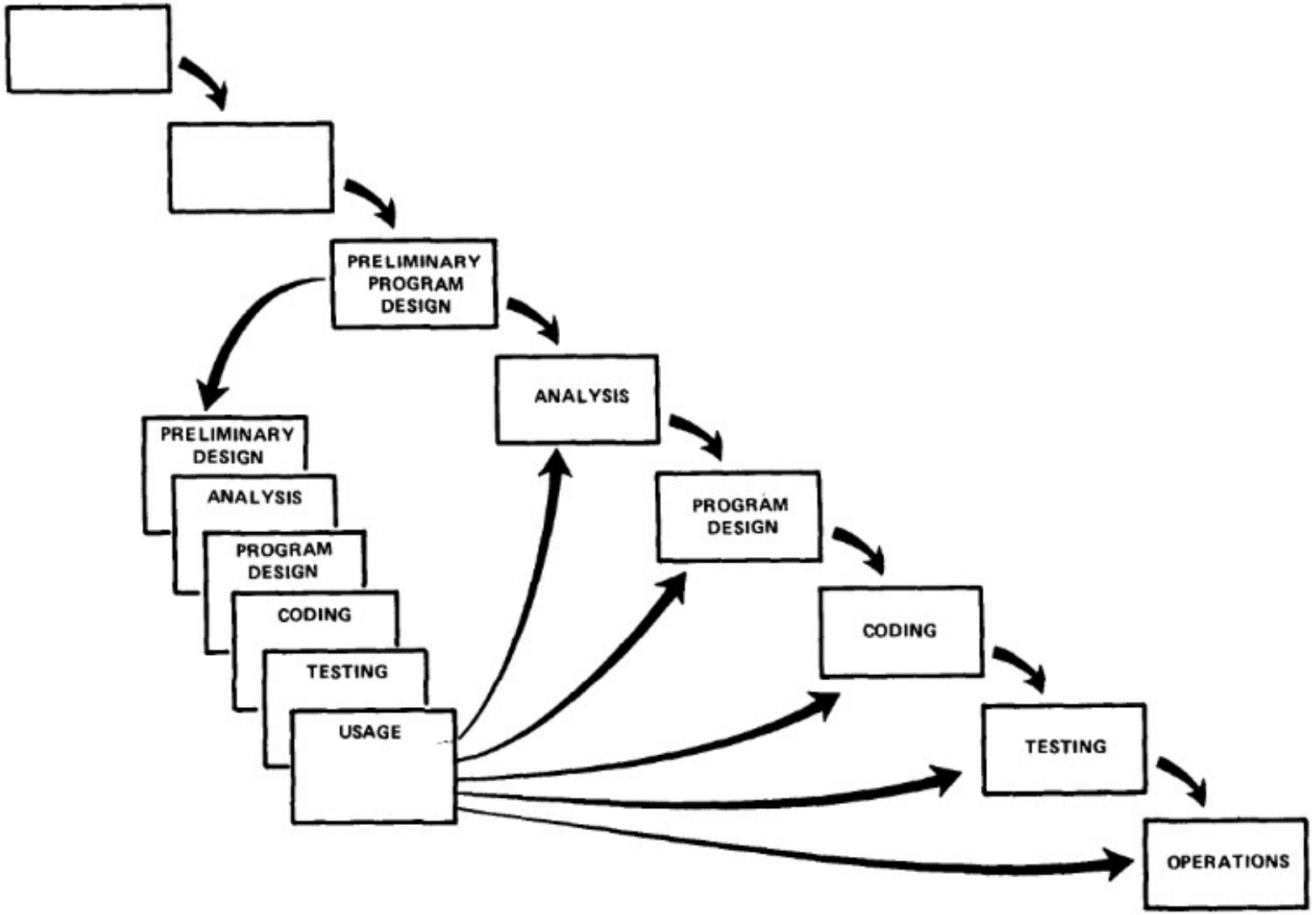
Модели разработки. Каскадная модель



оригинал доки про waterfall


SYSTEM REQUIREMENTS
SOFTWARE REQUIREMENTS
DOCUMENT NO. 1 SOFTWARE REQUIREMENTS
PRELIMINARY PROGRAM DESIGN
DOCUMENT NO. 2 PRELIMINARY DESIGN (SPEC)
ANALYSIS
DOCUMENT NO. 5 TEST PLAN (SPEC)
PROGRAM DESIGN
DOCUMENT NO. 3 INTERFACE DESIGN (SPEC)
DOCUMENT NO. 4 FINAL DESIGN (SPEC)
DOCUMENT NO. 4 FINAL DESIGN (SPEC)
CODING
DOCUMENT NO. 4 FINAL DESIGN (AS BUILT)
DOCUMENT NO. 5 TEST PLAN (SPEC) TEST RESULTS
TESTING
OPERATIONS
DOCUMENT NO. 6 OPERATING INSTRUCTIONS

daily double) is invariably and seriously optimistic.



Модели разработки. Каскадная модель

Плюсы:

- Полное документирование каждого этапа до начала разработки;
- Низкий риск ошибок за счет детальной документации;
- Прозрачность процессов для заказчика — он знает, сколько времени уйдет на каждый этап.

Недостатки:

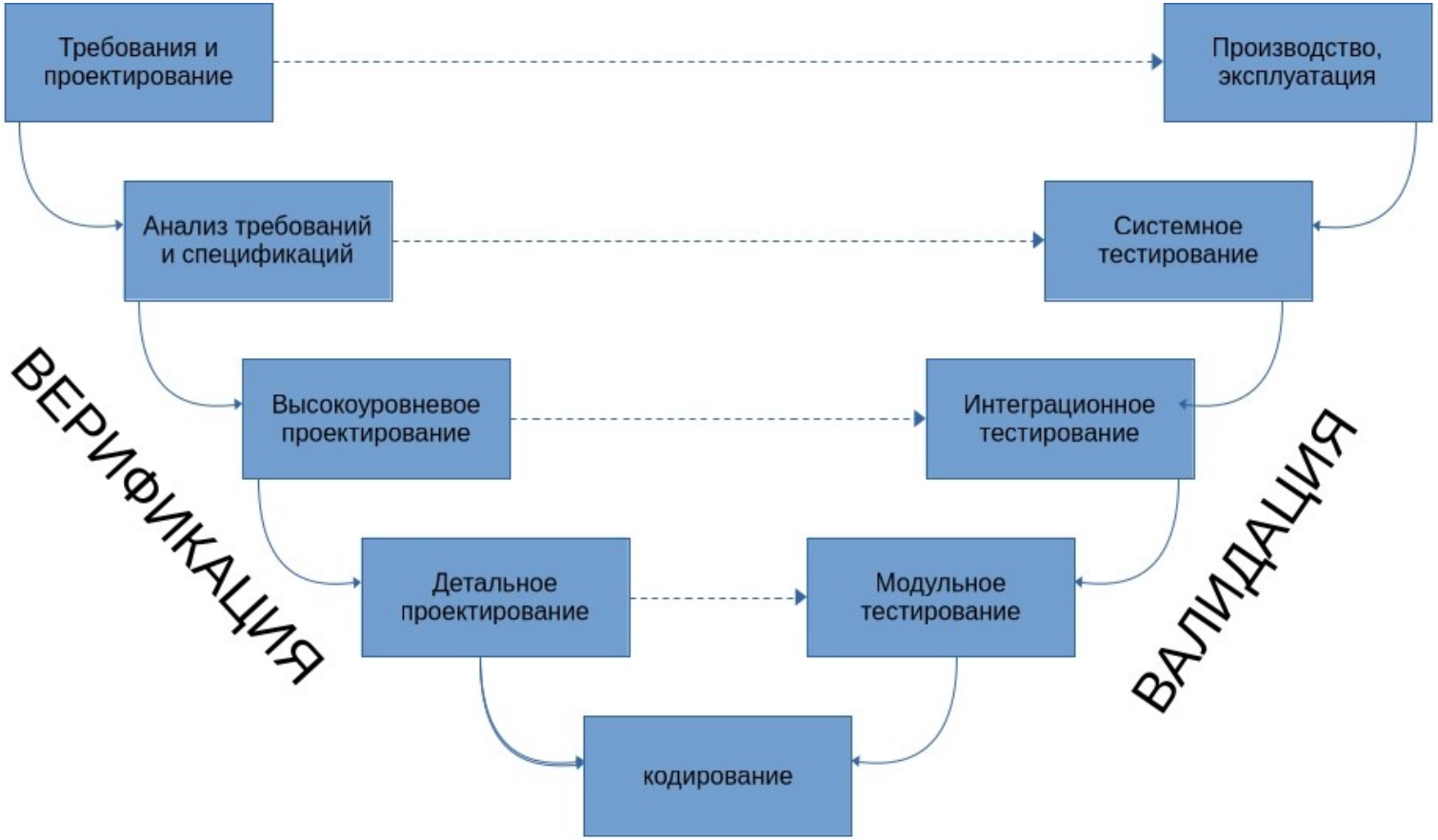
- Перед стартом нужно подготовить обширную техническую документацию, что занимает много времени;
- Сложно учесть все требования до старта разработки;
- Нет гибкости — если появились новые требования на этапе разработке, то откатить работу назад не получится;
- Заказчик видит результат в конце разработки — если итог его не устроит, придется начинать сначала.

Модели разработки. Каскадная модель

Когда подходит:

- Есть четкие и заранее известные требования, которые не будут меняться на этапе разработки.
- Работа должна идти по строгим регламентам — такое требование может быть актуальным для госучреждений или банковских систем.
- Разработка электроники/hardware (bonus)

Модели разработки. V-модель



Модели разработки. V-модель

Плюсы:

- Задачи по разработке и написанию тестов решаются параллельно
- Ошибки выявляют на ранних этапах, что снижает затраты на их исправление;
- За счет четкой структуры подходит для проектов с ясными и неизменными требованиями;
- Акцент на тестировании обеспечивает надежность продукта;
- Высокая степень анализа рисков.

Недостатки:

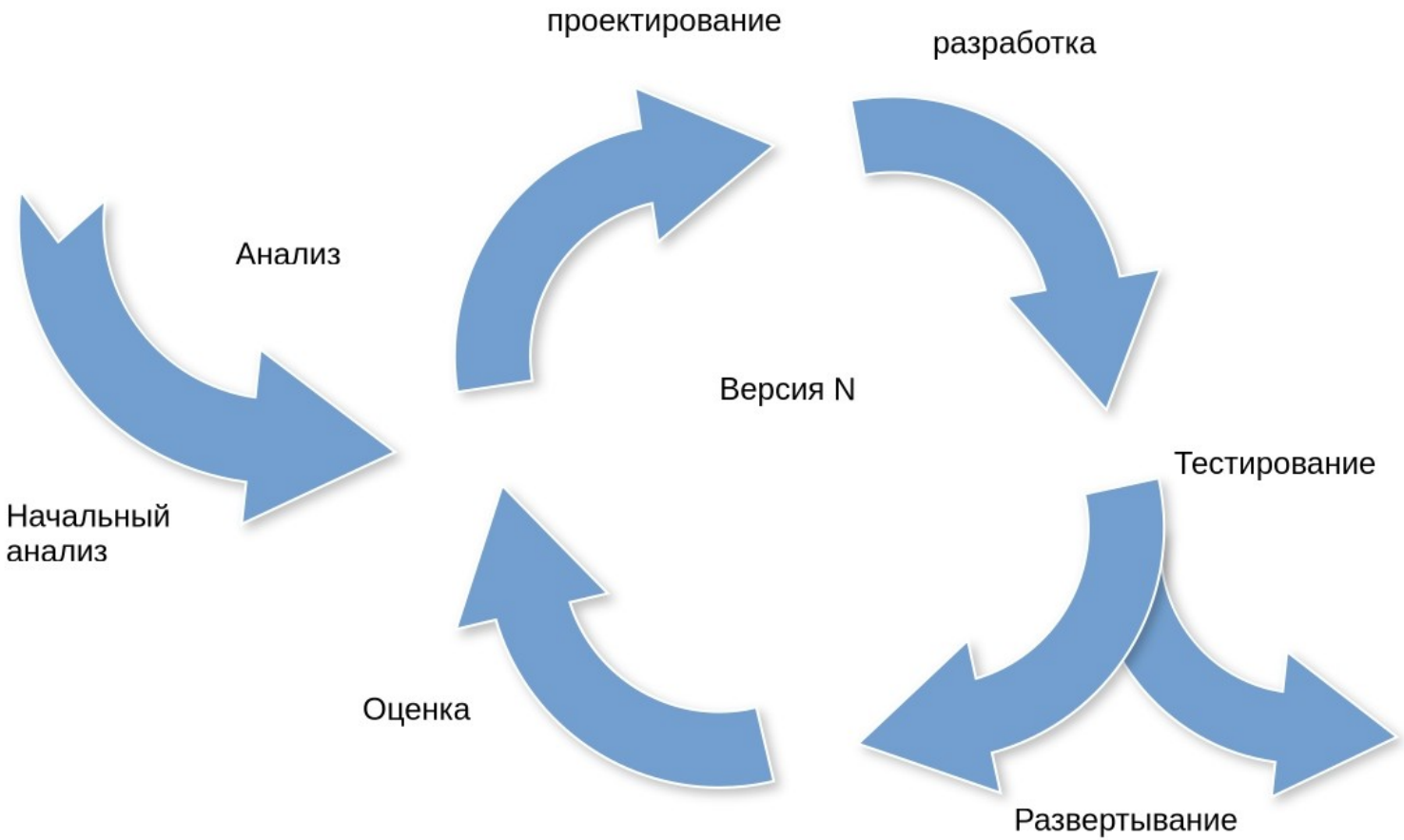
- Такая же строгая, как и каскадная, поэтому при появлении новых требований все придется начинать заново;
- Дорого

Модели разработки. V-модель

Когда подходит:

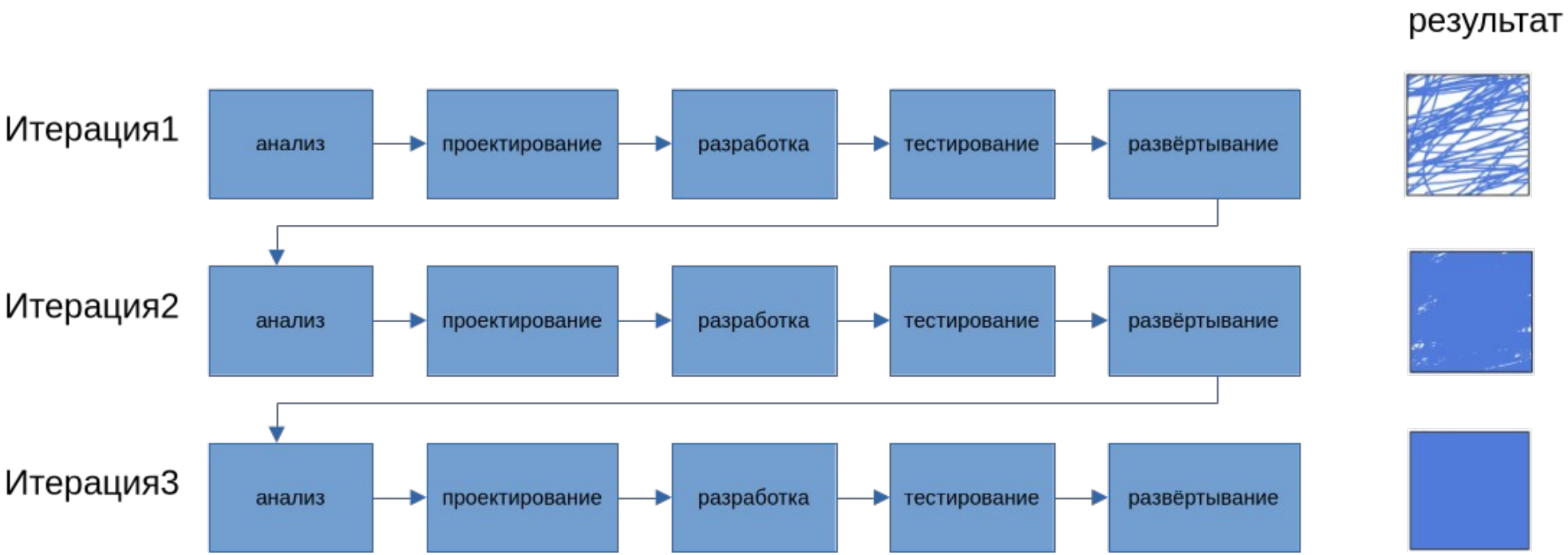
- Есть четкие и заранее известные требования, которые не будут меняться на этапе разработки.
- Работа должна идти по строгим регламентам
- Проект предполагает большое тестовое покрытие.

Модели разработки. Итеративная и инкрементальная модель



Bonus: https://habr.com/en/post/174845/

Модели разработки. Итеративная модель



Модели разработки. Итеративная модель

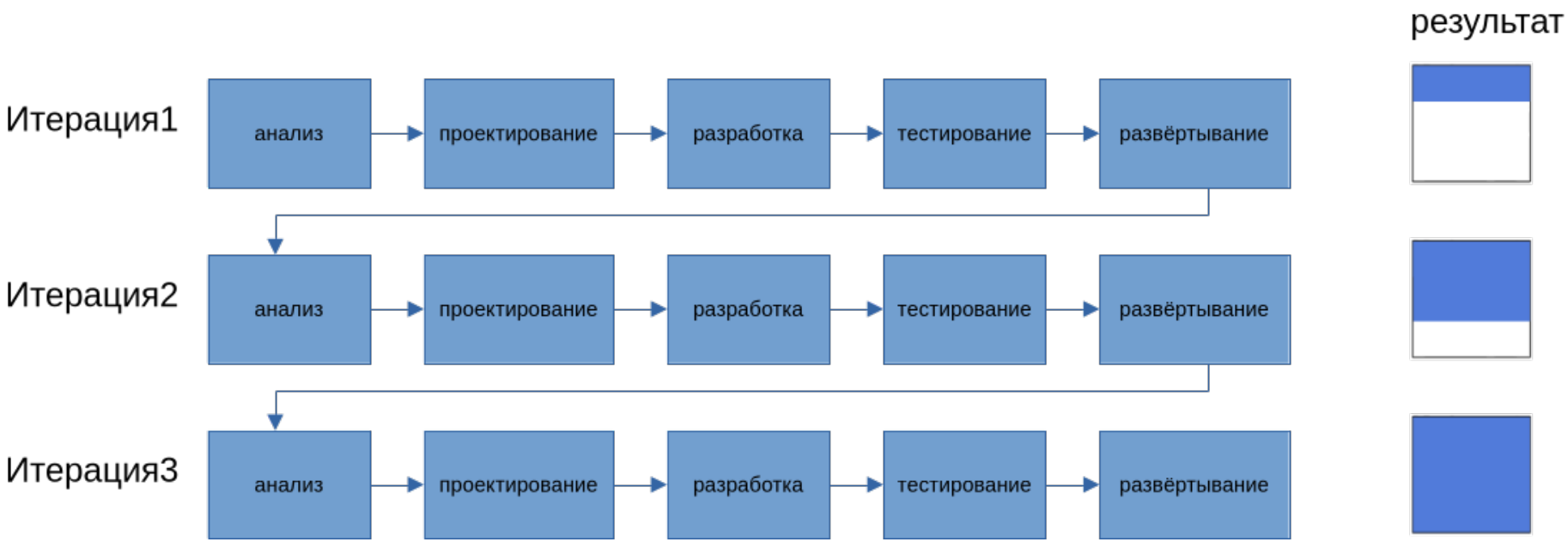
Плюсы:

- Возможность быстро выкатить ПО;
- Заказчик видит промежуточные результаты и может корректировать требования;
- Быстрая обратная связь от пользователей;
- Проще находить конфликты между требованиями.

Минусы:

- Могут возникнуть сложности с созданием рабочей архитектуры, так как изначально не всегда известны все требования.

Модели разработки. Инкрементальная модель



Модели разработки. Инкрементальная модель

Плюсы:

- Пользователи могут начать использовать продукт уже после первого инкремента;
- Заказчик видит прогресс и может вносить правки в следующие инкременты.

Минусы:

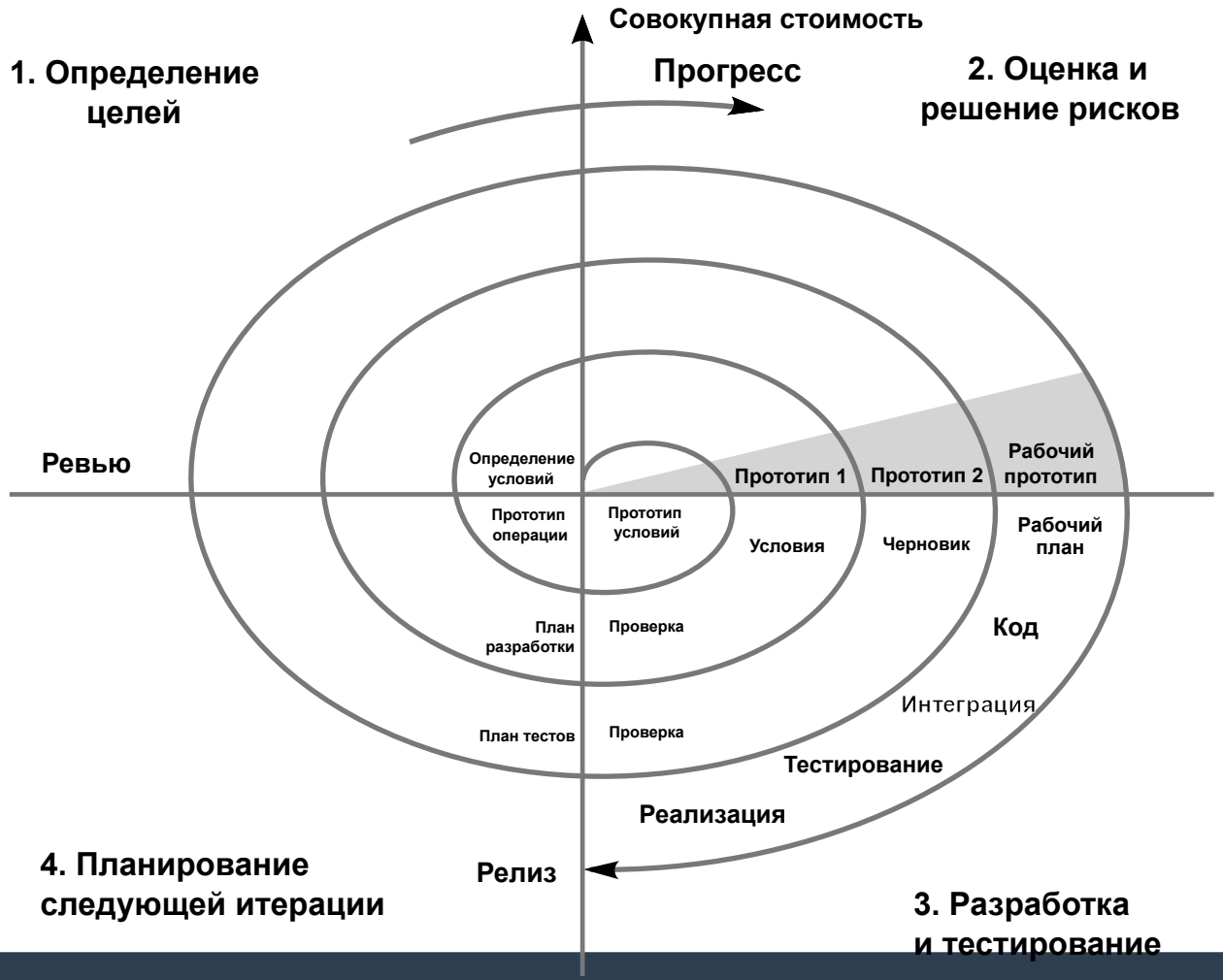
- Если команды параллельно работают над разными инкрементами, есть риск того, что модули будет сложно связать;
- Необходимо тщательное планирование, чтобы заранее определить, какие функции будут в каждом инкременте, и учесть их в архитектуре.

Модели разработки. Итеративная и инкрементальная модель

Когда подходит:

- Рабочее решение нужно в короткие сроки;
- Требования не до конца ясны и могут меняться в процессе работы;
- Проект большой, а ресурсов немного, поэтому приходится разбивать продукт на части и делать его постепенно.

Модели разработки. Спиральная модель



Модели разработки. Спиральная модель

Плюсы:

- Риски находят на ранних этапах, что снижает вероятность проблем в будущем;
- Можно вносить изменения в проект по мере его развития;
- Подходит для крупных систем, где много неопределённости и высокие требования к надёжности.

Минусы:

- Многочисленные циклы растягивают разработку и делают её дороже;
- Каждое новое требование заказчика запускает новый виток. Это делает разработку дороже, так как нужно снова тратить ресурсы на анализ.

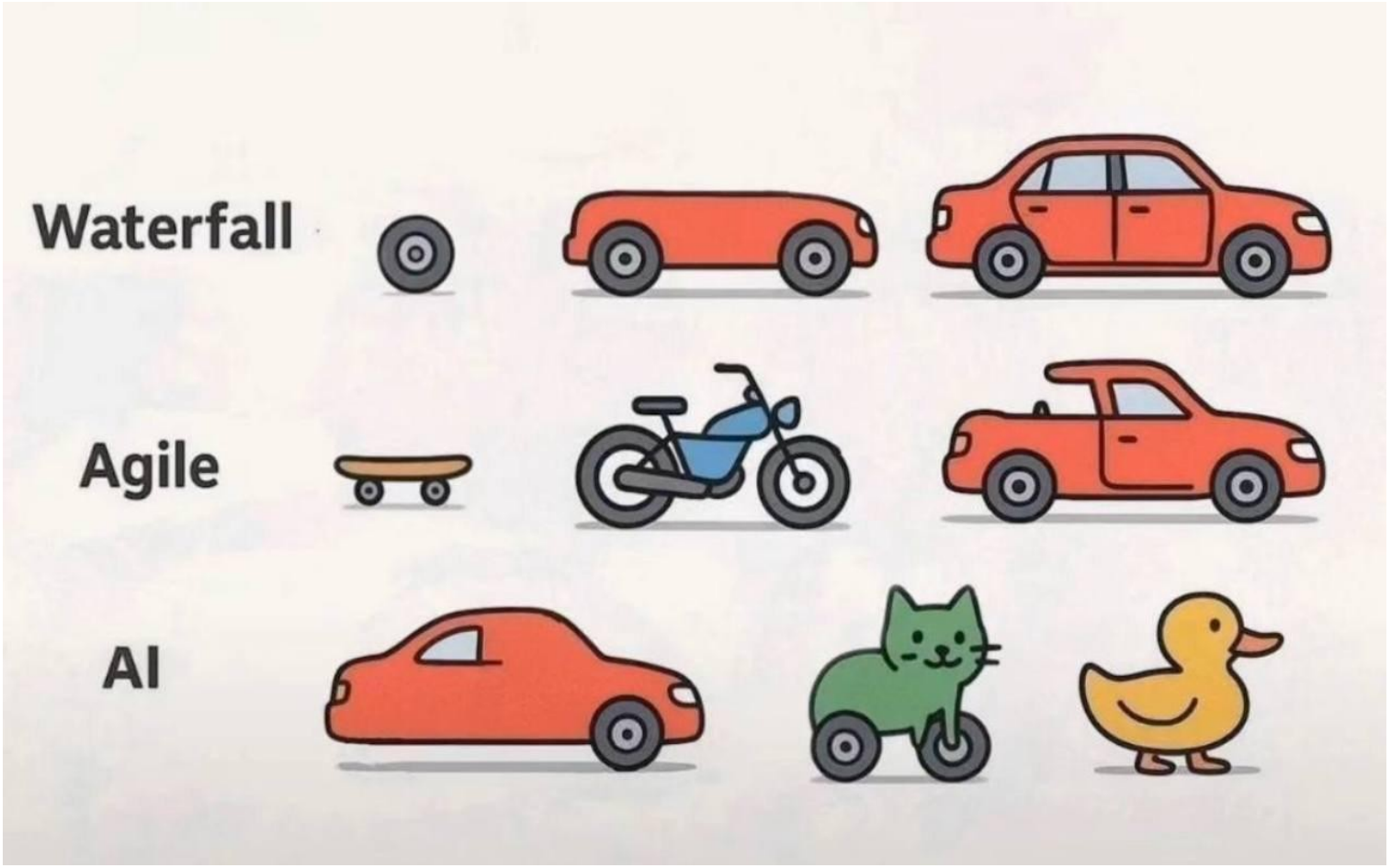
Модели разработки. Спиральная модель

Когда подходит:

- Пользователи сами не до конца понимают, что им нужно;
- Требования к проекту слишком сложные и могут меняться по ходу работы;
- Успех проекта не гарантирован, и нужно заранее оценить риски, чтобы решить, стоит ли продолжать работу;
- В проекте используют новые технологии, которые еще не до конца изучены, и есть риск, что они не дадут ожидаемого результата.

Модели разработки. Гибкая методология разработки (Agile)

Люди и их взаимодействие	важнее чем	Процессы и инструменты
Работающее ПО	важнее чем	Исчерпывающая документация
Взаимодействие с заказчиком	важнее чем	Обсуждения контракта
Реагировать на изменения	важнее чем	Следовать плану


Waterfall
Agile
AI

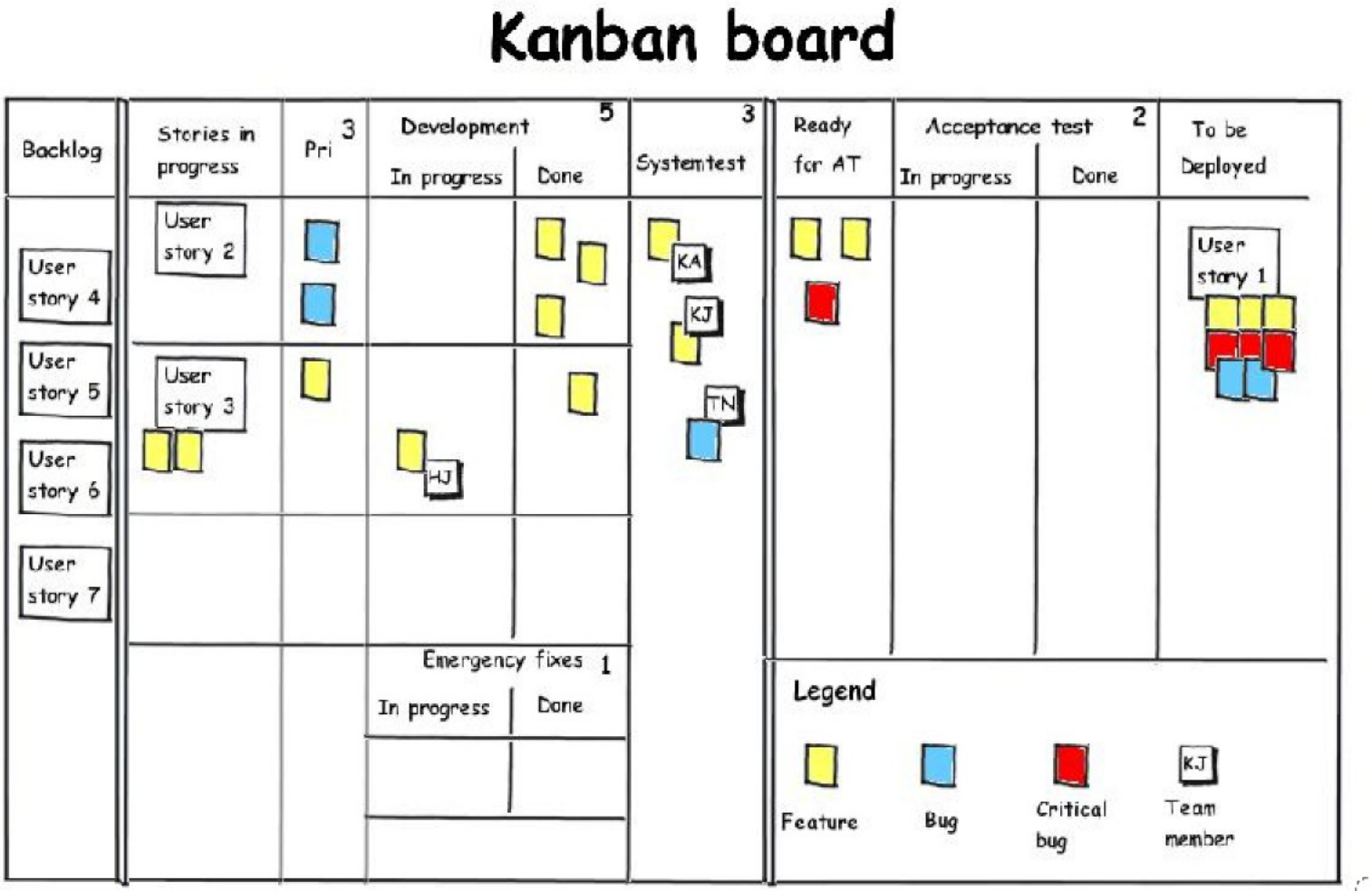
Методологии разработки ПО. Kanban

Основные подходы:
- Визуализация процесса
- Ограничение на количество ведущейся работы
-

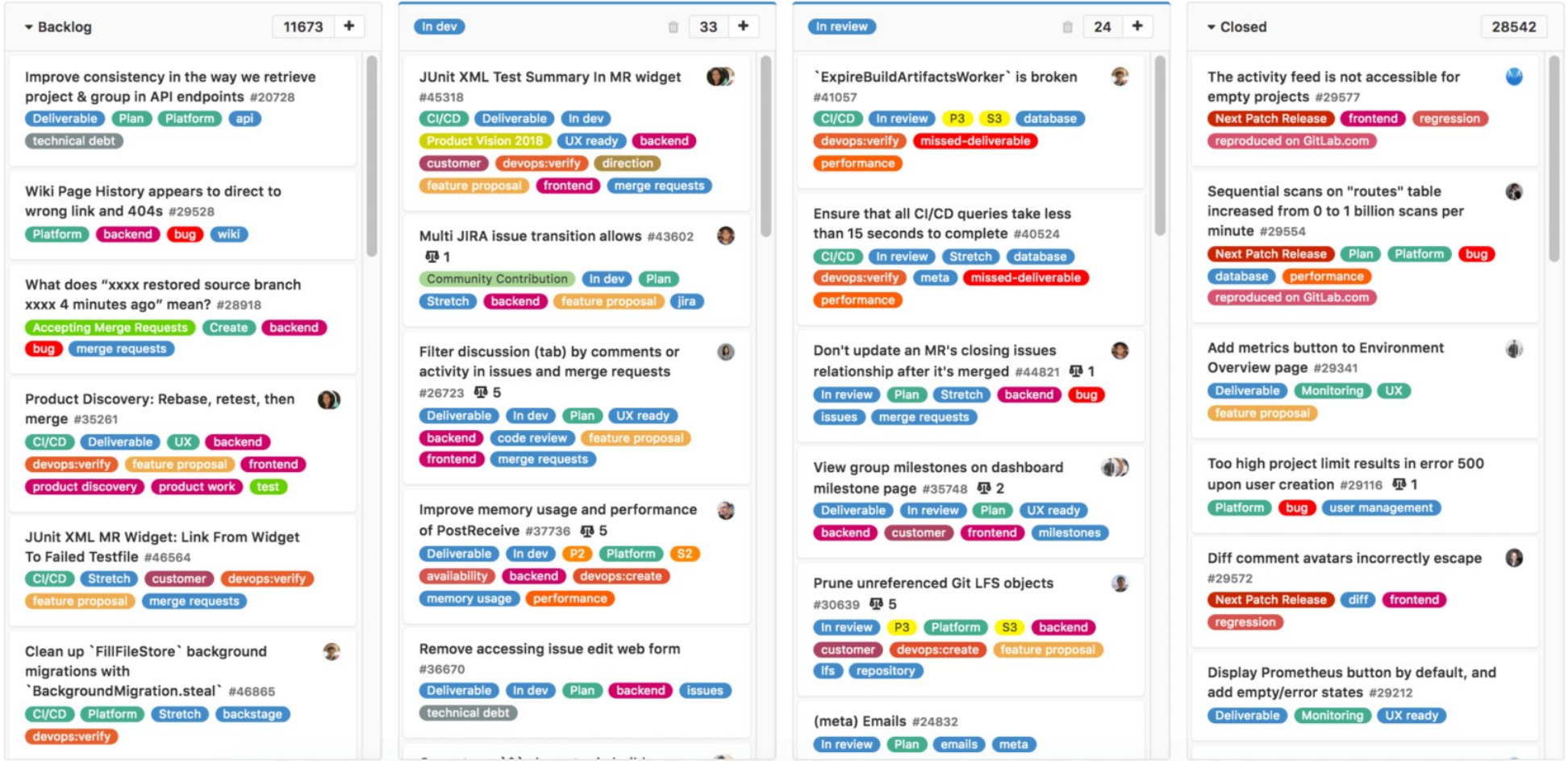
Принципы:
- Базирование на существующих методах разработки
- Сохранение существующего порядка, ролей и обязанностей
- Предварительная договорённость о проведении изменений
- Поощрение инициативы

Bonus: https://habr.com/en/post/230725/

Методологии разработки ПО. Kanban



Методологии разработки ПО. Kanban



Методологии разработки ПО. Scrum

Основные элементы:

- Скрам-команды;
- роли с ними связанные;
- события;
- артефакты;
- правила.

Scrum. Scrum-команда

Небольшая команда, в которой есть

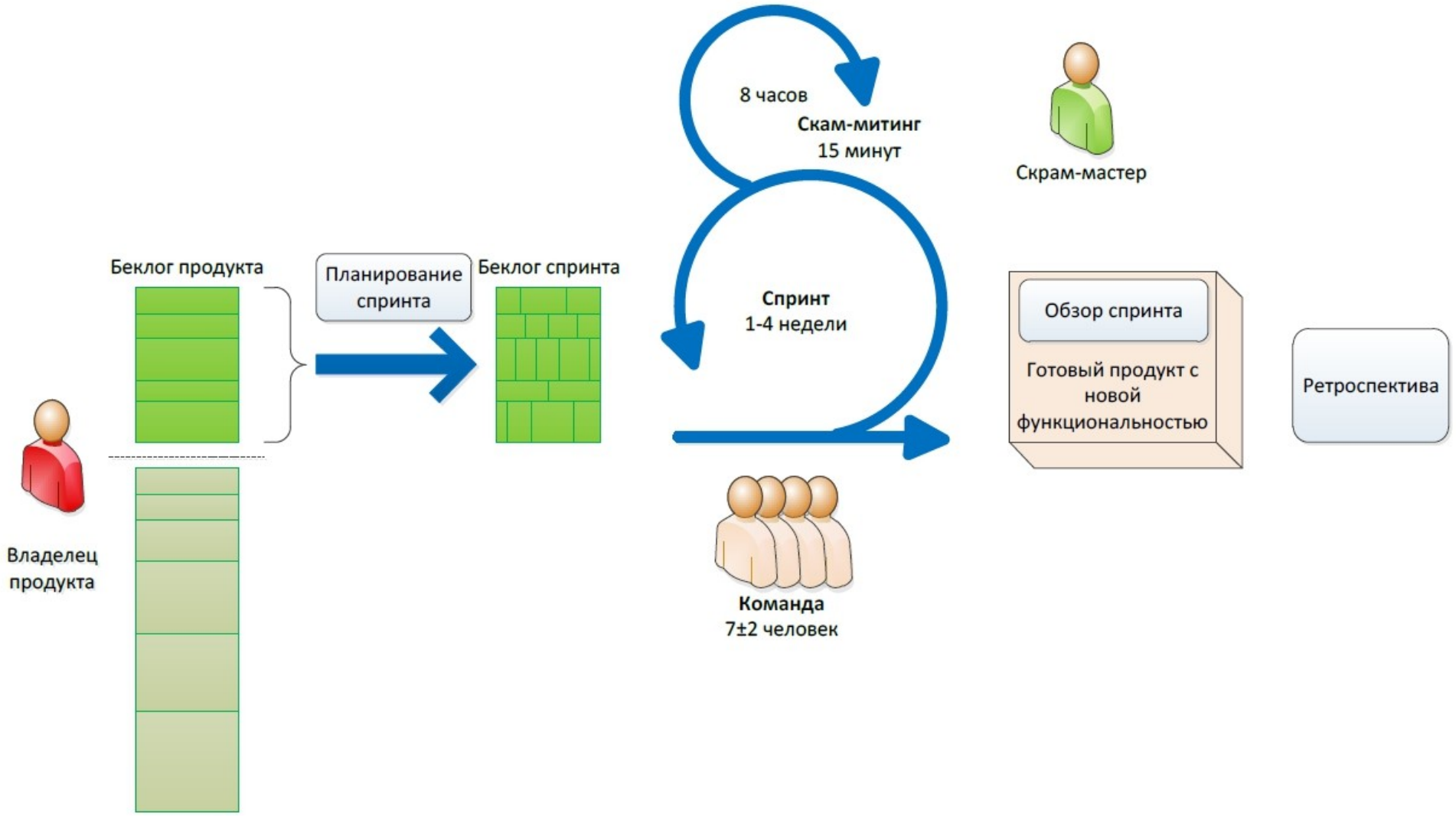
- Scrum-мастер;
- владелец продукта;
- команда разработки (7-9 человек).

Scrum. Scrum-события

- Планирование Спринта
- Ежедневный Скрам
- Обзор Спринта
- Ретроспектива Спринта

Scrum. Scrum-Артефакты

- Бэклог Продукта
- Бэклог Спринта
- Инкремент

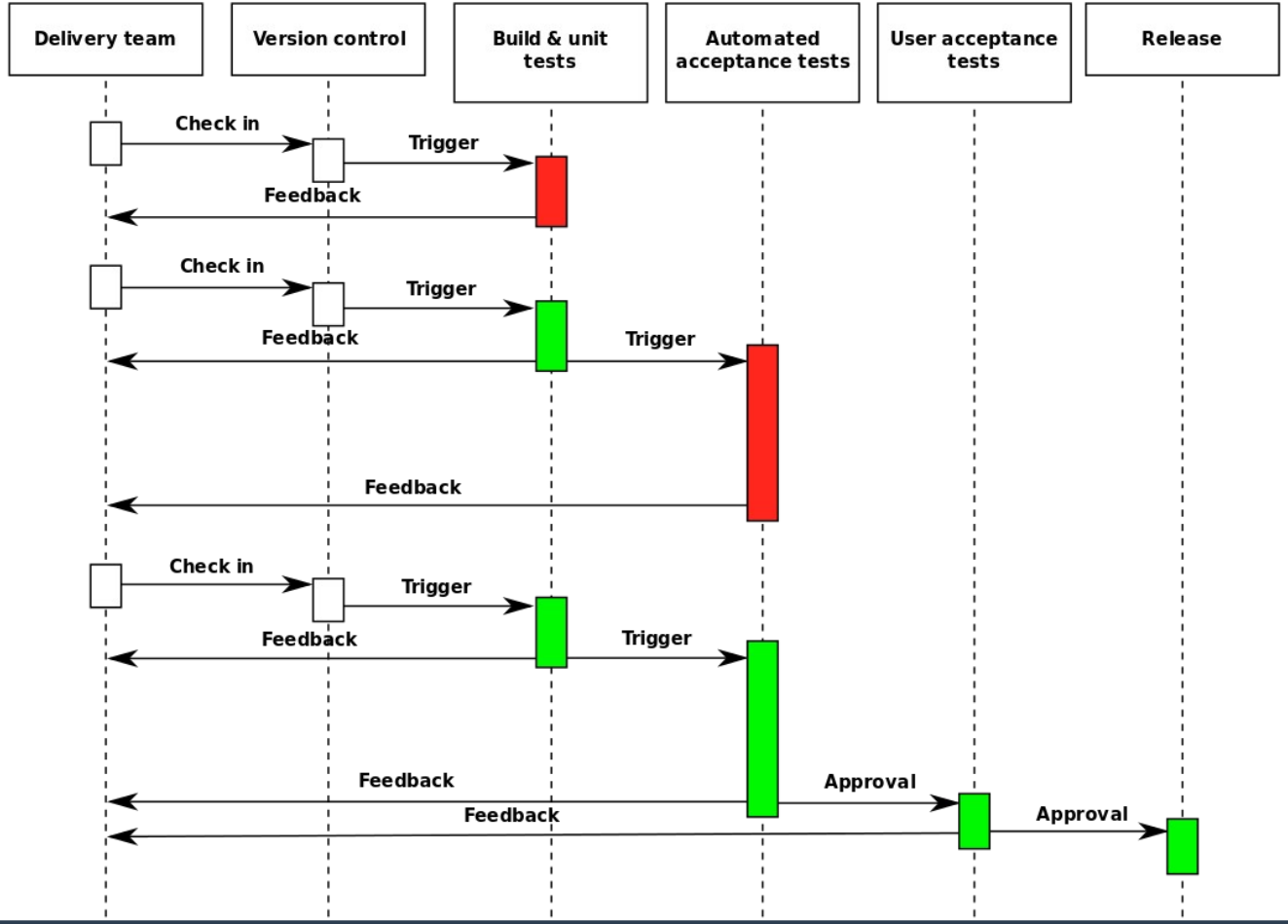


Bonus: https://habr.com/en/post/430890/

Прочие полезные практики

- Code-review (ревизия кода)
- разработка через тестирование (TTD — Test driven development)
- CI, CD (continious integration, continious delivery)
- Рефакторинг

Continious Deployment



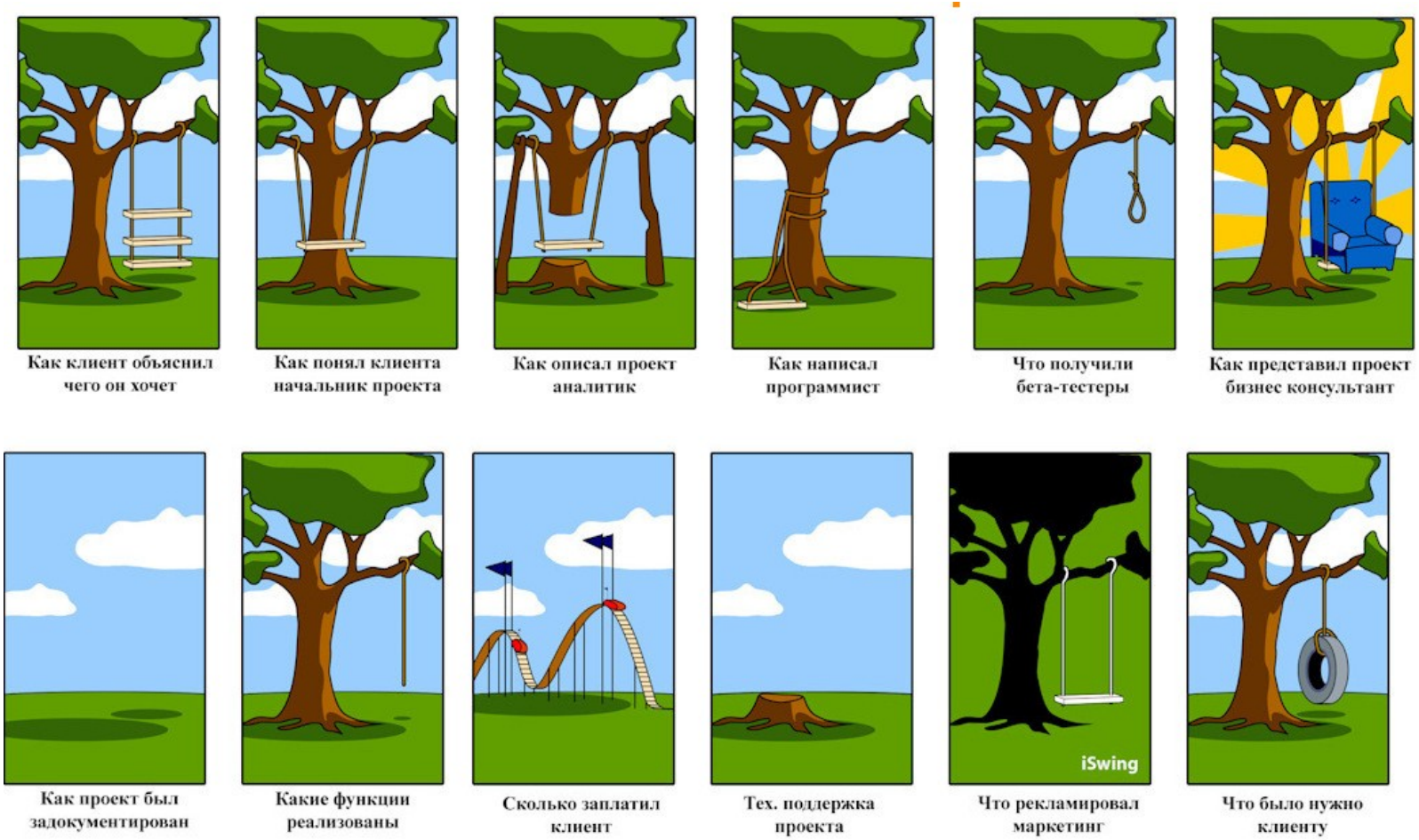
Что обычно думают программисты по поводу моделей разработки?

http://macode.ru/
http://programming-motherfucker.com/

Итоги

- Нет универсальной методологии/модели, которая одинаково хорошо подошла бы всем.
- Чем сильнее зарегулировать процесс разработки, тем более «предсказуемым» он станет (но предсказуемость != эффективность)
- Существуют практики позволяющие ускорить процесс разработки

Стандарты в области разработки ПО. А зачем?



Стандарты в области разработки ПО

Стандарты:

- Международные
  - **ISO** — International Organization for Standardization
  - **IEC** — International Electrotechnical Commission
  - **IEEE** — Institute of Electrical and Electronics Engineers
  - **RFC (IETF)** — request for comments (Internet Engineering Task Force)
  - …
- Региональные
  - **ГОСТ**
  - **ANSI** — American National Standards Institute
  - **NIST** — National Institute of Standards and Technology
  - **…**

ГОСТы

Для написания документации на программу используются две серии ГОСТов:

- **ГОСТ 19** Единая система программной документации (ЕСПД)
- **ГОСТ 34** Информационные технологии. Комплекс стандартов на автоматизированные системы

Процесс разработки ПО описан в **ГОСТ Р ИСО/МЭК 12207**

ГОСТ (государственный стандарт) – это нормативно-правовой документ, в соответствии требованиями которого производится стандартизация производственных процессов и оказания услуг.

Существуют ГОСТы, которые копируют содержимое международных стандартов. В имени этих ГОСТов указанно название международного стандарта, номер ГОСТа совпадает с оригинальным.

Государственный стандарт обязательно проходит процедуру регистрации, которая проводится специальным государственным органом - Госстандартом.

ГОСТы могут заменяться или отменяться.

Утвержденный ГОСТ содержит ключевые требования, которым должны соответствовать товары, работы и услуги, в отношении которых он принимается, для обеспечения их эффективной и безопасной эксплуатации.

Стандарты. ГОСТ 19. ЕСПД

ГОСТ 19.xxx Единая система программной документации (ЕСПД) — это комплекс государственных стандартов, устанавливающих взаимоувязанные правила разработки, оформления и обращения программ (или ПО) и программной документации. Т.е. этот стандарт относится к разработке именно ПО.

Определения из ЕСПД:

- Программа — данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определённого алгоритма.
- Программное обеспечение — совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

Стандарты. ГОСТ 34. ИТ

Определение из ГОСТ 34

• Автоматизированная система (АС) — система, состоящая из персонала и комплекса средств автоматизации его деятельности, реализующая информационную технологию выполнения установленных функций. В зависимости от вида деятельности выделяют, например, следующие виды АС: автоматизированные системы управления (АСУ), системы автоматизированного проектирования (САПР), автоматизированные системы научных исследований (АСНИ) и другие.

ГОСТ 34 разделяет виды обеспечения АС:

• Организационное;

• Методическое;

• Техническое;

• Математическое;

• Программное обеспечение;

• Информационное;

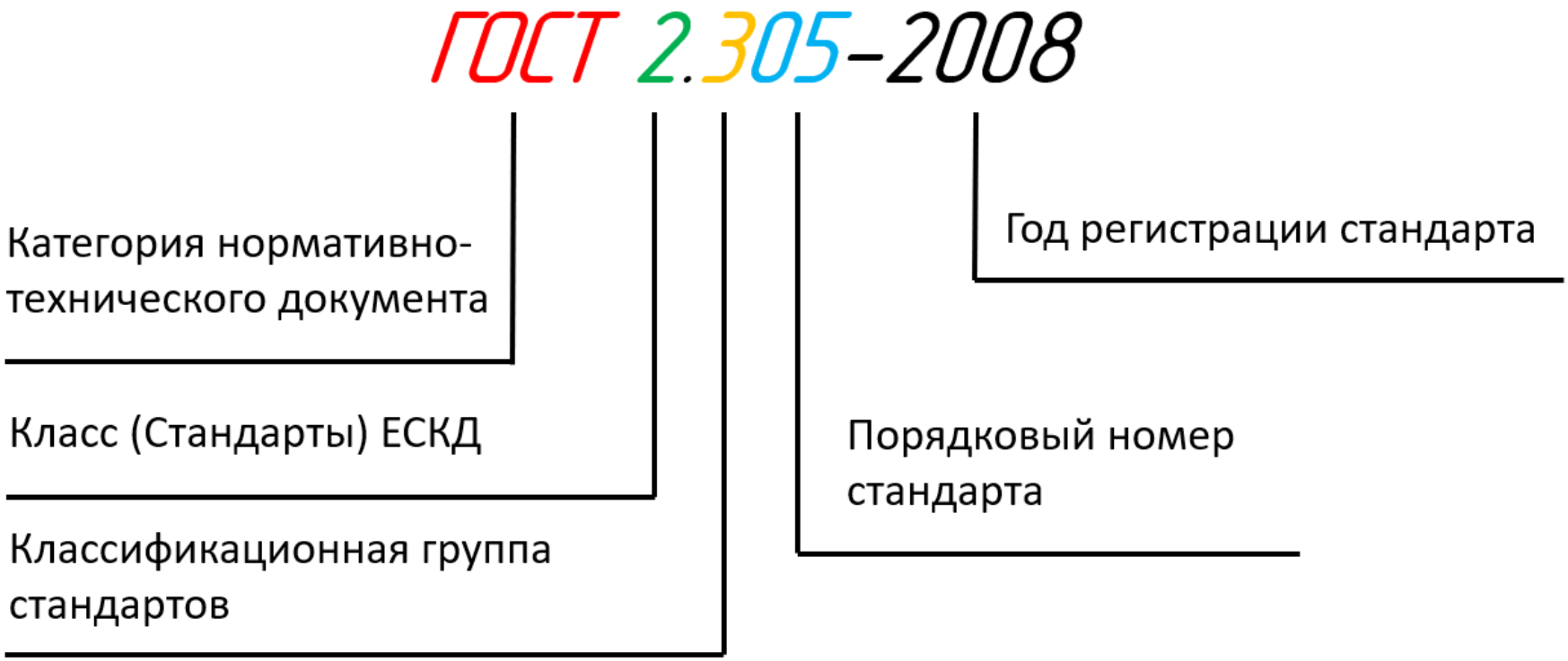
• Лингвистическое;

• Правовое;

• Эргономическое

NOTE: Автоматизированная система — это не программа, а комплекс видов обеспечения, среди которых есть и программное обеспечение

Стандарты. ГОСТ



ГОСТ 19 vs ГОСТ 34

Автоматизированная система, как правило, содержит организационное решение под конкретного пользователя и заказчика

Программа может быть создана и растиражирована под большое количество пользователей без привязки к какому-либо предприятию/

Если разрабатывается документация на программу, которую создают под конкретное предприятие, то используется ГОСТ 34.

Если разрабатывается документация на массовую программу, то используется ГОСТ 19.

Пункты технического задания ГОСТа 34 и ГОСТа 19 отличаются.

Стандарты. ГОСТ 19.xxx

Номер	Описание
ГОСТ 19.001-77	Общие положения
ГОСТ 19781-90	Термины и определения.
ГОСТ 19.101-77	Виды программ и программных документов
ГОСТ 19.102-77	Стадии разработки
ГОСТ 19.103-77	Обозначения программ и программных документов
ГОСТ 19.104-78	Основные надписи
ГОСТ 19.105-78	Общие требования к программным документам
ГОСТ 19.106-78	Требования к программным документам, выполненным печатным способом
ГОСТ 19.201-78	Техническое задание, требования к содержанию и оформлению
ГОСТ 19.202-78	Спецификация. Требования к содержанию и оформлению
ГОСТ 19.301-79	Программа и методика испытаний. Требования к содержанию и оформлению
ГОСТ 19.401-78	Текст программы. Требования к содержанию и оформлению
ГОСТ 19.402-78	Описание программы

Стандарты. ГОСТ 19.xxx

Номер	Описание
ГОСТ 19.403-79	Ведомость держателей подлинников
ГОСТ 19.404-79	Пояснительная записка. Требования к содержанию и оформлению
ГОСТ 19.501-78	Формуляр. Требования к содержанию и оформлению
ГОСТ 19.502-78	Описание применения. Требования к содержанию и оформлению
ГОСТ 19.503-79	Руководство системного программиста. Требования к содержанию и оформлению
ГОСТ 19.504-79	Руководство программиста. Требования к содержанию и оформлению
ГОСТ 19.505-79	Руководство оператора. Требования к содержанию и оформлению
ГОСТ 19.506-79	Описание языка. Требования к содержанию и оформлению
ГОСТ 19.507-79	Ведомость эксплуатационных документов
ГОСТ 19.508-79	Руководство по техническом обслуживанию. Требования к содержанию и оформлению
ГОСТ 19.601-78	Общие правила дублирования, учета и хранения
ГОСТ 19.602-78	Правила дублирования, учета и хранения программных документов, выполненых печатным способом
ГОСТ 19.603-78	Общие правила внесения изменений
ГОСТ 19.604-78	Правила внесения изменений в программные документы, выполненных печатным способом

Стандарты. Жизненный цикл ПО. ГОСТ Р ИСО/МЭК 12207

Процессы соглашения (6.1)
Приобретение
Поставка

Процессы организационного обеспечения проекта (6.2)
Менеджмент модели жизненного цикла
Менеджмент инфраструктуры
Менеджмент портфеля проектов
Менеджмента людских ресурсов
Менеджмент качества

Процессы проекта (6.3)
Планирование проекта
Оценка проекта и процесс управления
Менеджмент решений
Менеджмент рисков)
Менеджмент конфигурации
Менеджмент информации
Процесс измерений

Технические процессы (6.4)
Определение требований правообладателей
Анализ системных требований
Проектирование архитектуры системы
Процесс Реализации
Комплексирование системы
Квалифицированное тестирование системы
Инсталяция программных средств
Поддержка приёмки программных средств
Функционирование программных средств
Сопровождение программных средств
Прекращение применения программных средств

Процессы реализации ПС (7.1)
Реализация программных средств
Анализ требований программных средств
Проектирование архитектуры программных средств
Детальное проектирование программных средств
Конструирование программных средств
Комплексирование программных средств
Квалифицированное тестирование программных средств

Процессы повторного применения программных средств (7.3)
Проектирование доменов
Менеджмент повторного применения активов
Менеджмент повторного применения программ

Процесс поддержки ПС (7.2)
Менеджмент программной документации
Менеджмент конфигурации
Обеспечение гарантий качества программных средств
Верификация программных средств
Валидация программных средств
Ревизия программных средств
Аудит программных средств
Решение проблем в программных средствах

Чего почитать

- Ф. Брукс «Мифический человеко-месяц»
- С. Макконнелл «Совершенный код»