Лекция №5

Технологии разработки

Технология

Программная инженерия:

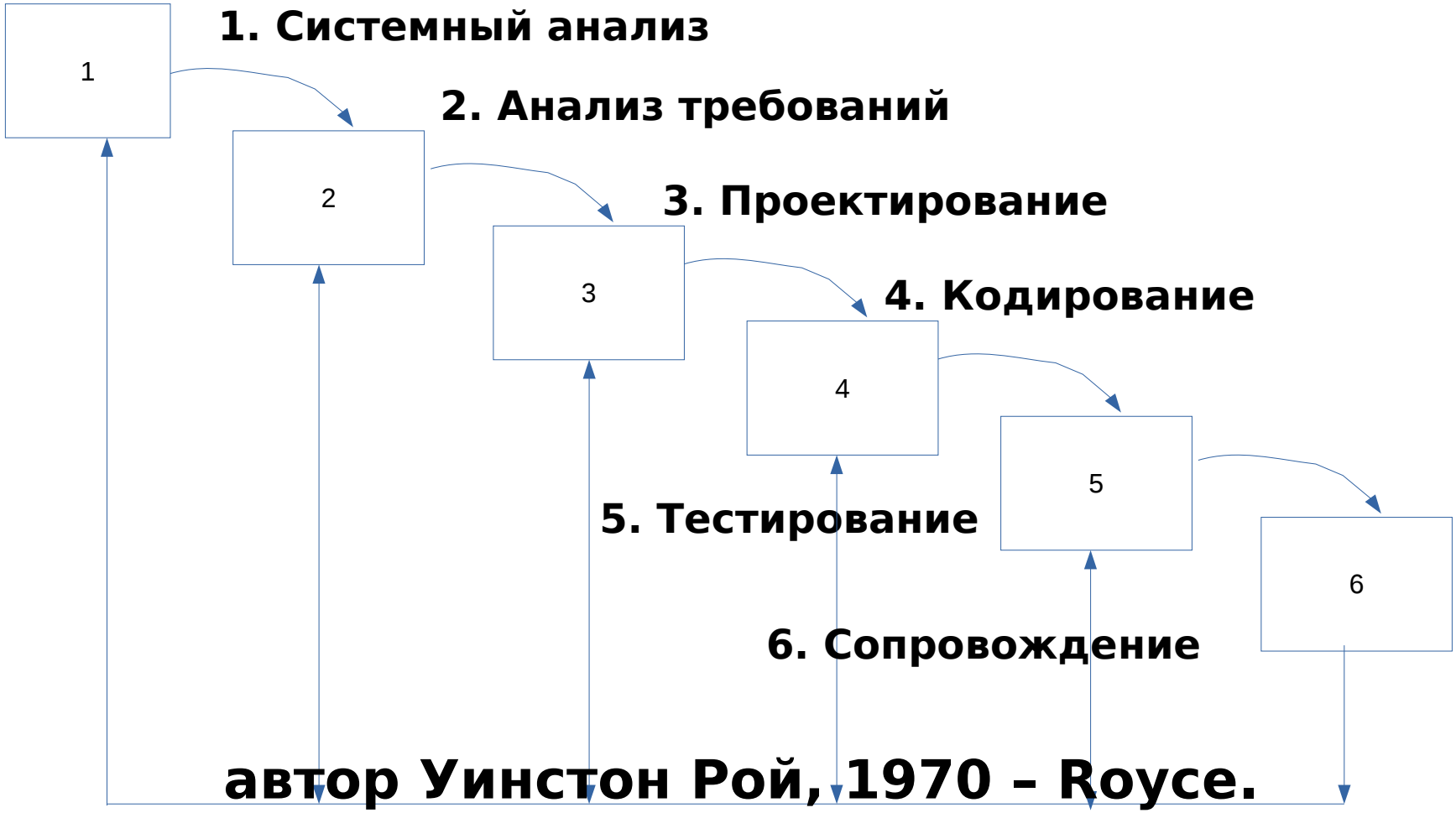
«…разработка крупных (масштабных, многокомпонентных, многоуровневых, разнотехологичных…) систем, большим коллективом разных (по функциональным обязанностям, по местоположению, по личностным характеристикам…) людей, предназначенных для эксплуатации и модернизации (масштабирования, интеграции, повышения «сорта и качества») в течении длительного периода времени

разработка = анализ + проектирование + программирование (кодирование) + тестирование + отладка

Этапы процесса разработки

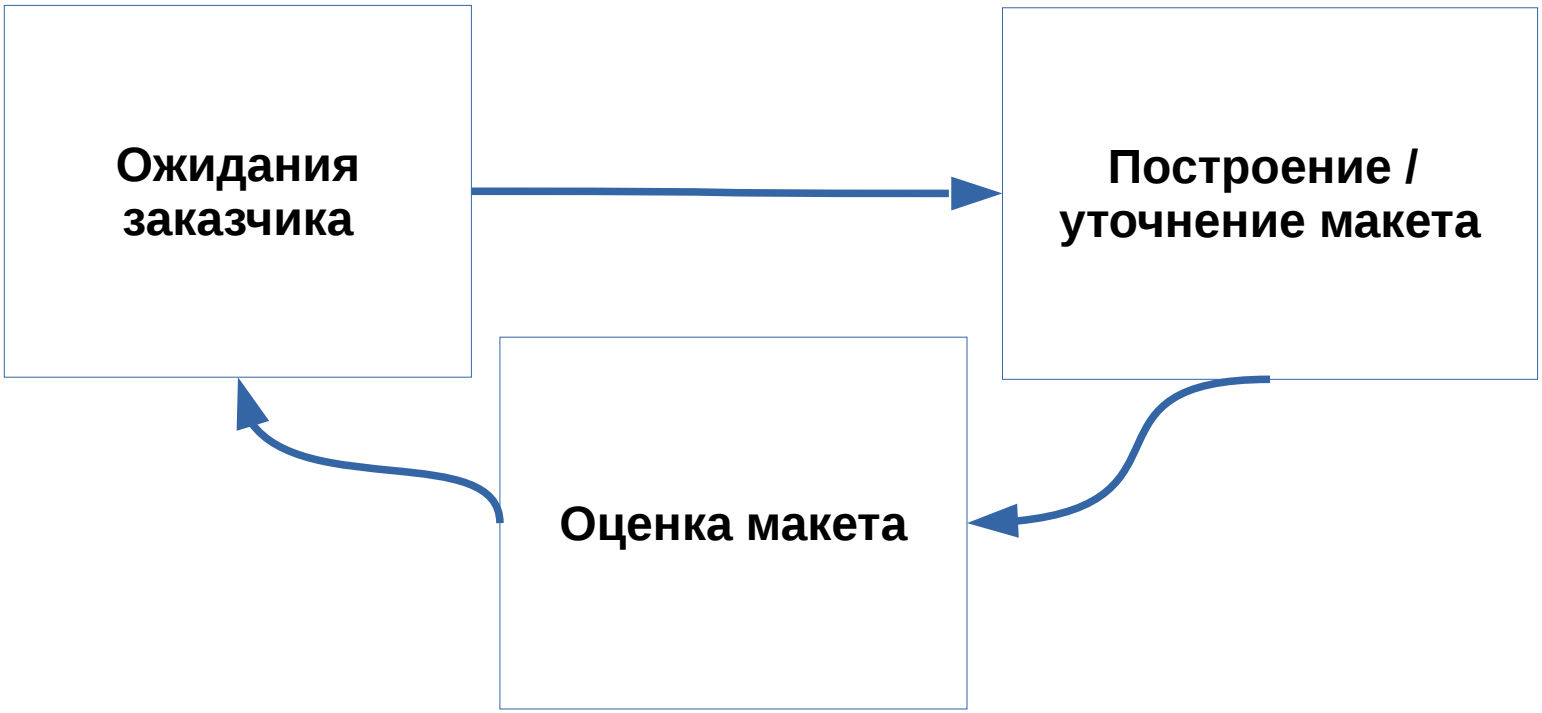
1. Определение процесса разработки
2. Управление проектом разработки
3. Описание целевого программного продукта
4. Проектирование
5. Разработка
6. Тестирование частей
7. Интеграция частей и тестирование в целом
8. Сопровождение

жизненный цикл



автор Уинстон Рой, 1970 – Royce. Managing the development of large software systems: concepts and techniques

макетирование
прототипирование



инкрементальная модель

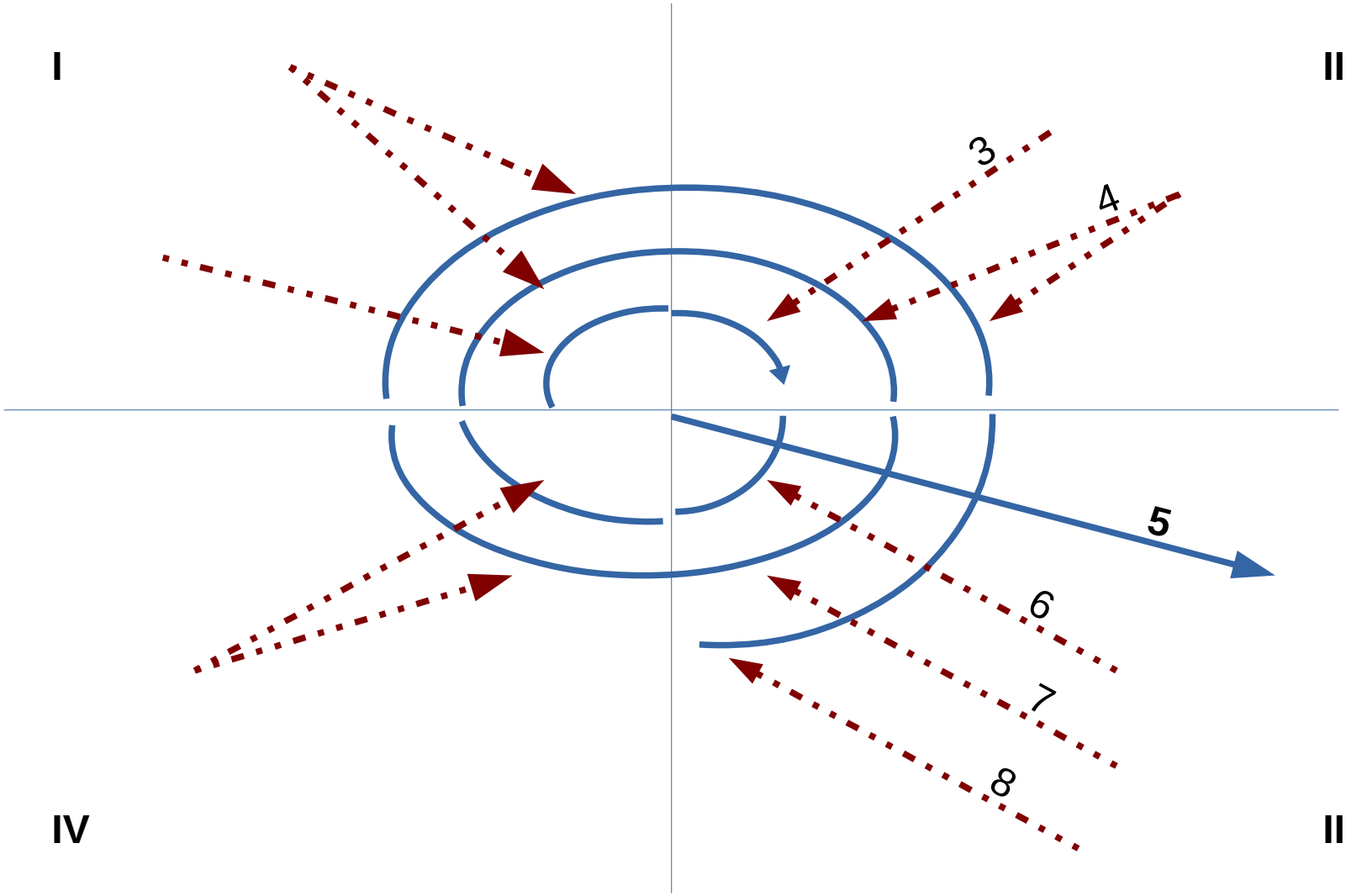
RAD

RAPID APPLICATION DEVELOPMENT

1. Бизнес-моделирование
2. Моделирование данных
3. Моделирование обработки
4. Генерация приложения
5. Тестирование и объединение

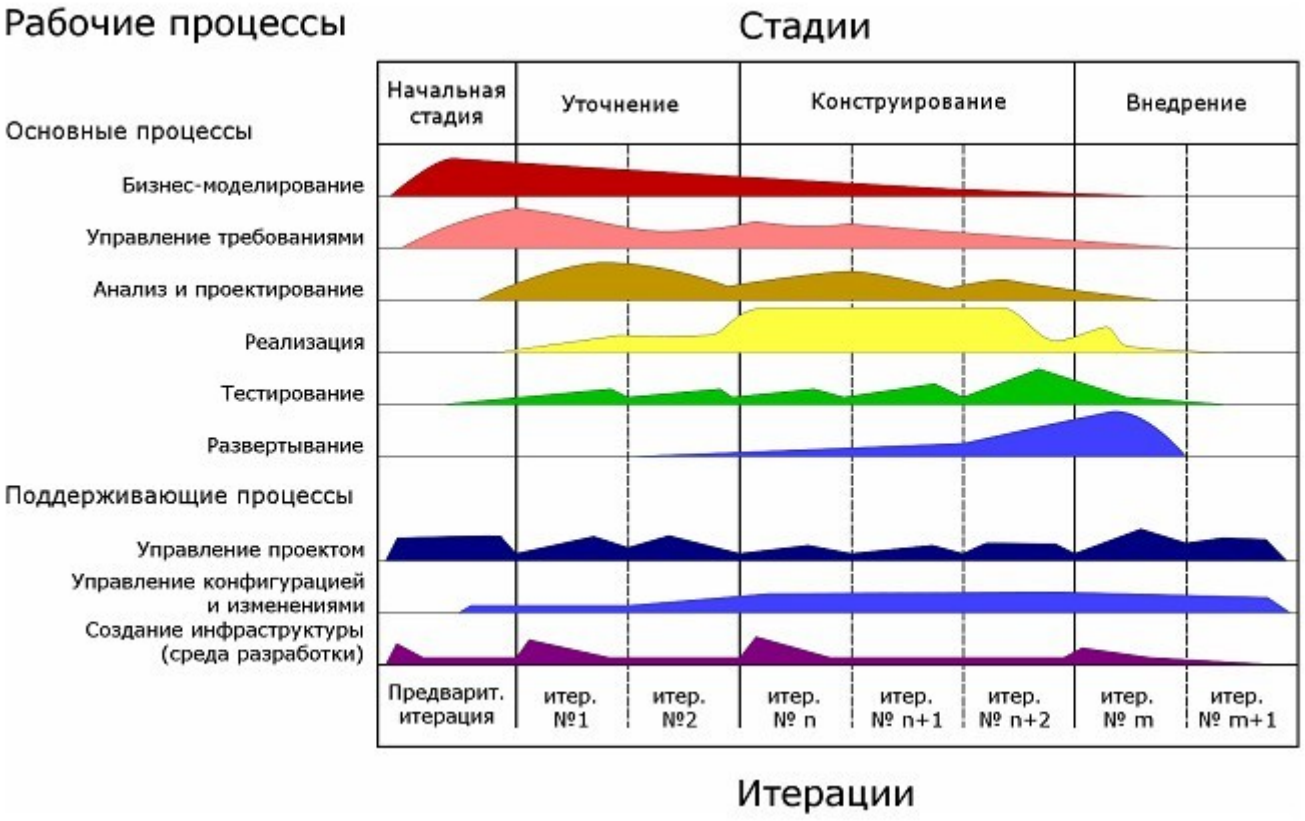
спиральная модель

Барри Боэм, 1988



USDP Якобсон, Буч и Рамбо, 1999

UNIFIED SOFTWARE DEVELOPMENT PROCESS



Гибкие технологии

Гибкая методология разработки ориентирована на использование итеративного подхода, при котором продукт создается постепенно, небольшими шагами, включающими реализация определенного набора требований.

Ключевые постулаты гибкой разработки

- Люди и их взаимодействие
- Создание работающего программного обеспечения
- Сотрудничество с заказчиком
- Реакция на изменение

Принципы гибкой методологии

1) Высшим приоритетом следует считать удовлетворение пожеланий заказчика
2) Не игнорировать изменения требований
3) Частое создание новых работающих версии ПО
4) Заказчики и разработчики должны работать совместно
5) Проекты должны воплощать в жизнь целеустремленные люди
6) Эффективный метод передачи информации – разговор лицом к лицу

Принципы гибкой методологии (продолжение)

7) Работающая программа – основной показатель прогресса в проекте
8) *Гибкие процессы способствуют долгосрочной разработке*
9) Непрестанное внимание к качественному проектированию
10) Простота
11) *Самые лучшие решения выдают самоорганизующиеся команды*
12) Команда должна регулярно задумываться над тем, как стать ещё более эффективной

Гибкие методологии

Agile Modeling	Adaptive software development
Agile Unified Process	Feature driven development
OpenUP	Getting Real
Agile Data Method	MSF fog Agile Software Development
DSDM	**Scrum**
Extreme programming	

XP

Кент Бек, 1999

eXtreme Programming

	XP-экстремум	*XP-реализация*
Проверка кода	Код проверяется всё время	Парное программирование
Тестирование	Тестирование выполняется всё время, даже с помощью заказчика	Тестирование модуля, функциональное тестирование

XP

Проектирование	Проектирование – часть ежедневной работы разработчика	Реорганизация (refactoring)
Простота	Для системы выбирается простейшее проектное решение, поддерживающее ее текущую функциональность	Самая простая вещь, которая могла бы работать. Принцип KISS. «Это вам не понадобится»
Архитектура	Каждый постоянно работает над уточнением архитектуры	Метафора

XP

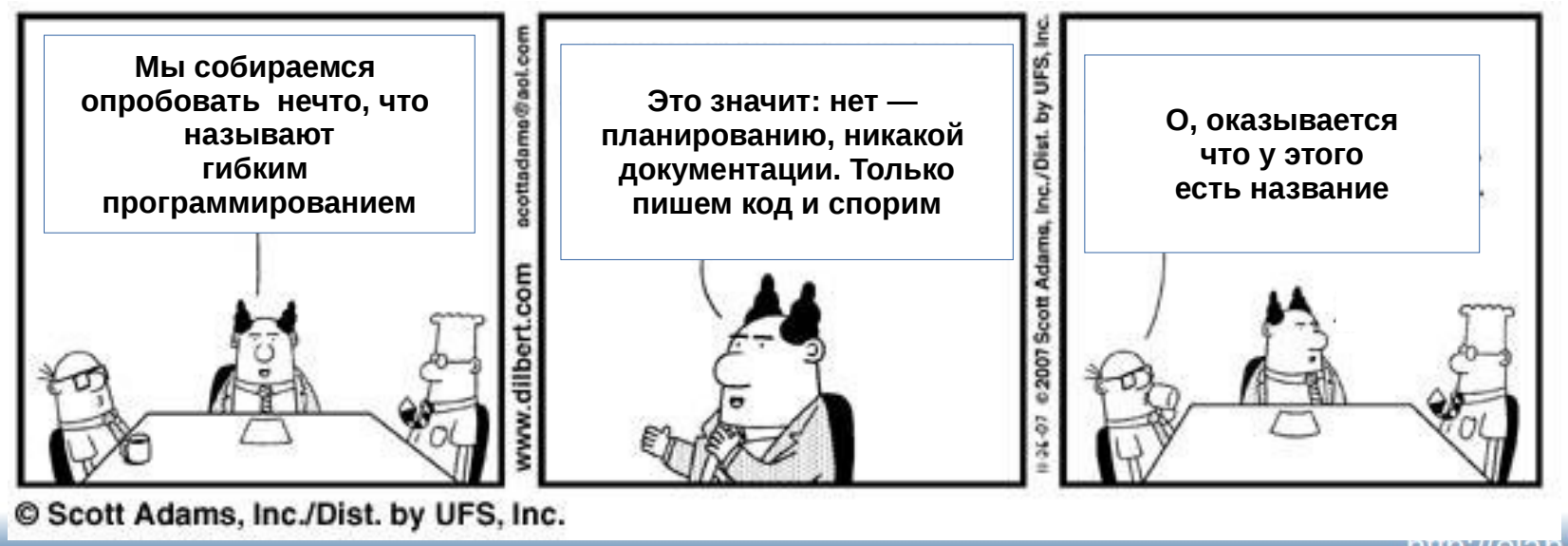
Тестирование интеграции	Интегрируется и тестируется несколько раз в день	Непрерывная интеграция
Короткие итерации	Итерации предельно коротки – секунды, минуты, часы, а не неделя, месяц, год	Игра планирования

базис XP

1)Игра планирования (*Planning game*)
2)Частая смена версий (*Small releases*)
3)Метафора (*Metaphor*)
4)Простое проектирование
5)Тестирование (*TDD - Test Driven Development*)
6)Реорганизация (*Refactoring*)
7)Парное программирование
8)Коллективное владение кодом
9)Непрерывная интеграция.
10)40-часовая неделя.
11)Локальный заказчик.
12)Стандарты кодирования

Agile

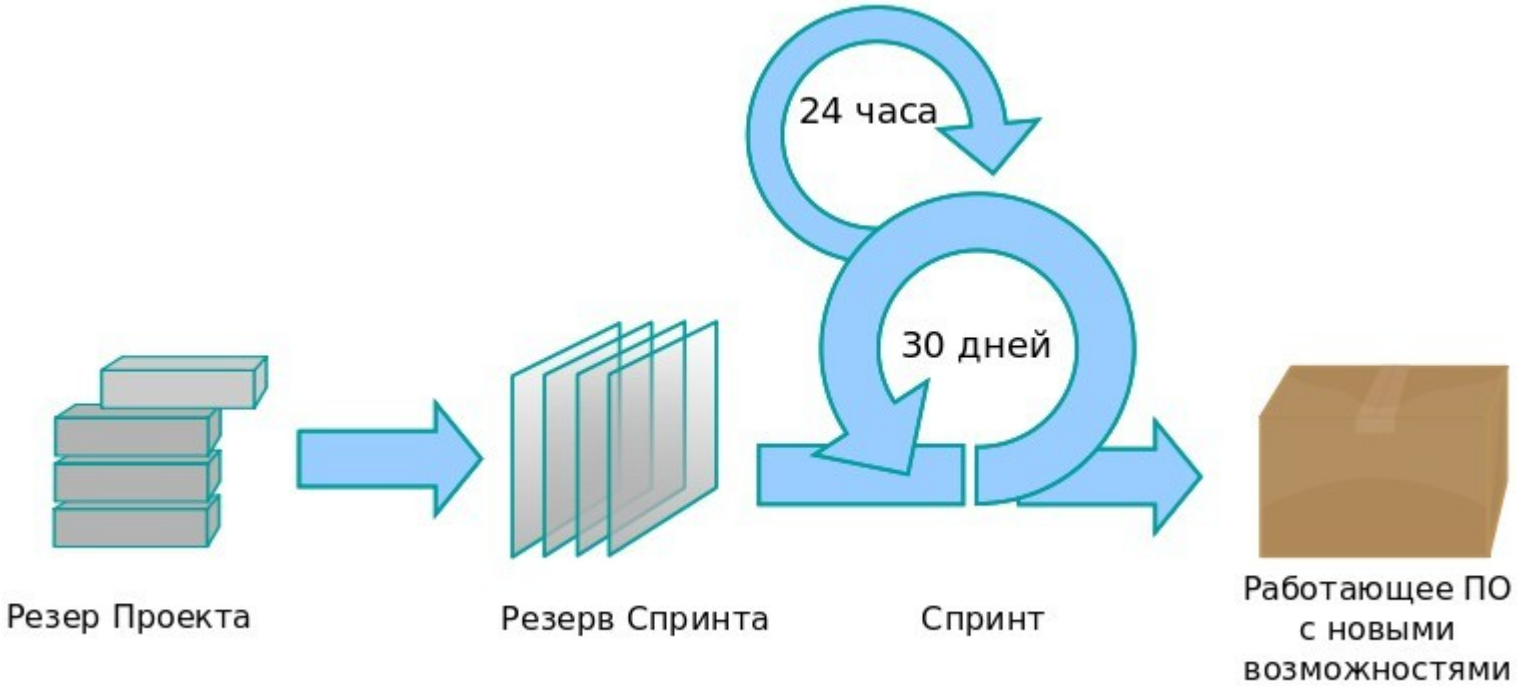
В феврале 2001 в штате Юта США был выпущен «Манифест гибкой методологии разработки программного обеспечения». Данный манифест был одобрен и подписан представителями методологий экстремального программирования, Crystal Clear, DSDM, Feature driven development, Scrum, Adaptive software development, Pragmatic Programming.



Scrum

Scrum — методология управления разработкой информационных систем для гибкой разработки программного обеспечения. Scrum чётко делает акцент на качественном контроле процесса разработки. Кроме управления проектами по разработке ПО Scrum может также использоваться в работе команд поддержки программного обеспечения (software support teams), или как подход управления разработкой и сопровождением программ: Scrum of Scrums.

Scrum


24 часа
30 дней
Резер Проекта
Резерв Спринта
Спринт
Работающее ПО
с новыми
возможностями

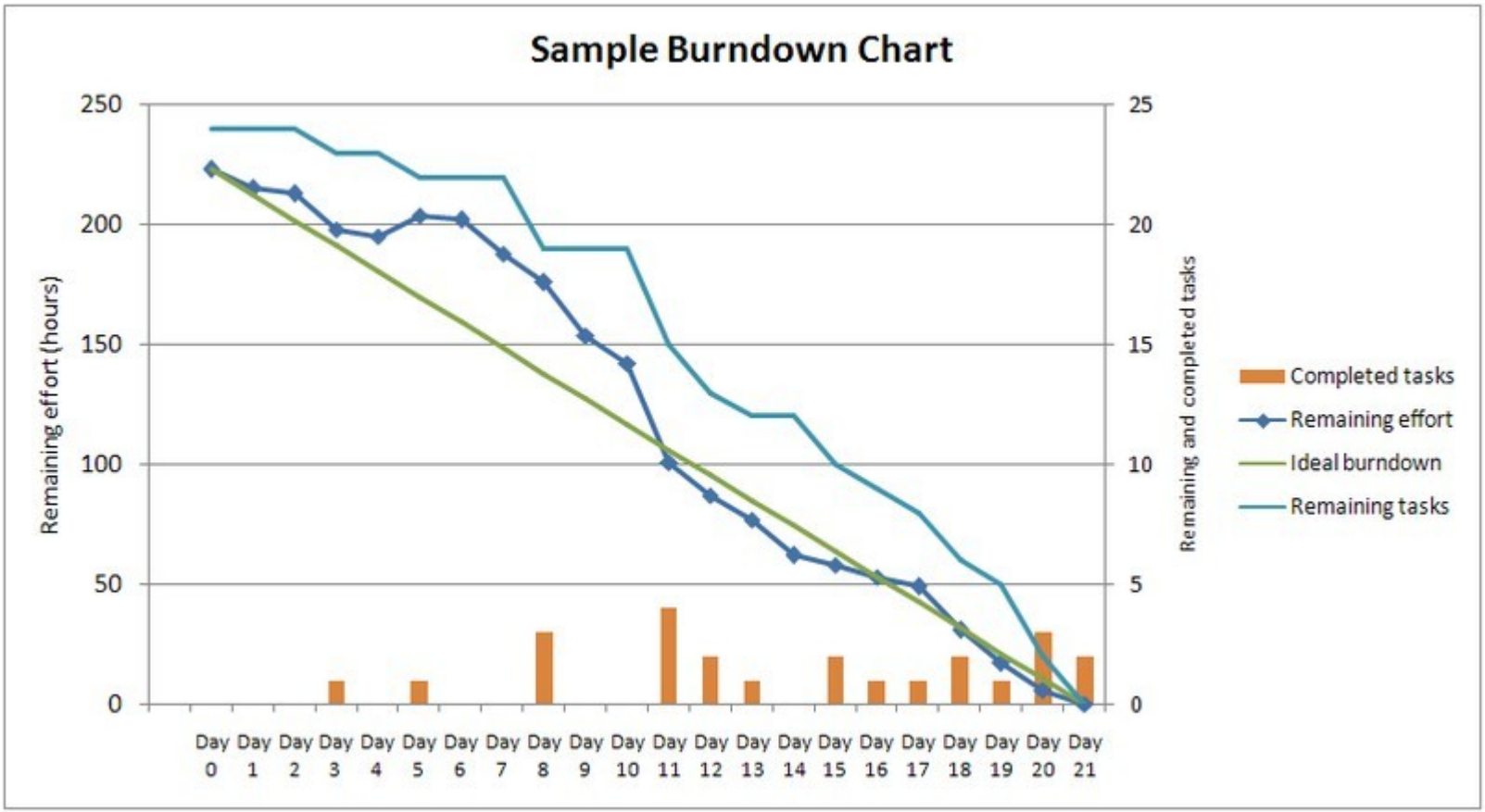
Scrum

Спринт — итерация в скрам, в ходе которой создаётся функциональный рост программного обеспечения. Жёстко фиксирован по времени. Длительность одного спринта от 2 до 4 недель.

Scrum

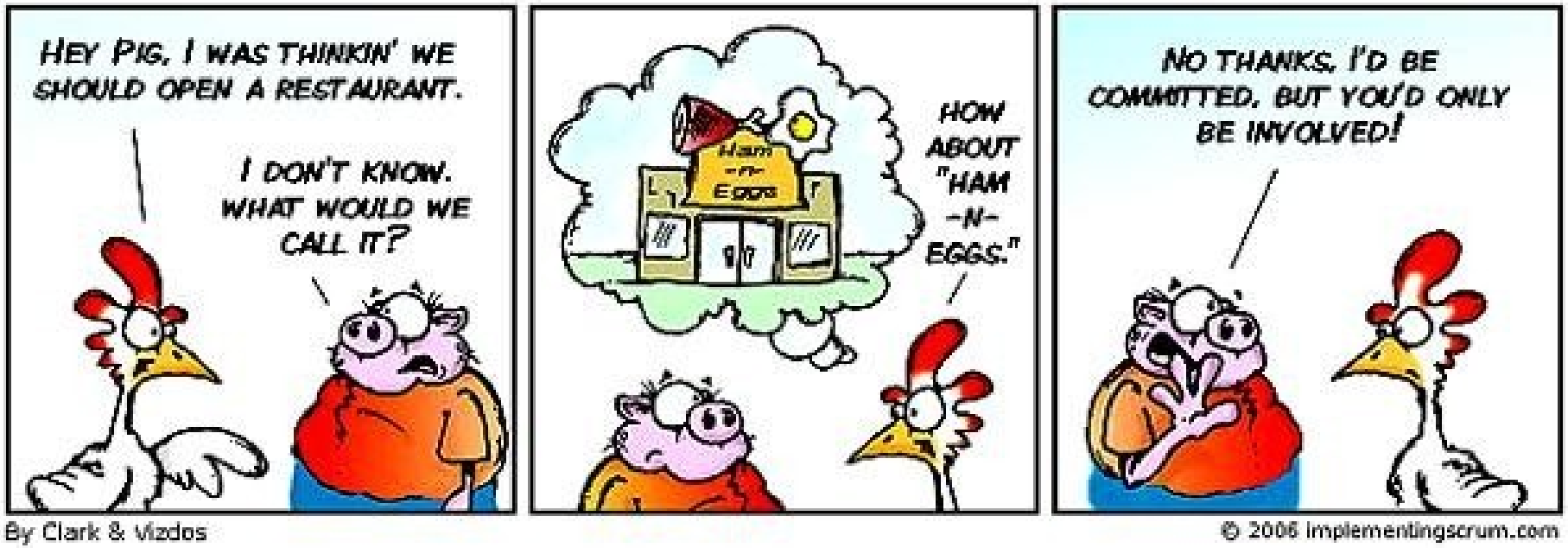
Резерв проекта — это список требований к функциональности, упорядоченный по их степени важности, подлежащих реализации. Элементы этого списка называются «пожеланиями пользователя» (user story) или элементами резерва (backlog items). Резерв проекта открыт для редактирования для всех участников скрам процесса.

Scrum


Sample Burndown Chart
Remaining effort (hours)
250
200
150
100
50
0
25
20
15
10
5
0
Remaining and completed tasks
Day 0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Day 8
Day 9
Day 10
Day 11
Day 12
Day 13
Day 14
Day 15
Day 16
Day 17
Day 18
Day 19
Day 20
Day 21
Completed tasks
Remaining effort
Ideal burndown
Remaining tasks

Scrum (роли)

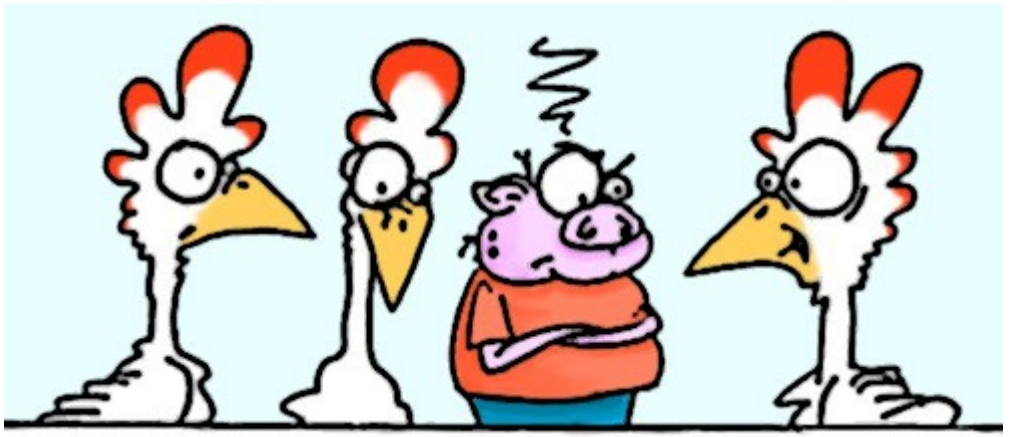
По методике Scrum в производственном процессе есть определенные роли, разбитые на 2 группы «свиней» и «кур».



Scrum (роли)

Свиньи:

- Скрам-мастер (ScrumMaster)
- Владелец проекта (Product Owner)
- Скрам-команда (Scrum Team)

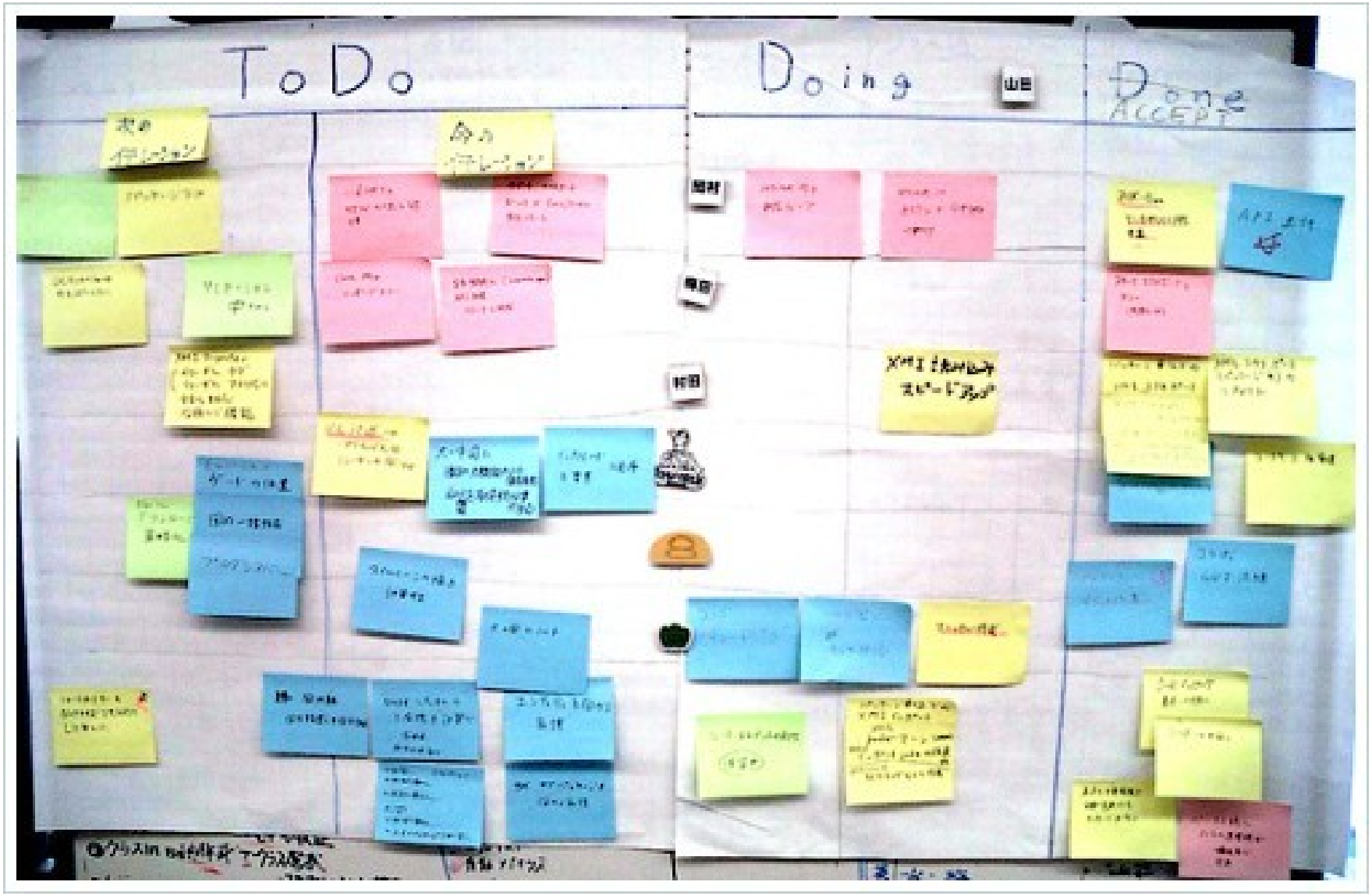


Scrum (роли)

Куры:

- Пользователи (Users)
- Клиенты, Продавцы (Stakeholders)
- Управляющие (Managers)
- Эксперты-консультанты (Consulting Experts)

Scrum (встречи)



MSF

Microsoft Solutions Framework.

Эта модель сочетает в себе свойства двух стандартных производственных моделей: каскадной (waterfall) и спиральной (spiral). Процесс MSF ориентирован на “вехи” (milestones) – ключевые точки проекта, характеризующие достижение в его рамках какого-либо существенного (промежуточного либо конечного) результата.

Базовые принципы

- Единое видение проекта. Успех коллективной работы над проектом немыслим без наличия у членов проектной группы и заказчика единого видения (shared vision), т.е. четкого, и, самое главное, одинакового, понимания целей и задач проекта. Формирование единого видения и последующее следование ему являются столь важными, что модель процессов MSF выделяет для этой цели специальную фазу (фаза “Выработка концепции”), которая заканчивается соответствующей вехой.
- Проявляйте гибкость – будьте готовы к переменам. Каскадная модель исходят из того, что все требования могут быть четко сформулированы в начале работы над проектом, и далее они не будут существенно изменяться. MSF основывается на принципе непрерывной изменяемости условий проекта при неизменной эффективности управленческой деятельности.

Базовые принципы

- Концентрируйтесь на бизнес-приоритетах. Продукт должен удовлетворить определенные нужды потребителей и принести в некоторой форме выгоду или отдачу. Это может означать, например, эмоциональное удовлетворение – как в случае компьютерных игр. Что же касается организаций, то неизменным целевым фактором продукта является бизнес-отдача (business value).Обычно продукт не может приносить отдачу до того, как он полностью внедрен. Поэтому модель процессов MSF включает в свой жизненный цикл не только разработку продукта, но и его внедрение.
- Поощряйте свободное общение. Модель процессов MSF предполагает открытый и честный обмен информацией как внутри команды, так и с ключевыми заинтересованными лицами. Модель процессов MSF предлагает проведение анализа хода работы над проектом в определенных временных точках. Документирование результатов делает ясным прогресс, достигнутый в работе над проектом - как для проектной команды, так и для заказчика и других заинтересованных в проекте сторон.

Фазы и вехи


Внедрение завершено
Deployment Complete
Готовность решения утверждена
Release Readiness Approved
Концепция проекта утверждена
Vision/Scope Approved
Разработка завершена
Scope Complete
Планы проекта утверждены
Project Plans Approved
MSF
Envisioning
Planning
Developing
Stabilizing
Deploying

Фаза выработки концепции

Основными задачами фазы выработки концепции являются создание ядра проектной группы и подготовка документа общего описания и рамок проекта (vision/scope document). Формирование видения проекта и специфицирование его рамок – не одно и тоже, хотя для успеха проекта необходимо и то, и другое. *Видение (vision)* – это ничем не ограничиваемое представление о том, каким должно быть решение. *Рамки (scope)* же дают четкие границы того, что из предложенного этим видением будет реализовано в условиях существующих проектных ограничений.

Веха “Концепция утверждена”

Это главная веха фазы выработки концепции. К моменту ее достижения проектная группа и заказчик должны прийти к соглашению об общих задачах проекта, включаемой и не включаемой в решение функциональности и временных рамках.

Результаты

- Результатами фазы выработки концепции являются:
- Общее описание и рамки проекта (vision/scope document).
- Документ оценки рисков (risk assessment document).
- Описание структуры проекта (project structure document).

Рекомендуемые промежуточные вехи

Ядро проектной группы сформировано

Черновой вариант концепции проекта составлен

Фаза планирования

В начале фазы планирования проектная группа анализирует и документирует проектные требования. Они разделяются на четыре общих категории: бизнес-требования (business requirements), потребительские требования (user requirements), эксплуатационные требования (operational requirements) и системные требования, относящиеся к решению в целом (system requirements). В ходе проектирования решения и создания его функциональной спецификации необходимо следить за *соответствием (traceability)* между имеющимися требованиями и проектируемой функциональностью. Это соответствие не обязательно будет взаимооднозначным. Оно служит одним из способов контроля корректности дизайна и его пригодности для достижения поставленных перед решением целей.

Фаза планирования

Существует три уровня процесса проектирования: концептуальный дизайн (conceptual design), логический дизайн (logical design) и физический дизайн (physical design). Работа над логическим дизайном начинается через некоторое время после начала концептуального дизайна, и работа над физическим дизайном стартует через некоторое время после начала работы над логическим.

Результаты процесса проектирования документируются в *функциональной(-ых) спецификации(-ях) (functional specification(s)).* Функциональные спецификации детально описывают вид и поведение каждой составляющей решения. Также для всех составляющих описывается их архитектура и дизайн.

Веха “Планы проекта утверждены”

Утвержденные спецификации, планы и календарные графики образуют базовую версию проекта (project baseline). Она включает все соглашения, принятые на основе консенсуса с учетом трех плановых параметров проекта: ресурсов, времени и функциональности решения. После того, как базовая версия проекта создана и утверждена, проектная группа приступает к фазе разработки.

Результаты

- Результатами фазы планирования являются:
- Функциональная спецификация.
- План управления рисками.
- Сводный план и сводный календарный график проекта.

Рекомендуемые промежуточные вехи

Базовая ве **создана**

Базова **цана**

Базовая ве **проекта**

Средь **ты**

Фаза разработки

На фазе разработки проектная группа фокусируется на создании компонент решения (включая как документацию, так и программный код). Однако некоторая часть этой работы может продолжаться также на фазе стабилизации, если такая необходимость выявлена в процессе тестирования. Данная фаза также включает в себя разработку инфраструктуры.

Следует обратить внимание, что активность проектной команды на этом этапе не ограничивается написанием разработчиками кода – все ролевые кластеры принимают деятельное участие в создании и тестировании решения.

Веха “Разработка завершена”

Эта веха является кульминацией фазы разработки. К моменту ее наступления создание всех составляющих завершено, и решение готово к тестированию и стабилизации. Заказчики, потребители, персонал сопровождения и другие заинтересованные стороны получают возможность оценить решение и выявить все оставшиеся проблемы и неурегулированные вопросы, которые должны быть улажены до выпуска решения.

Результаты

- Результатами фазы разработки являются:
- Исходный и исполнимый код приложений.
- Скрипты установки и конфигурирования.
- Окончательная функциональная спецификация.
- Материалы поддержки решения.
- Спецификации и сценарии тестов.

Рекомендуемые промежуточные вехи

Концепция подтверждена

Билд n завершен, билд n+1 завершен...

Фаза стабилизации

Во время фазы стабилизации производится тестирование разработанного решения. При этом внимание фокусируется на его эксплуатации в реалистичной модели производственной среды. Проектная группа занимается приоритезацией и устранением ошибок, а также подготовкой решения к выпуску.

Обычно в начале фазы стабилизации скорость выявления ошибок командой тестирования превосходит скорость, с которой эти ошибки могут устраняться командой разработчиков. Невозможно предсказать, сколько ошибок будет найдено и как много времени понадобится на их устранение. Однако существует два статистических признака, помогающих проектной группе оценить уровень стабилизации решения. Это точка конвергенции (bug convergence) и точка достижения нуля ошибок (zero bug bounce).

Веха “Готовность решения утверждена”

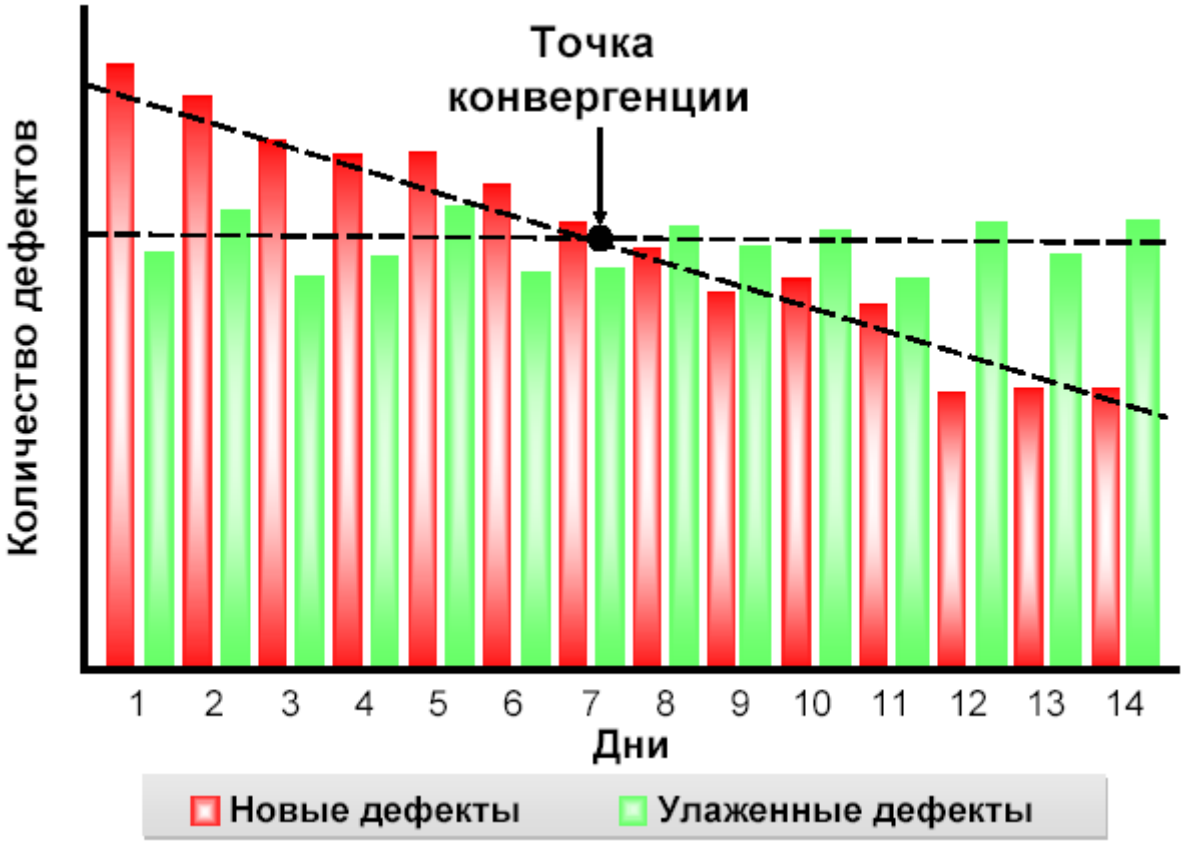
К моменту наступления этой вехи проектная группа завершает разрешение всех существенных проблем и происходит выпуск или внедрение решения. Ответственность за непрерывное управление и поддержку решения формально переходит от проектной группы к командам сопровождения.

Результаты

- Результатами фазы стабилизации являются:
- Окончательный продукт (golden release).
- Документация выпуска (release notes).
- Материалы поддержки решения.
- Результаты и инструментарий тестирования.
- Исходный и исполнимый код приложений.
- Проектная документация.
- Анализ пройденной фазы (milestone review).

Рекомендуемые промежуточные вехи

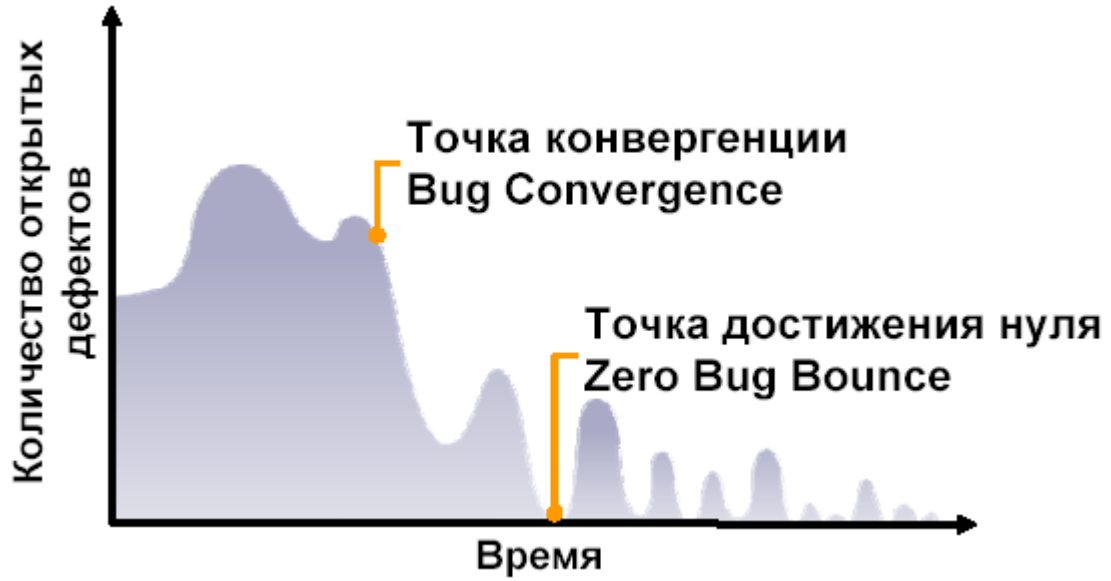
Точка конвергенции
В точке конвергенции (bug convergence) становится заметен существенный прогресс в устранении ошибок, то есть скорость устранения ошибок начинает превосходить скорость их обнаружения.



Рекомендуемые промежуточные вехи

Точка достижения нуля

Точка достижения нуля (zero-bug bounce) – это момент, когда впервые все выявленные ошибки оказываются устраненными.Вслед за ней пики количества активных ошибок должны становиться все меньше, вплоть до полного угасания в момент, когда решение уже достаточно стабильно для выпуска первой версии-кандидата.



Рекомендуемые промежуточные вехи

Версии-кандидаты

Контрольное тестирование завершено

Тестирование приемлемости для потребителей завершено

Пилотное внедрение завершено

Фаза внедрения

Во время этой фазы проектная группа внедряет технологии и компоненты решения, стабилизирует внедренное решение, передает работу персоналу поддержки и сопровождения и получает со стороны заказчика окончательное одобрение результатов проекта. По завершению внедрения проектная группа производит анализ выполненной работы и удовлетворенности заказчика.
Во время этой фазы по ходу переноса компонент решения из среды тестирования в производственную среду могут продолжаться меры по стабилизации решения.

Веха “Внедрение завершено”

Данная веха – кульминация фазы внедрения. К этому времени решение должно начать давать заказчику ожидаемую бизнес-отдачу, а проектная группа – свернуть свою деятельность. Проектная группа должна получить от заказчика подтверждение того, что его цели достигнуты. Решение должно быть стабильным и четко удовлетворять выработанным критериям успешности. Стабильность решения - готовность систем его эксплуатации и сопровождения.

Результаты

- Результаты фазы внедрения включают в себя:
- Информационные системы эксплуатации и поддержки.
- Процедуры и процессы.
- Базы знаний, отчеты, журналы протоколов (logbooks).
- Версии проектных документов, массивы данных (load sets) и программный код, разработанные во время проекта.
- Отчет о завершении проекта (project close-out report).
- Окончательные версии всех проектных документов.
- Показатели удовлетворенности заказчика и потребителей.
- Описание последующих шагов.

Рекомендуемые промежуточные вехи

Ключевые компоненты развернуты

Внедрение на местах завершено

Внедренное решение стабилизировано

Рекомендуемые методики

Стимулируйте изобретательность расширяя функциональность и ограничивая ресурсы

Фиксируйте календарный график

Календарное планирование на неопределенное будущее

Используйте параллельно работающие компактные команды

Разбивайте большие проекты на осуществимые части

Рекомендуемые методики

Извлекайте уроки из пройденных вех

Используйте прототипирование

Используйте частые билды и быстрые тесты

Частые итерации разработки и внедрения

Избегайте расползания рамок проекта

Оценка снизу вверх

Интегрирование представленных проектной группой оценок