



Управление производственным процессом разработки программного обеспечения

Введение

ФИТ Лекция №1. 25'

Лектор

Дмитрий Александрович Талочкин

Выпускник ФИТ НГУ 2005
MBA OU

20+ лет разрабатываю программное обеспечение.

Управляю:

20 – 40 постоянно действующих проектных команд
250+ человек

КМС по дзюдо, катаюсь на всем,
что движется: сноуборд, лыжи, кайт, виндсурефр, мото,
бегаю полумарафоны.



Семинаристы



Дмитрий Александрович
Талочкин

@dtalochkin гр. 22201



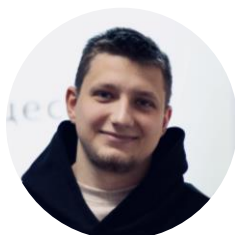
Владислав Евгеньевич
Кукарин

@Koshekans гр. 22202, 22204



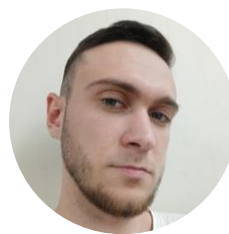
Андрей Викторович
Макаров

@MakarofAV гр. 22203



Андрей Андреевич
Корнещук

@andrewKorneshchuk гр. 22206



Никита Вадимович
Яковлев

@kisslyyi гр. 22208



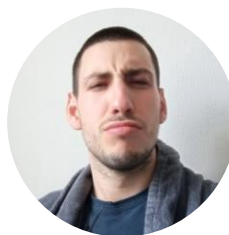
Данила Валерьевич
Иванченко

@danillo19 гр. 22209



Владислав Игоревич
Симонов

@Brobchik гр. 22207



Сергей Васильевич
Мясоедов

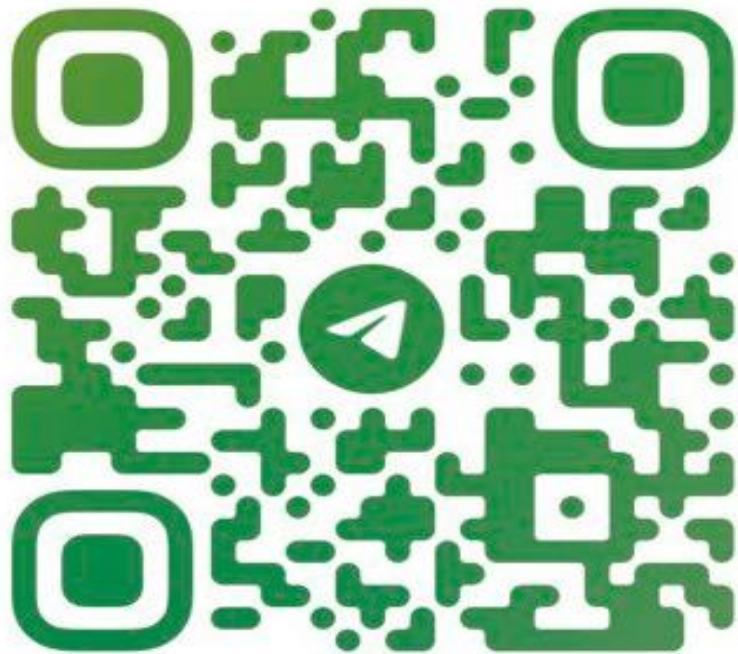
@sergomyaso гр. 22205



Новое в курсе 2025

1. Безопасная разработка, сертификация, требования регуляторов
2. Soft skills и Hard skills
3. Trunk based development, Release Based Development
4. Эмоциональный интеллект
5. Цифровые экосистемы от CJM до UI Kit
6. Сервис ориентированные архитектуры и микросервисы
7. Shift-left практики в тестировании
8. Аналитика, системная аналитика, метрики
9. и многое другое.

Материалы и контроль



@NSU_SDP_2025

1. Дифференцированный зачет
2. Оценка по практическим занятиям (учебный проект)
3. Учитывается посещение лекций
4. В спорных ситуациях экзаменует и ставит оценку лектор

О чем этот курс?

Организация
проекта
и методология

Информационная
безопасность

Тестирование
и верификация

Управление
рисками

Конфигурационное
управление

Планирование
и оценка

Управление
знаниями



Чему мы должны научиться?

Выбирать

Учитывая внешнее окружение и свойства продукта.

Инструменты

Наиболее уместные и эффективные для данной стадии жизни продукта.



С учетом ограничений

Оказывающих существенное влияние на продукт и его производство.

Ограничения

Продукт

- ✓ Работа в режиме 24/7
- ✓ Необходимость ежедневных релизов, проверки гипотез
- ✓ Корпоративные продукты со сложным жизненным циклом
- ✓ Поддержка множества версий на различном оборудовании

Регуляторные

- ✓ Программное обеспечение для объекта критической инфраструктуры
- ✓ Платежные сервисы
- ✓ Продукт для открытого рынка с большой пользовательской базой
- ✓ Внутренний корпоративный продукт

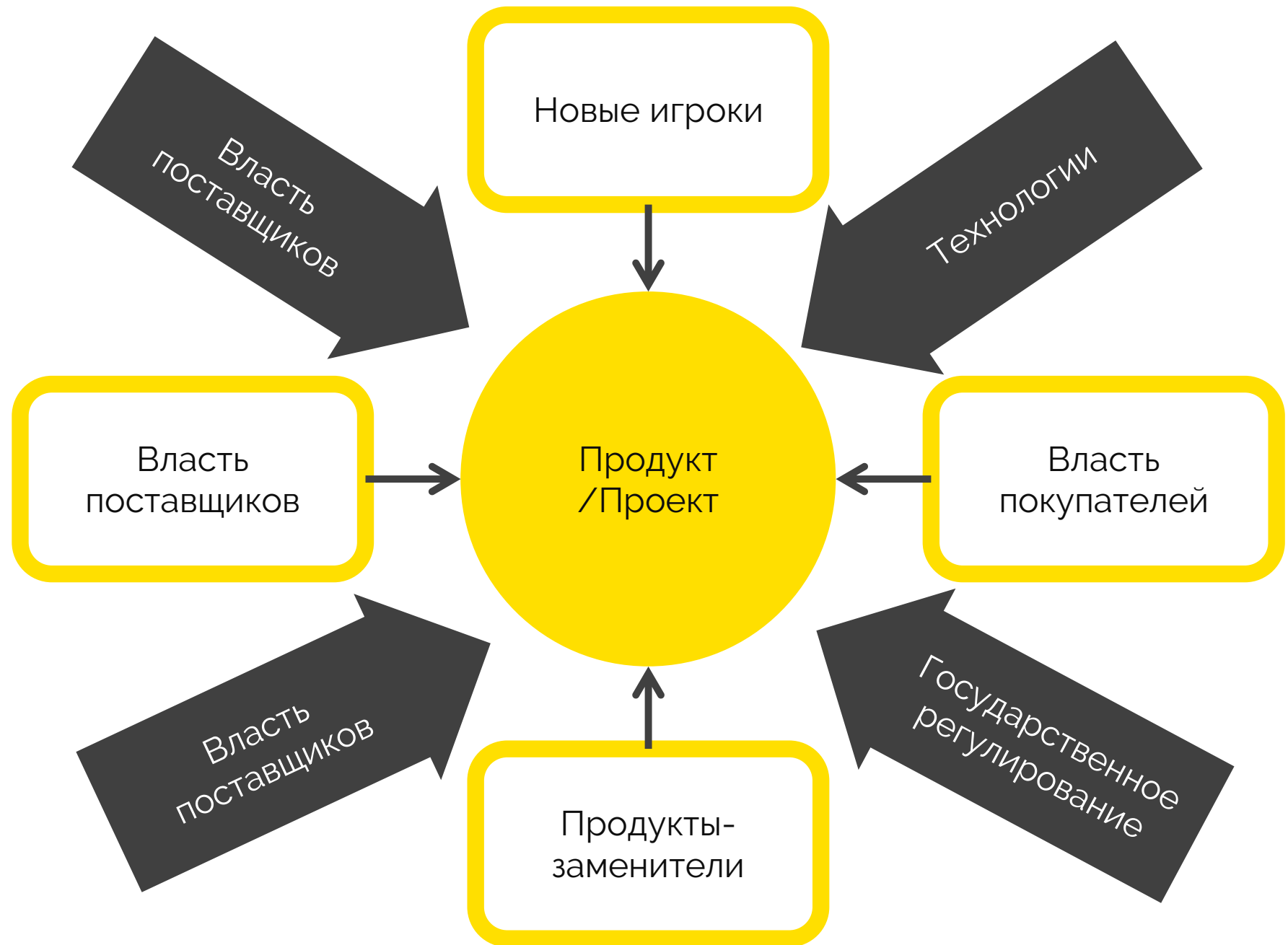
Команда

- ✓ квалифицированная команда senior разработчиков
- ✓ 20 – 30% команды senior разработчики, остальные среднего уровня
- ✓ Географически распределенная команда в различных часовых поясах

Продукты заменители

- ✓ Уникальные продукты
- ✓ Создаваемые на замену, например в рамках импорто-замещения
- ✓ Рыночные и не рыночные продукты

Модель Портера



Soft & Hard skills

Soft skills

1. Лидерство и команд образование
2. Управление временем, проблемами, стрессом
3. Способы организации проектной работы
4. Конфликтология
5. Групповая динамика и мотивация
6. Эффективная коммуникация и навыки публичных выступлений

Hard skills

1. Планирование и оценка
2. Управление интеграцией и содержанием
3. Управление расписанием
4. Управление стоимостью
5. Управление качеством
6. Управление поставками
7. Управление коммуникация
8. Управление конфигурациями

Проект

Имеет **четко определенное**
во времени **начало** и **конец**

Решает проблему или
удовлетворяет требования
конкретного **клиента** или **заказчика**

Конечный набор задач для
достижения определенной **цели**

Команда существует только во время
действия проекта

Ограничения

1. Требования
2. Цели заказчика
3. Условия функционирования
- 4. Бюджет**

Риски

1. Сделать не то,
2. Сделать не так,
3. Не уложиться в бюджет
4. Не уложиться в сроки

Недостатки

1. низкое вовлечение,
2. сложности с переводом в эксплуатацию

Продукт

Не имеет окончания, конец продукта это негативное событие

Решает проблему как минимум нескольких клиентов, а в идеале целого сегмента **рынка**

Постоянный процесс изменений продукта

Команда существует постоянно и является носителем экспертизы

Ограничения

1. Требования рынка,
2. Технологии,
3. UX,
4. **Окупаемость (монетизация и продвижения)**

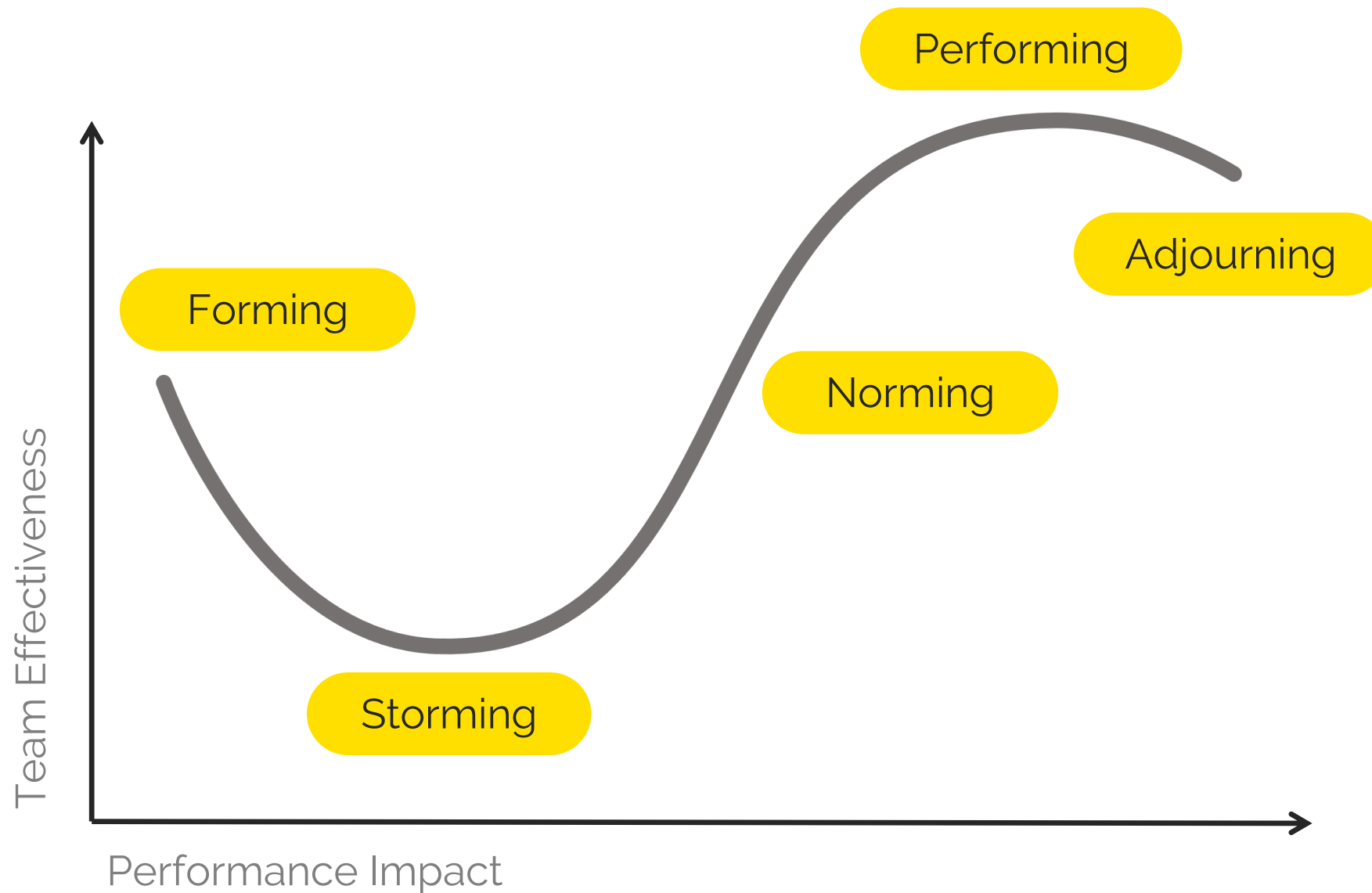
Риски

1. не обеспечить окупаемость,
2. проиграть конкурентную борьбу

Недостатки

1. ресурсов всегда не хватает,
2. необходимость выдерживать постоянную скорость

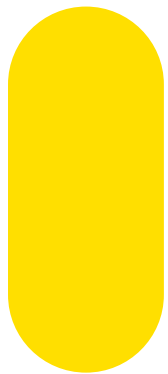
Модель Такмена Дженсена



1. Проект проходит все фазы формирования команды
2. Продуктовая команда работает в режиме максимальной производительности



Несколько слов о заинтересованности



Пользователь –

надежное программное обеспечение, максимально быстрое обновление

Разработчик –

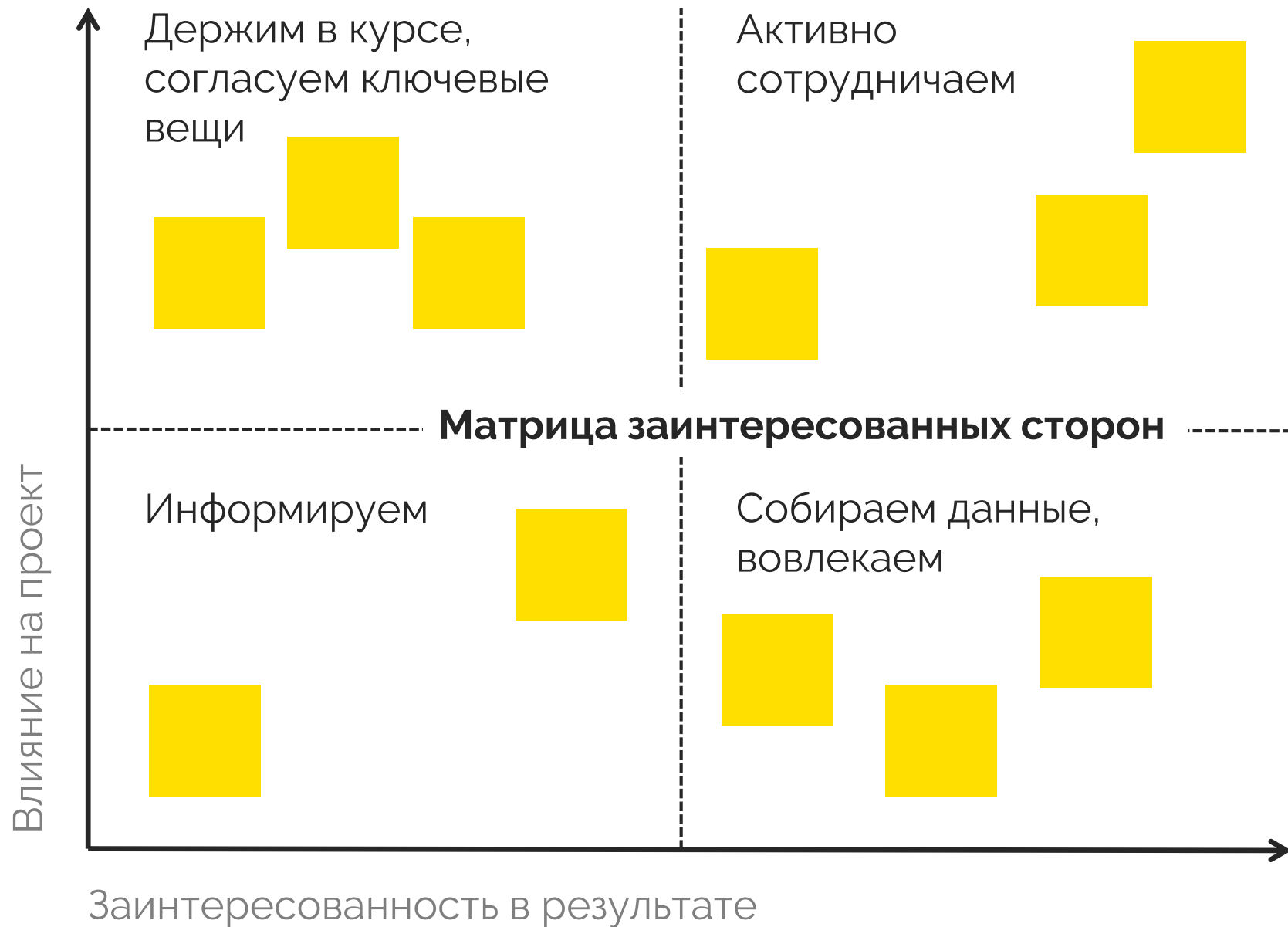
отсутствие бюрократии, прозрачный процесс разработки

Руководитель проекта

(владелец продукта) –

низкий TTM, предсказуемость и повторяемость, прозрачность

Модель Менделоу



- Определяем заинтересованные стороны
- Выявляем зоны потенциальных конфликтов интереса
- Выстраиваем стратегию коммуникации и управления

Модель Менделоу (комплексный взгляд)



Value Stream Mapping



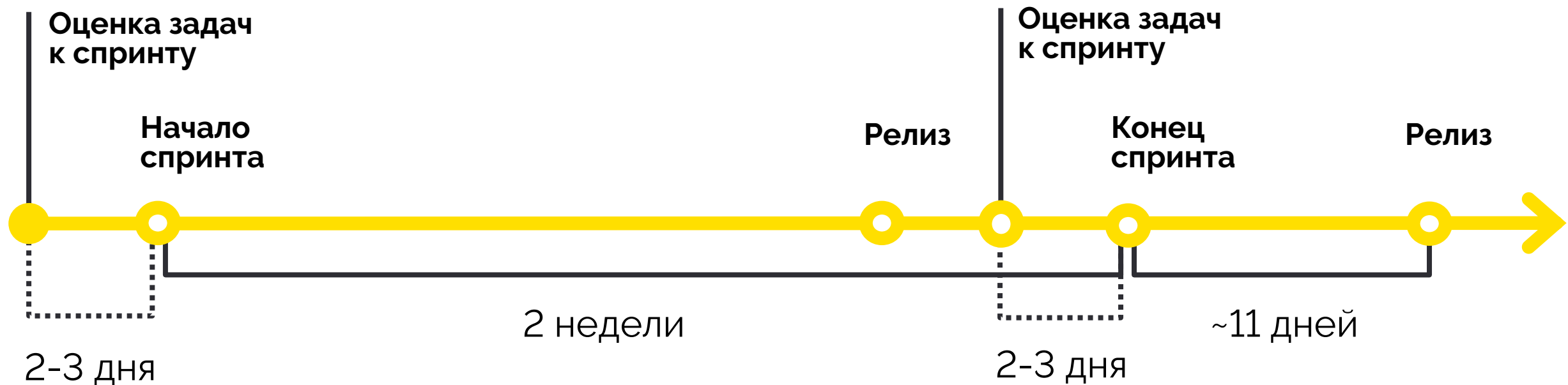
Value Stream Mapping –

последовательность шагов, осуществление, которых приводит к созданию ценности (продукта, услуги для пользователя)

Основные понятия

1. Информационный поток
2. Продуктовый поток
3. Временные интервалы
4. Время цикла
5. Время подготовки
6. Доступность
7. Такт

Пример: Типовой жизненный цикл задачи



От начала спринта:

1. Разработка – 3 дня
2. Ручное тестирование – 1 день
3. Ревью и исправления – 2 дня
4. Пауза на время дежурства – 1 день
5. Ожидание приемки на Stage – 3 дня
6. Ожидание остальных задач релиза – 11 дней



Например, к чему стремимся

Было	Станет
Релиз 1 раз в неделю	Релиз ежедневно
Ревью ветки > 1 дня	Ревью Task < 2 часа
Ручная выкладка хотфикса	Автоматическая выкладка хотфикса
Развертывание проекта на Prod ~ 2 часа	Развертывание проекта на Prod ~15 минут
Ручное регрессионное тестирование ~ 16 часов	Автоматическое регрессионное тестирование ~ 15 минут

Эволюция подходов к разработке

1970s

Structured programming since **1969**

Cap Gemini SDM, originally from PANDATA, the first English translation was published in **1974**.
SDM stands for System Development Methodology

1980s

Structured systems analysis and design method (SSADM) from **1980** onwards
Information Requirement Analysis/Soft systems methodology

1990s

Object-oriented programming (OOP) developed in the early **1960s** and became a dominant programming approach during the **mid-1990s**

Rapid application development (RAD), since **1991**

Dynamic systems development method (DSDM), since **1994**

Scrum, since **1995**

Team software process, since **1998**

Rational Unified Process (RUP), maintained by IBM since **1998**

Extreme programming, since **1999**

2000s

Agile Unified Process (AUP) maintained since **2005** by Scott Ambler

Disciplined agile delivery (DAD) Supersedes AUP

2010s

Scaled Agile Framework (SAFe)

Large-Scale Scrum (LeSS)

DevOps

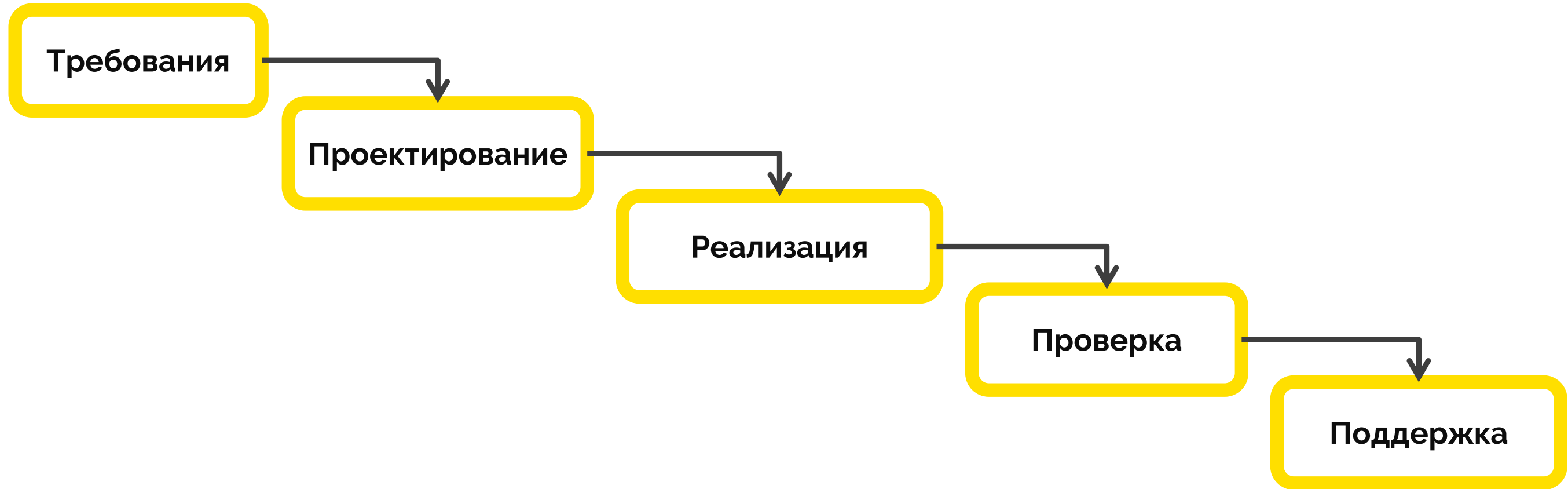


Методологии разработки

1. **Каскадная модель** (waterfall)
2. **Спиральная модель**
3. **Итеративная модель**, RUP (Rational Unified Process)
4. **Гибкая разработка** (Agile manifesto): Scrum, Kanban

Каскадная модель

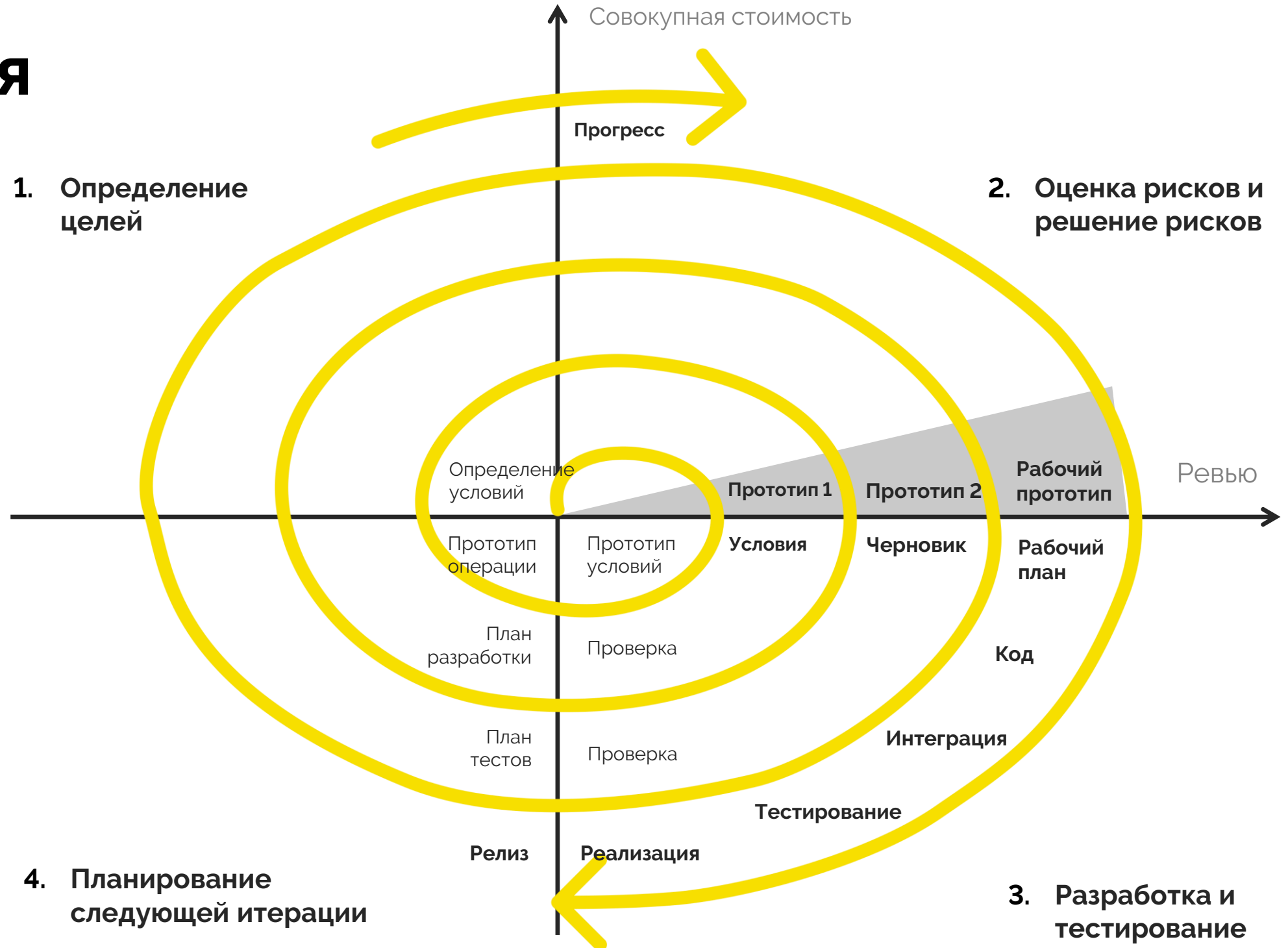
Модель водопада [W. W. Royce, 1970]



Проблемы водопадной модели

1. Многие вещи можно делать разными способами и **выбрать лучший способ можно только ретроспективно**
2. Не всю неопределенность можно разрешить в начале проекта
3. Часто неопределенность возникает из **внешних факторов**
4. В процессе работы могут возникать **непредвиденные изменения** в требованиях
5. Если на каком-то этапе возникла проблема, требуется **возвращаться на несколько шагов назад**
6. В результате **стоимость** проекта **всегда больше запланированной**

Спиральная модель

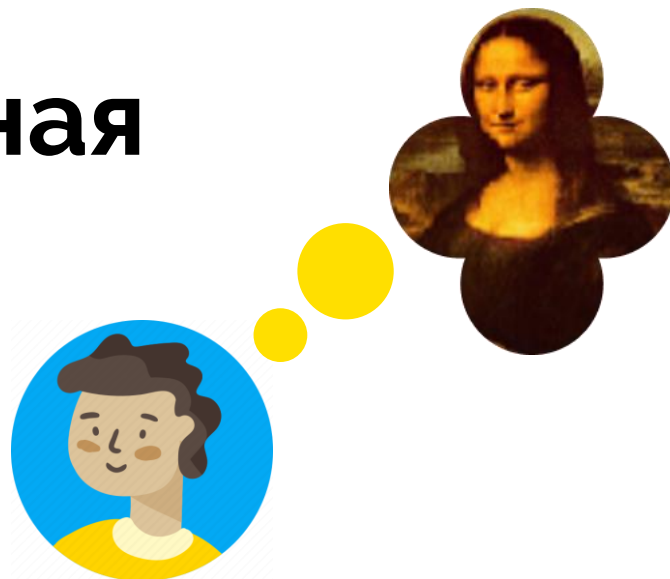




6 инвариантов спиральной модели

1. Определение всех артефактов одновременно
2. Определение и согласование со всеми стейкхолдерами критериев успеха в каждом цикле
3. Степень риска определяет уровень затрат для его разрешения
4. Степень риска определяет уровень детализации
5. Контрольные точки (майлстоуны)
6. Концентрация на системе и ее жизненном цикле

Инкрементная модель



1



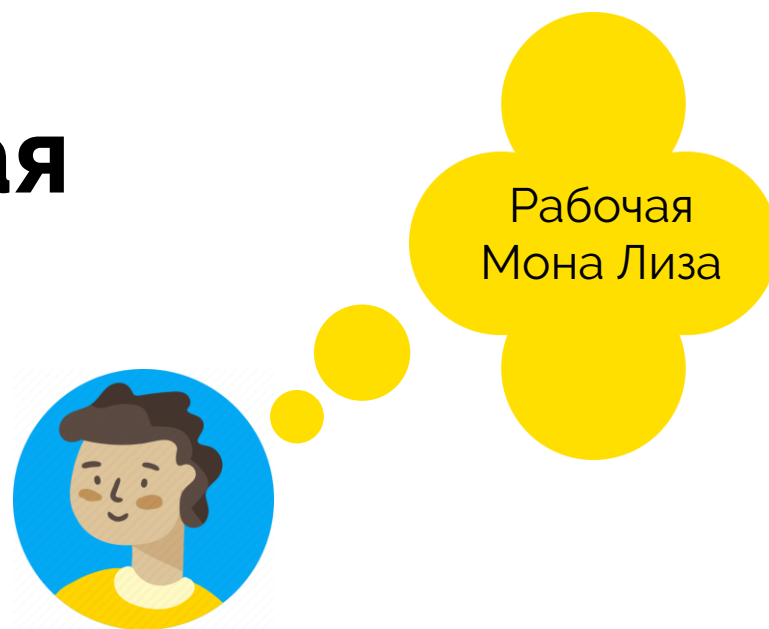
2



3



Итеративная модель



1



2



3





Риски “водопадных” моделей

1. Не удовлетворить функциональные требования
(сделать не то)
2. Не удовлетворить нефункциональные требования
(сделать не так)
3. Сделать неэффективно
(долго, дорого в эксплуатации, некачественно)

Agile манифест (2001)

Мы постоянно открываем для себя более совершенные методы разработки программного обеспечения, занимаясь разработкой непосредственно и помогая в этом другим.

Благодаря проделанной работе мы смогли осознать, что:

Люди и взаимодействие важнее процессов и инструментов

Работающий продукт важнее исчерпывающей документации

Сотрудничество с заказчиком важнее согласования условий контракта

Готовность к изменениям важнее следования первоначальному плану

То есть, не отрицая важности того, что справа, мы всё-таки ценим то, что слева.



12 принципов гибкой разработки ПО

1. Наивысший приоритет — удовлетворение заказчика посредством частых и регулярных поставок ценного для него ПО
2. Изменения в требованиях приветствуются даже на поздних этапах реализации проекта
3. Работающий продукт следует выпускать каждые несколько недель, в крайнем случае — каждые несколько месяцев
4. Наиболее эффективный и действенный способ передачи информации — это встреча членов команды разработки ПО
5. Представители бизнеса и команда разработки должны работать над проектом вместе
6. Проекты строятся вокруг мотивированных людей



12 принципов гибкой разработки ПО

7. Работоспособный результат — это главная мера прогресса проекта
8. Непрерывающаяся деятельность, держать постоянную скорость разработки
9. Постоянное внимание к техническому совершенству и качественной архитектуре способствует гибкости
10. Простота — это искусство не делать лишней работы
11. Лучшая архитектура, требования и дизайн создаются в самоорганизующихся командах
12. Команда постоянно ищет способы стать более эффективной путем настройки и коррекции своих действий

eXtreme Programming

Лучшие практики разработки доведенные до экстремального состояния и способные поддерживать друг друга.

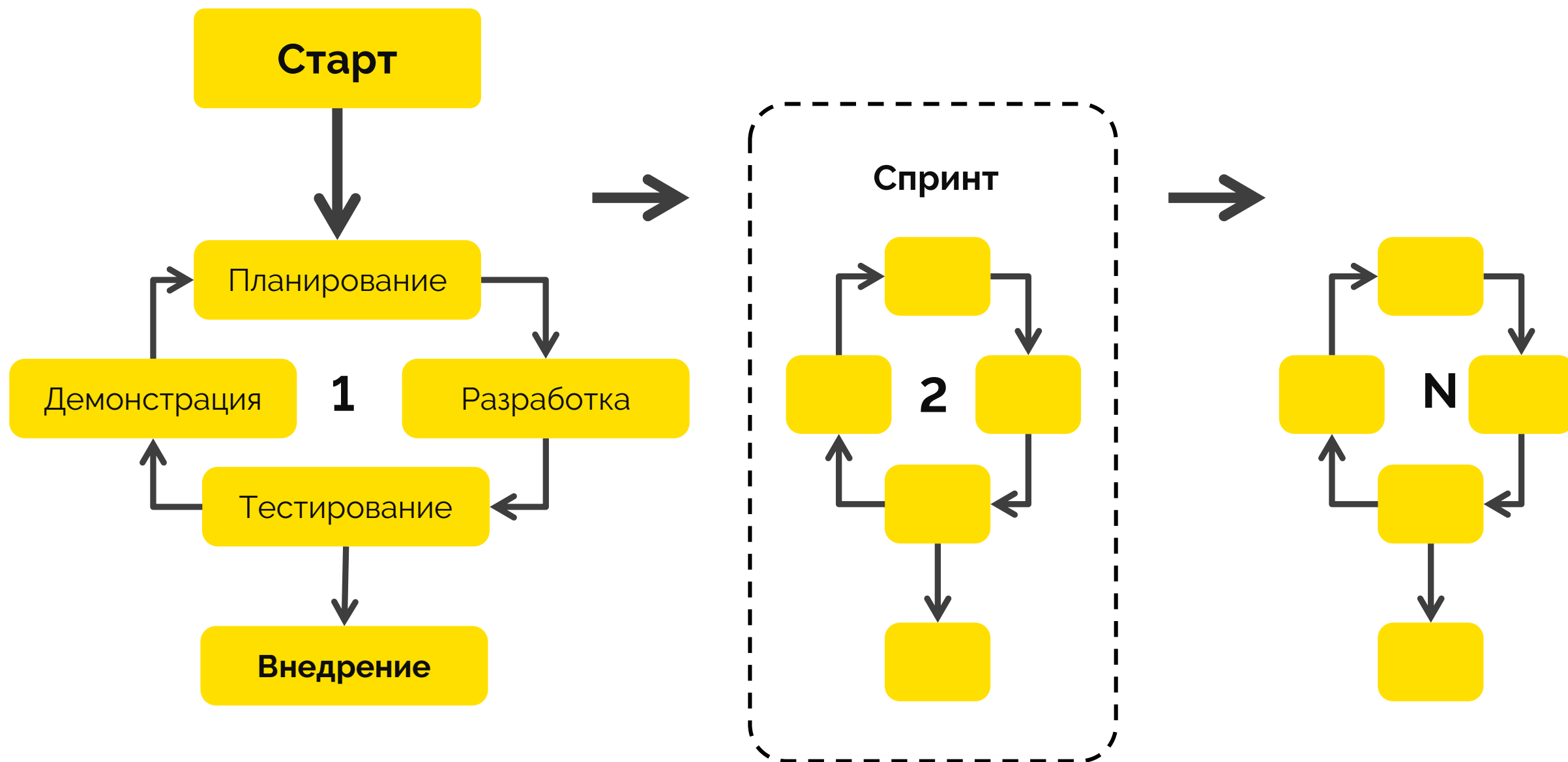
Кент Бек, конец 90-х



12 практик XP

1. Парное программирование
2. Игра в планирование
3. Test Driven Development
4. Заказчик как член команды
5. Непрерывная интеграция
6. Рефакторинг
7. Частые релизы
8. Стандарты кодирования
9. Коллективное владение кодом
10. Простая архитектура
11. Метафора системы (общий словарь)
12. Постоянная скорость

Гибкая разработка



Scrum: История

Кен Швабер, конец 90-х

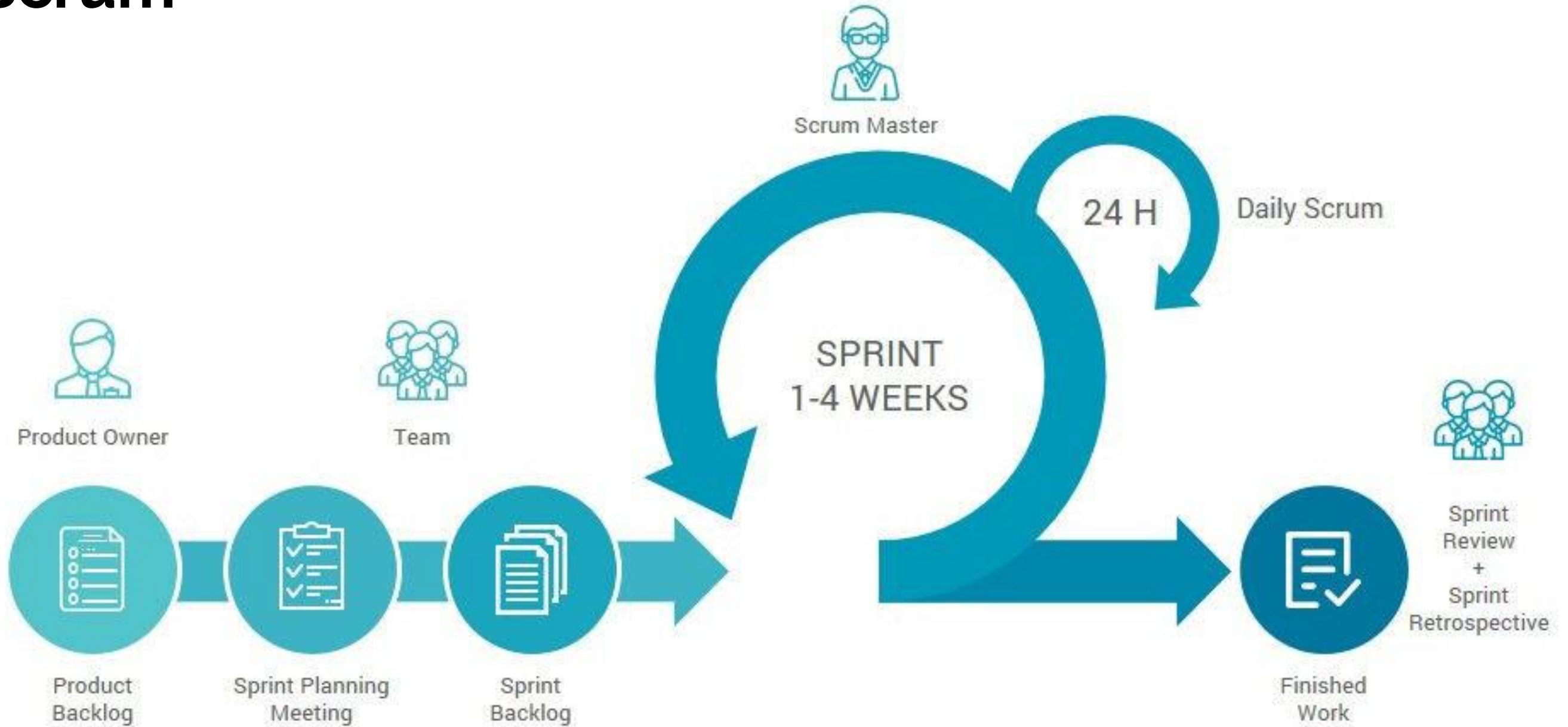
Название и терминология позаимствована из регби.

Спринт — ограниченная по времени итерация

Бэклог — набор задач которые должны быть сделаны

Бэклог спринта — задачи которые должны быть сделаны в текущем спринте

Scrum





Scrum. Спринт

1. Планирование в начале спринта
8 часов
 - а) Приоритезация бэклога с участием РО
 - б) Выбор задач командой
2. Ежедневная планерка
15 минут
3. Что делалось вчера
 - а) Что будет делаться сегодня
 - б) Что мешает
 - в) Демонстрация**2 часа**
4. Ретроспектива
3 часа
5. Груминг бэклога (по необходимости)



Недостатки Scrum

1. Плохо переносит увеличение команды
2. Фокус на краткосрочных целях в ущерб долгосрочному планированию
3. Плохо переносит глобальные задачи
4. Страдают нефункциональные активности (тестирование, документация)
5. Срочные задачи всегда вызывают конфликты

Kanban

(Toyota Production System)

1. Ограничение на количество задач в очереди
2. Оставшиеся без задач помогают отстающим
3. Гарантия исполнения стандартных задач за определенное время





Правила работы с канбан-доской


1. Не забывайте перемещать карточки на доске в соответствии с движением задачи.
2. Все задачи должны быть на доске и иметь приоритет по выполнению.
3. Используйте оптимальное количество статусов на доске.
4. У каждой команды должны быть своя доска.
5. Определите оптимальное количество задач в каждом статусе (если будет 100 карточек на доске, она потеряет свою наглядность и простоту).



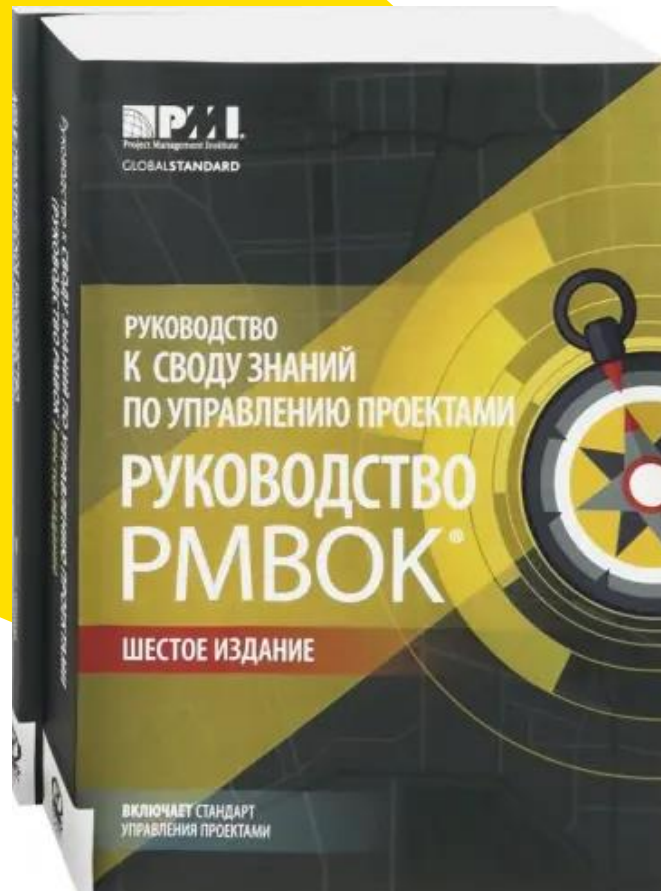
Канбан. Область применимости

1. Большое количество однотипных задач
2. Большая взаимозаменяемость
3. Отсутствие фиксированных сроков
4. Техническая поддержка, работа с клиентами, системное администрирование

Scrum vs Канбан

	<div>Scrum</div> <div></div>	<div>Kanban</div> <div></div>
Длительность	Регулярные спринты определенной продолжительности	Непрерывный процесс
Релизы	В конце каждого спринта	Постоянное обновление продукта
Метрики	Сколько работы команда сделает за спринт (velocity, story points)	Время исполнения, время разработки, WIP (work-in-progress лимит)
Роли	Product owner, scrum master, development team	Нет определенных ролей
Изменения	Изменения во время спринта нежелательны, перенос на новый спринт	Изменения в требованиях допустимы в любое время
Обязательства	Выполнение всей запланированной работы на спринт	Исполнение стандартных задач за определенное время (на основе статистики)

PMBoK



PMBoK — это совокупность процессов, практик и инструментов, необходимых для успешной реализации проектов, описывает основные знания и практики в области управления проектами.

Это универсальное руководство, которое содержит набор базовых концепций, терминов и методов, необходимых для эффективного управления проектами на любом уровне сложности и в различных областях деятельности



PMBoK

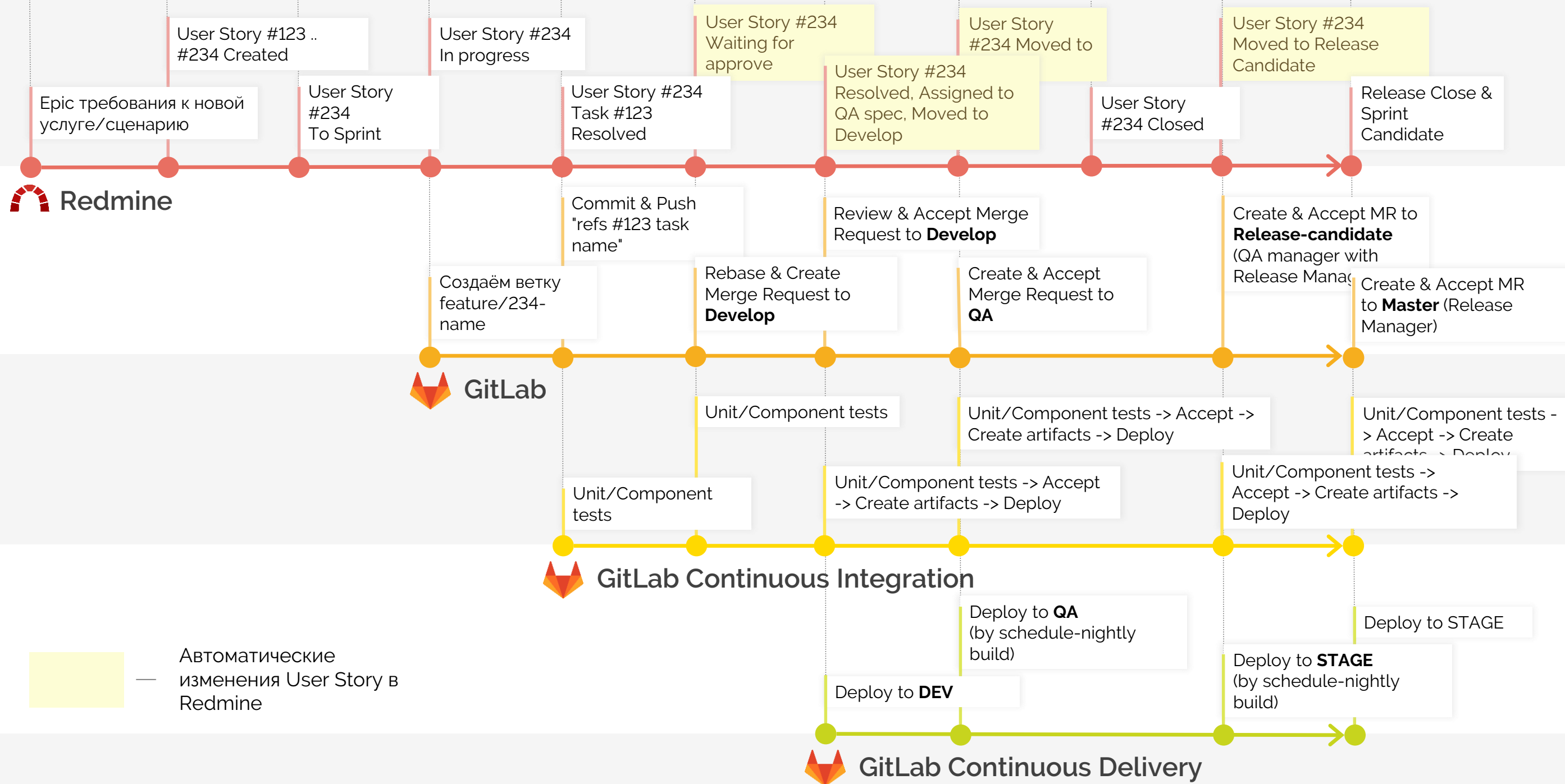
- 1. Процессы управления проектами.** Выделяют 49 процессов, которые распределены по 5 группам: инициация, планирование, выполнение, мониторинг и контроль, закрытие. Каждый процесс имеет цели, входы, выходы и инструменты.
- 2. Виды знаний.** Существует 10 областей знаний: управление содержанием проекта, временем, деньгами, качеством, заинтересованными сторонами, коммуникациями, ресурсами, поставками, рисками и интеграцией.
- 3. Процессы управления стейкхолдерами.** Они включают идентификацию, анализ и учет потребностей, ожиданий и влияния стейкхолдеров на проект.
- 4. Процессы управления коммуникациями.** Для успешного выполнения проекта необходимо обеспечить эффективный обмен информацией со всеми участниками проекта.
- 5. Процессы управления качеством.** К ним относятся: планирование, обеспечение и контроль качества продукта или услуги проекта с целью удовлетворения требований заинтересованных сторон.
- 6. Процессы управления рисками.** Они заключаются в определении потенциальных угроз для проекта и разработку стратегий по их минимизации или устранению.



Промышленный процесс

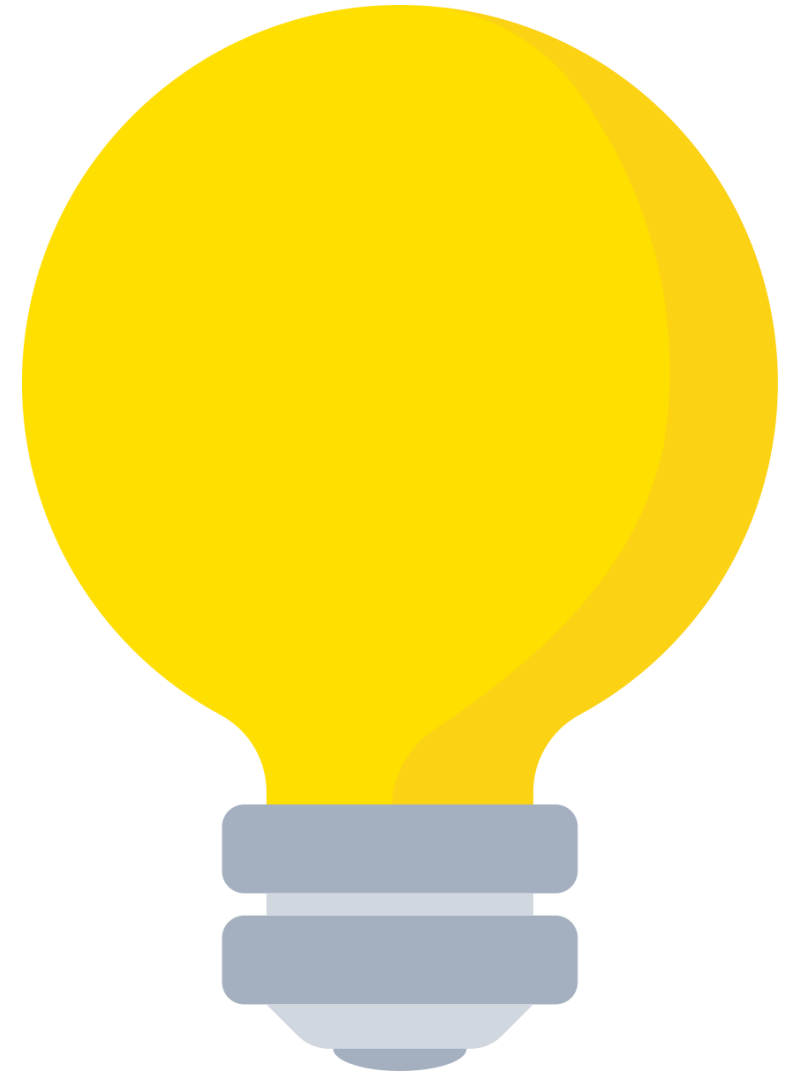
1. Разработка в команде
2. Минимизация непроизводственных издержек (на коммуникацию, на исправление ошибок, на разработку того, что в итоге окажется ненужным)
3. Воспроизводимость результатов
4. Предсказуемость сроков и результатов
5. Прозрачность на всех этапах

Пример внедрение промышленного процесса



Ключевые мысли

1. Управление производством требует как hard skills так и soft skills.
2. Свойства продукта лежат в основе проектирования процесса производства.
3. Внешние факторы оказывают существенное влияние инструменты управления и технологии производства.
4. Квалификация команды должна соответствовать выбранным инструментам производства.
5. Производственный процесс может меняться во времени в зависимости от задач.
6. Производственный процесс дает повторяемость результата.
7. Процесс это набор математически четких правил изменения состояния объектов управления.



Вопросы?





**Спасибо
за внимание**