МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

ПО ОРГАНИЗАЦИИ ПРОИЗВОДСТВЕННОГО ПРОЦЕССА РАЗРАБОТКИ ГОСУДАРСТВЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ С УЧЕТОМ ПРИМЕНЕНИЯ ИТЕРАЦИОННОГО ПОДХОДА К РАЗРАБОТКЕ

2022 год

ОГЛАВЛЕНИЕ

1. ОБЩИЕ ПОЛОЖЕНИЯ

1) Настоящий документ содержит методические рекомендации по организации производственного процесса создания и развития государственных информационных систем с учетом применения итерационного подхода к разработке (далее — Методические рекомендации, Системы) и с использованием современных методов и практик управления реализацией цифровых продуктов и сервисов.

2) Документ соответствует положениям законодательства Российской Федерации, регулирующим создание и развитие Систем.

3) Положения данного документа распространяются на деятельность государственных органов (далее — Ведомств), их подведомственных организаций, частных партнеров в соответствии с соглашениями о государственно-частном партнерстве, концессионеров в соответствии с концессионными соглашениями, организаций, уполномоченных нормативными правовыми актами на создание Систем, коммерческих организаций, выполняющих (или планирующих выполнять) работы по созданию и развитию Систем в рамках Государственных контрактов, и организаций иных организационно-правовых форм.

4) Цели документа:

   а. сформировать для всех Ведомств и организаций, указанных в *п. 3)* настоящего документа, единое понимание методов и практик создания и развития Систем с использованием итерационного подхода;

   б. разработать предложения по использованию итерационного подхода в зависимости от масштаба создаваемых Систем (по количеству компонентов Систем, связей между ними и охвату пользовательской аудитории) и сложности организации работ по созданию и развитию Систем (количества привлекаемых специалистов).

5) Использование настоящих Методических рекомендаций направлено на обеспечение:

   а. создания Систем в интересах предоставления государственных услуг, государственных функций, включая контрольно-надзорную деятельность в условиях изменяющихся потребностей клиентов и условий (контекста) использования создаваемых Систем;

   б. сокращения сроков и повышение качества предоставления клиентам государственных сервисов с использованием создаваемых Систем;

   в. создания и развития Систем в рамках установленных бюджетов и с минимальными рисками недостижения целей создания Системы.

6) В ходе выполнения работ по созданию (развитию) Систем рекомендуется использовать взаимосвязанные методы и практики итерационной разработки, бережливого производства и DevSecOps[1], указанные в *Приложении № 1* к настоящему документу.

7) В ходе создания (развития) Систем требуется выполнять мероприятия, направленные на предотвращение появления и/или устранение уязвимостей разрабатываемого программного обеспечения, устанавливаемых законодательством Российской Федерации и действующими нормативными актами в области защиты обрабатываемой информации и создания Систем в защищенном исполнении, в том числе указанные в документе «Методические рекомендации по обеспечению разработки безопасного программного обеспечения на ЕЦП «ГосТех».

8) В настоящем документе использованы термины, их определения и сокращения, приведенные в *Приложении № 2* и (или) определенные действующим законодательством.

9) Методические рекомендации предполагают развитие и уточнение по результатам анализа их практического применения и изменений нормативных документов, определяющих порядок создания и развитие Систем.

[1] DevSecOps — акроним из слов Development, Security и Operations — набор методов и практик, основанных на автоматизации рабочих процессов создания и эксплуатации Системы, предполагающих непрерывное улучшение Системы путём непрерывной интеграции реализуемых компонентов, проверок безопасности и непрерывной доставки реализованной функциональности до клиента

2. ПОРЯДОК СОЗДАНИЯ И РАЗВИТИЯ СИСТЕМЫ

10) Порядок реализации мероприятий по созданию, развитию, вводу в эксплуатацию, эксплуатации и выводу из эксплуатации Систем определен постановлением Правительства Российской Федерации от 6 июля 2015 г. № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных систем и дальнейшего хранения содержащейся в их базах данных информации» (далее — Постановление № 676).

11) В настоящем документе представлены особенности производственного процесса итеративной разработки Систем, детализирующие ключевые положения Постановления № 676.

2.1. Структура требований к Системе

12) Система разрабатывается в соответствии с требованиями, определяемыми техническим заданием на создание Системы, порядок формирования, согласования и утверждения которого определяется Постановлением № 676. При этом в техническом задании на создание и развитие Системы (или ее части) может быть предусмотрено требование о применении итерационного подхода к разработке.

13) В случае применения итерационного подхода к разработке полный объем требований к Системе, устанавливаемых в техническом задании, должен быть сформирован в виде Групп требований, реализация каждой из которых должна обеспечивать одну или несколько возможностей Системы. При этом для Групп требований должны быть установлены конечные сроки их реализации и критерии приёмки.

14) Каждая Группа требований в общем случае включает *(Рисунок 1)*:

а. **функциональные требования**, реализация которых создает ценность для клиентов;

б. **нефункциональные требования**[2] — требования к видам обеспечения, в том числе требования к безопасности, надежности, производительности, удобству обслуживания и использования, масштабируемости. Нефункциональные требования обязательны к учету при проектировании Системы и могут рассматриваться как ограничения, например, требование к обеспечению функционирования Системы в определенных операционных системах.

[2] Нефункциональные требования классифицируются также как **атрибуты** системы и/или системного окружения (например, Леффингуэлл Дин, Уидриг Дон, Принципы работы с требованиями к программному обеспечению. Унифицированный подход, 2002).

в. наряду с функциональными требованиями дополнительно рассматривают **архитектурные (или технические) требования**, обеспечивающие реализацию заданной ценности для клиента, а также создание, улучшение и автоматизацию контуров разработки, тестирования и развертывания.

15) Приоритеты реализации Групп требований, указанных в техническом задании, могут определяться (уточняться) непосредственно при создании (развитии) Системы (в рамках специальных мероприятий по планированию работ) в порядке, устанавливаемым Ведомством, ответственным за создание и развитие Системы, например, с использованием Методики определения приоритетов, представленной в *Приложении №1*. Управление приоритетами обеспечивает выполнение в первую очередь наиболее важных работ, дающих максимальный эффект, и снижение рисков нереализации запланированной функциональности.

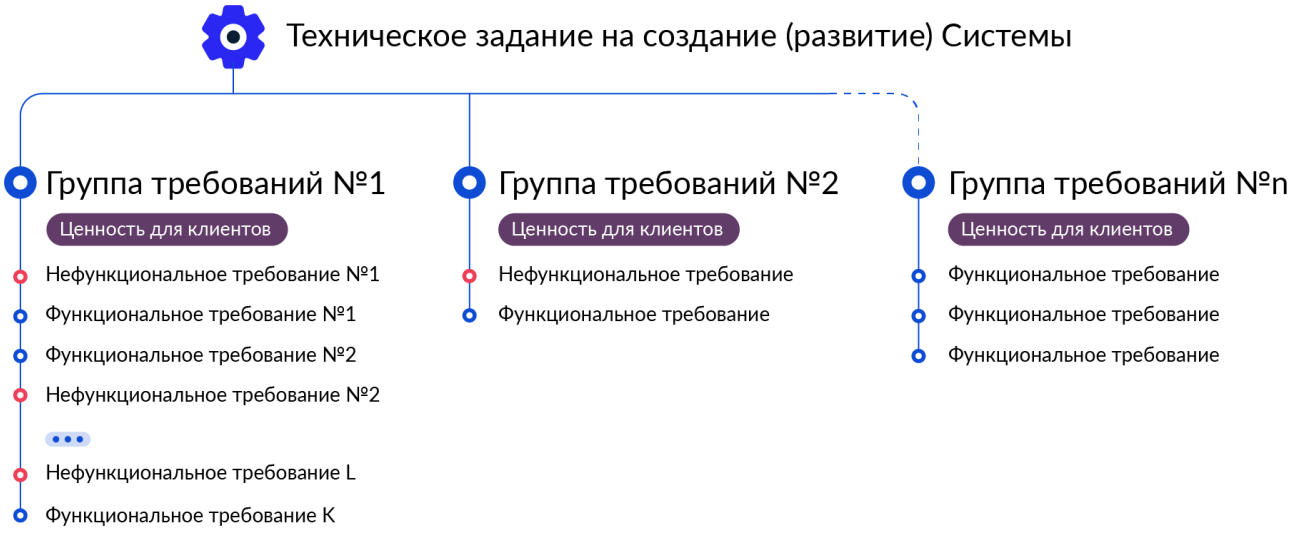

Рисунок 1. Структура требований к Системе

2.2. Этапы создания и развития Системы

16) Постановление № 676 определяет следующие этапы выполнения работ по созданию или развитию Системы:

а. разработка документации на Систему и ее части;

б. разработка рабочей документации на Систему и ее части;

в. разработка или адаптация программного обеспечения;

г. пусконаладочные работы;

д. проведение предварительных испытаний Системы;

е. проведение опытной эксплуатации Системы;

ж. проведение приемочных испытаний Системы.

При этом допускается создание Системы Очередями, а при создании каждой последующей Очереди системы могут проводиться мероприятия по развитию предыдущих Очередей системы

17) Методические рекомендации предлагают использовать итерационный подход к организации разработки Системы на каждой из Очередей, основанный на цикле Деминга-Шухарта *(Рисунок 2)* и включающий следующие повторяющиеся стадии:

а. **Планирование:** определение целей, планирование работ по их достижению, а также выделение и распределение ресурсов, необходимых для выполнения запланированных работ;

б. **Выполнение:** непосредственное выполнение запланированных работ;

в. **Оценка результатов:** сбор информации о результатах выполнения запланированных работ, например, ключевых показателей эффективности (КПЭ), оценка результатов, выявление и анализ отклонений от плановых значений показателей, установление причин отклонений;

г. **Корректировка:** принятие мер по устранению причин отклонений от запланированного результата, изменение целей (если все варианты по корректировке отклонений исчерпаны), изменение планов и распределения ресурсов.

Рисунок 2. Типовой цикл выполнения работ с использованием итерационного подхода

18) В зависимости от сложности создаваемой Системы предлагается использовать несколько уровней разработки, на каждом из которых работа организуется

итерационно. В общем случае предлагается использовать трехуровневую модель процесса создания и развития Системы *(Рисунок 3)*:

1. Первый уровень состоит из последовательно реализуемых **Очередей** создания и развития Системы, на каждом из которых осуществляется реализация Технического задания на Систему (или ее части) в целом.
2. Второй уровень состоит из последовательно реализуемых циклов - Инкрементов (фиксированной продолжительностью от 8 до 12 недель), в рамках которых осуществляется реализацией Групп требований.
3. Третий уровень состоит из последовательности **Итераций** (фиксированной продолжительностью от 1 до 4 недель), на которых осуществляется реализация функциональности Системы, определяемой отдельными требованиями из Группы требований. Короткие периоды времени, отводимые на Итерацию, помогают разработчикам, а также другим заинтересованным сторонам регулярно тестировать и оценивать технические и функциональные гипотезы, реализуемые в Системе. Каждая итерация включает точку интеграции, в рамках которой собираются различные аспекты системы — функциональность, качество, согласование архитектурных решений и пригодность для использования всеми командами.

Рисунок 3. Многоуровневая модель процесса создания и развития Системы

19) В рамках каждой Очереди могут быть созданы Система, часть Системы, а также могут выполняться мероприятия по развитию Системы.

20) Сроки реализации каждой Очереди Системы рекомендуется привязывать к бюджетному процессу в Российской Федерации.

21) Содержание мероприятий по реализации каждой очереди Системы определяется этапами выполнения работ, указанными в Постановлении № 676, а также иными дополнительными мероприятиями для обеспечения итерационного подхода к разработке *(Таблица 1)*.

Таблица 1. Стадии создания Системы (Очереди)

№	Стадия создания (развития) Системы	Мероприятия по созданию (развитию) Системы
1.	**Планирование**	1.1. Создание Рабочей группы 1.2. Разработка (уточнение) Концепции, технико-экономического обоснования, правовых актов, являющихся основанием для создания или развития Системы 1.3. Разработка технического задания на создание Системы (Очереди Системы) 1.4. Определение Исполнителя 1.5. Разработка документации на Систему и ее части
2.	**Выполнение работ**	2.1. Разработка и адаптация программного обеспечения 2.2. Разработка Рабочей документации на Систему 2.3. Пусконаладочные работы 2.4. Ввод в эксплуатацию частей (отдельных компонентов) Системы
3.	**Проверка выполненных работ**	3.1. Проведение испытаний Системы (Очереди) в целом 3.2. Ввод в эксплуатацию Системы (Очереди) в целом 3.3. Мониторинг функционирования Системы
4.	**Корректировка**	4.1. Определение требований к развитию Системы 4.2. Оценка и согласование бюджета развития Системы 4.3. Переход к стадии «Планирование» следующей очереди Системы

22) Организация работ с использованием итерационного подхода на уровне Итераций и Инкрементов описана в *Разделах 3–6* настоящего документа.

2.3. Процесс реализации Групп требований к Системе

23) Итерационный подход к разработке предполагает выполнение последовательности Инкрементов, каждый из которых включает в себя реализацию Групп требований к Системе в плановые сроки, в соответствии с предопределенными целями, критериями приемки каждого требования и критериями готовности к вводу в эксплуатацию. Это обеспечивает в рамках каждого Инкремента предсказуемый непрерывный процесс разработки новой ценности для клиентов (повышение качества результатов оказания Государственных услуг и выполнения государственных функций с использованием создаваемой и развиваемой Системы).

24) В рамках каждого Инкремента реализуются одна или несколько Групп требований к Системе *(Рисунок 4)*. В свою очередь Инкременты реализуются более короткими циклами разработки — Итерациями, в рамках которых реализуются отдельные функциональные и нефункциональные требования.

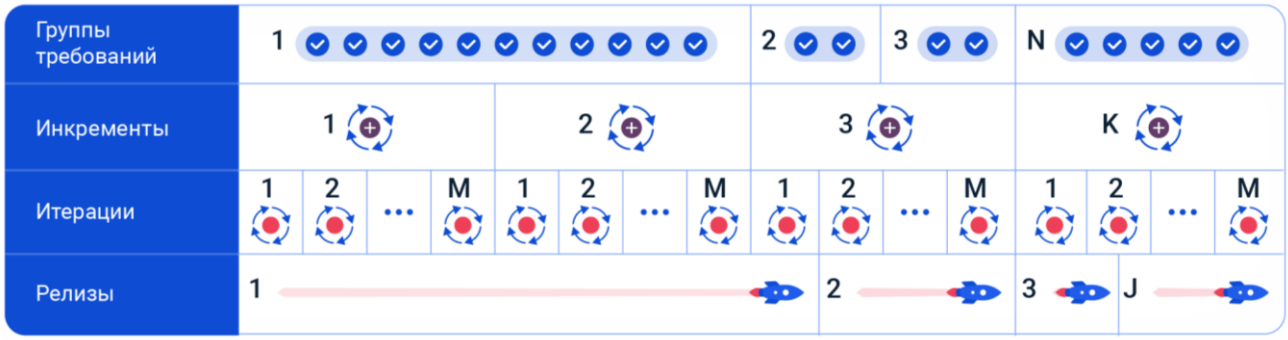

Рисунок 4. Реализация групп требований в рамках итераций

25) В рамках каждого Инкремента регулярно оцениваются результаты выполненных работ. Эта регулярная оценка обеспечивает управление рисками создания и развития Системы, необходимое для обеспечения того, чтобы бюджетные затраты на создание и развитие Системы приносили соразмерную отдачу в виде предоставляемой клиентам ценности. Регулярные оценки выполняемых работ проводятся в интересах обеспечения целевого эффекта оказания государственных услуг и выполнение государственных функций с использованием Системы.

26) После реализации Группы требований и после выполнения мероприятий по проверке выполненных работ, указанных в *Разделе 5*, может быть подготовлен и по решению уполномоченных должностных лиц Ведомства введен в эксплуатацию релиз Системы, что предполагает предоставление клиентам части реализованной функциональности. Такой подход обеспечивает:

1. оперативное предоставление ключевых, наиболее важных возможностей Системы клиентам в рамках первого релиза;

2. сокращение сроков подготовки последующих релизов к вводу в эксплуатацию при условии, что не потребуется повторение мероприятий по подготовке и вводу Системы в эксплуатацию, выполненных при вводе в эксплуатацию первого релиза. Например, если осуществляется модернизация Системы, не приводящая к повышению класса ее защищенности и (или) к изменению архитектуры системы защиты информации в части изменения видов и типов программных, программно-технических средств и средств защиты информации, изменения структуры системы защиты информации, состава и мест расположения Системы и ее компонентов — действие ранее выданного аттестата соответствия после модернизации (установки обновлений) не прекращается, и эксплуатация Системы может быть продолжена, а в дальнейшем — проведена повторная ее аттестация (ФСТЭК, Приказ от 29.04.2021 № 77 «Об утверждении порядка организации и проведения работ по аттестации объектов информатизации на соответствие требованиям о защите информации ограниченного доступа, не составляющей государственную тайну», *п. 34)*.

2.4. Организация команд разработки

27) Основной организационной единицей, обеспечивающей создание ценности для клиентов, является **Команда разработки**. Каждая команда разработки представляет собой самоорганизующуюся кросс-функциональную группу специалистов, обладающих необходимыми компетенциями для решения возложенных на них задач.

28) Так как сложность взаимодействия в таких группах увеличивается по мере увеличения числа специалистов в группе, рекомендуется формирование небольших команд (например, обычно лучше иметь две команды по пять человек, чем одну команду из десяти). Рекомендуется в Команду разработки включать 5…11 человек, которые проектируют, реализуют, тестируют и обеспечивают предоставление пользователям заданной функциональности.

29) Формирование Команд разработки должно осуществляться с учетом специфики и требуемых компетенций для решения задач. Команды разработки предпочтительно создавать на долгосрочную перспективу, состав команд должен быть стабилен. Также желательно обеспечить максимальную загрузку членов команды. Рекомендуется формировать Команды разработки таким образом, чтобы прогнозная оценка времени использования специалистов с требуемыми компетенциями превышала 70%. В противном случае специалистов необходимо включать в команды, обеспечивающие деятельность нескольких команд.

30) Поскольку сложность и объем задач по созданию Системы может потребовать более 11 человек, необходимо использование нескольких Команд разработки. Кроме того,

создание и развитие Системы требует широких и разнообразных компетенций и навыков, поэтому рекомендуется использование различных типов Команд разработки. Условно может быть использована следующая типизация:

а. команда ценности — основная команда, предназначенная для создания потока ценности и имеющая возможность доставлять ценность непосредственно клиенту (конечному пользователю);

б. команда сложных подсистем — специализированная команда, организованная вокруг конкретных подсистем, требующих глубоких специальных навыков и опыта;

в. команда платформы — специализированная команда, организованная вокруг разработки и поддержки цифровых платформ, на базе которых строится Система, и предоставляющая услуги другим командам;

г. команда поддержки — специализированная команда для оказания помощи (поддержки) другим командам со специализированными возможностями и новыми технологиями.

31) **Команды ценности** отвечают за создание и доставку требуемых функциональных возможностей Системы для клиентов (конечных пользователей) как можно быстрее, безопаснее и не требуя передачи другим командам для выполнения возложенных на них задач. Командам, ориентированным на создание ценности потребуется набор компетенций для реализации требований к Системе. Эти компетенции включают (но не ограничиваются):

а. создание проектных и архитектурных решений;

б. разработка (адаптация) программного обеспечения Системы, включая серверные компоненты и пользовательские интерфейсы;

в. мониторинг и обеспечение устойчивой работоспособности;

г. тестирование и обеспечение качества;

д. обеспечение безопасности приложений.

32) **Команда сложных подсистем** отвечает за создание и поддержку частей Системы, которые в значительной степени зависят от специальных знаний. Команда сложных подсистем берёт на себя ответственность за создание и обслуживание тех частей системы, которые требуют глубоких и постоянных специализированных технических знаний, включая узкоспециализированные компоненты Системы, критически важные для безопасности элементы Системы, которые имеют высокую стоимость отказа, специальные алгоритмы.

Примеры сложных подсистем могут включать, например, математические модели, алгоритмы согласования сделок в реальном времени, подсистемы отчетности о транзакциях для финансовых услуг, алгоритмы распознавания лиц.

В общем случае, в зависимости от особенностей создаваемой Системы может быть сформировано несколько команд сложных подсистем.

33) **Команды платформ** создаются для поддержки цифровых платформ[3], используемых в ходе создания Системы, и предоставления консультационных услуг командам ценности. Предоставляемые услуги, должны быть хорошо документированы, просты в использовании, соответствовать цели и предлагать возможности повторного использования.

Использование цифровых платформ призвано существенно сократить время реализации функциональных возможностей Системы, а в случае необходимости, обеспечивать быстрое прототипирование создаваемой функциональности.

34) **Команды поддержки** создаются для развёртывания и поддержки технологий разработки, а также оказания помощи командам в приобретении этих новых знаний и навыков в освоении новых технологий, обеспечивающих основные рабочие процессы.

Команды поддержки предоставляют экспертные знания и обеспечивают поддержку технологий:

а. механизмов DevOps, включая механизмы непрерывной интеграция и сборки;

б. автоматизированного тестирования;

в. непрерывной интеграции;

г. автоматической сборки релизов;

д. инженерных практик качества;

е. средств безопасности при разработке;

ж. развёртывания и поддержки контура разработки, тестового контура и контура эксплуатации.

При этом команды поддержки не предназначены для устранения проблем с качеством реализации функциональности Системы, создаваемой командами ценности. Команды поддержки могут быть постоянными или временными, а также обеспечивать поддержку различных команд.

---

[3] Цифровая платформа — это набор готовых сервисов, к которым могут получить доступ разработчики, например, через набор прикладных программных интерфейсов (API).

35) С учетом описанных типов команд и их особенностей для создания Системы могут быть организованы Группы команд разработки, отвечающие особенностям создаваемой Системы *(Рисунок 5)*.

Рисунок 5. Обобщённая структура Групп команд разработки

36) Эффективные Группы команд обычно состоят из 50-150 человек. Верхний предел основан на числе Данбара, которое предполагает ограничение на количество людей, с которыми можно сформировать эффективные, стабильные рабочие отношения. Нижний предел основан на эмпирических наблюдениях.

37) Учитывая указанные ограничения по размеру, существует два основных шаблона организации разработки: меньшие потоки создания ценности могут быть реализованы с помощью одной Группы команд, а большие потоки создания ценности должны обеспечиваться несколькими Группами команд разработки *(Рисунок 6)*. В последнем случае, при превышении количества специалистов, участвующих в создании (развитии) крупномасштабных Систем, 150 человек, необходимо создание нескольких Групп разработки и для обеспечения координации работ их предлагается объединять в Пулы команд разработки.

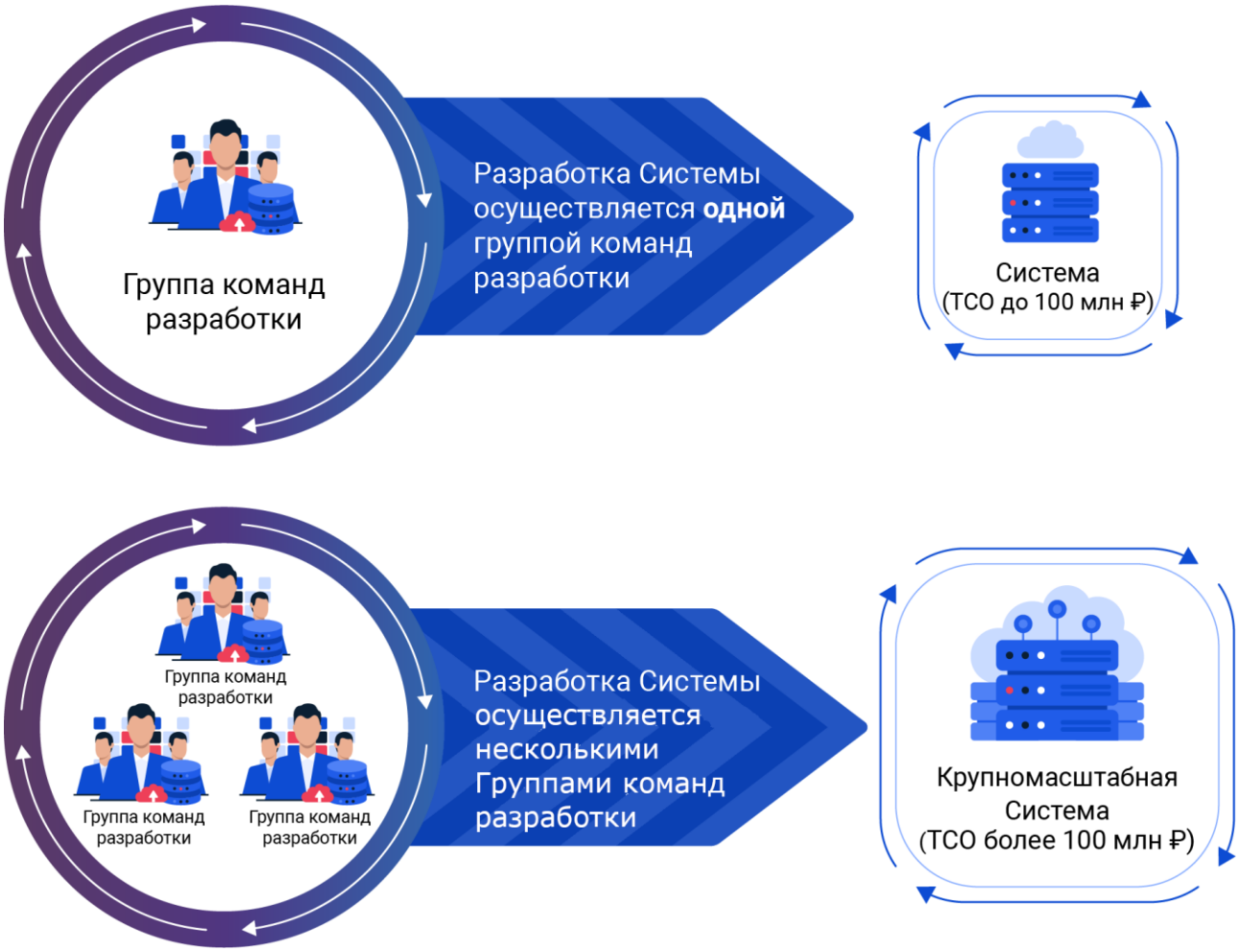

Рисунок 6. Варианты организации Групп команд разработки

38) Для организации Пулов команд *(Рисунок 7)* может быть использован подход определения типов Групп команд разработки аналогично представленному выше для Команд разработки.

Группы команд разработки, ориентированные на создание ценности (группы команд создания ценности), как и команды ценности, должны включать необходимых специалистов, навыков и компетенций которых достаточно для решения возложенных на них задач. Сферы ответственности этих групп разработки, как правило, такие же, как и для команд ценности.

Группы команд сложных подсистем могут быть использованы при построении крупномасштабных Систем для снижения нагрузки на Группы команд разработки ценности. Например, подсистема наведения для автономного транспортного средства вполне может потребовать привлечения Группы команд разработки сложной подсистемы.

Группы команд разработки, ориентированные на создание и поддержку платформы (группы команд платформы), предоставляют услуги Группам команд, ориентированным на создание ценности. Например, система связи, которая управляет данными, передаваемыми между различными подсистемами автономного транспортного средства, может быть представлена как платформа с четко определенными интерфейсами. Одним из дополнительных преимуществ Группы команд платформы является то, что она также поддерживает единые цифровые платформы для всех Групп команд разработки.

Рисунок 7. Обобщённая структура Пула команд разработки

39) Для управления работами по созданию и развитию Системы предлагается сочетание децентрализованного и централизованного подхода. Децентрализованной подход предполагает использование для выполнения работ кросс-функциональных самоорганизующихся команд, которые в рамках установленных полномочий могут принимать самостоятельные решения, в том числе в части порядка (последовательности) выполнения работ по реализации функций Системы в рамках отдельных итераций, при условии выполнения принятых обязательств по достижению целей, режима и способов работ над задачами, правил коммуникации между специалистами команды. Это сокращает временные и ресурсные издержки выполнения работ и позволяет использовать инновационные решения, предлагаемые разработчиками.

При этом сохраняется необходимость централизованного принятия критичных решений (архитектурных, финансовых, контрактных и иных) должностными лицами Ведомств, а также обеспечения координации команд.

40) Команды разработки должны работать стабильными циклами — итерациями. Использование единых по продолжительности циклов выполнения работ позволяет координировать Команды разработки в рамках Группы команд (так же, как и Группы

команд разработки в рамках Пула команд) и устанавливать единые временные рамки и единые процессы для них.

41) Стабильным циклом для Группы команд (или Пула команд) является Инкремент *(п. 24)*. Для согласования выполняемых работ по созданию и развитию Системы в целом между Командами разработки в ходе выполнения Инкремента должны проводиться регулярные мероприятия по координации работ, а также единые мероприятия по планированию предстоящих Инкрементов и подведению итогов выполнения работ *(Рисунок 8)*.

Рисунок 8. Координация Группы команд разработки

2.5. Особенности реализации первой очереди Системы

42) При планировании очередей создания и развития Системы целесообразно в качестве первой очереди рассмотреть:

1. Создание минимально достаточной версии Системы для подтверждения проектных решений и оперативной демонстрации ключевых возможностей Системы (представление ценности клиентам);
2. Проектирование Системы — в отдельных случаях создания сложных крупномасштабных Систем.

2.5.1. Разработка минимально достаточной версии Системы

43) С целью проверки правильности разработанных проектных решений и **в интересах снижения рисков, потерь времени и выхода за рамки бюджета** рекомендуется использовать подход «бережливого запуска» создания Системы, основанный на методах бережливого производства *(Приложение № 1)* и предполагающий создание первой очереди Системы, реализующей одну или несколько Групп требований в соответствии с техническим заданием и разработанными и согласованными (при необходимости) проектными (техническими) решениями в рамках расчётных сроков и бюджета.

44) Подход «бережливого запуска» предполагает создание минимально достаточной версии Системы (МДВ), разрабатываемой в рамках запланированного периода, состоящего из одного или нескольких Инкрементов, выполняемых в пределах установленных сроков и бюджета до тех пор, пока не будет реализована и проверена функциональность Системы, предоставляющая клиентам необходимую ценность, либо разработанные проектные решения продемонстрирует свою несостоятельность, например, в силу превышения запланированных сроков и/или бюджета или невостребованности результата клиентами.

45) В рамках «бережливого запуска» проводятся исследовательские работы (формируются гипотезы) по выбору технологий реализации Системы и выбор соответствующей инфраструктуры (программно-аппаратных средств) разработки. В том числе, проводится проверка возможности реализации предусмотренных в техническом задании требований (части требований) к программному обеспечению Системы посредством использования программ для электронных вычислительных машин, размещенных в Национальном фонде алгоритмов и программ, а в случае, если такие программы для электронных вычислительных машин выявлены в ходе указанной проверки, — установление наличия в техническом задании соответствующих требований по их использованию для создания Системы.

46) Если для создания Системы привлекаются внешние исполнители, рекомендуется заключение отдельного Государственного контракта или соглашения о государственно-частном партнёрстве, концессии или иных соглашений на создание МДВ.

47) Подход «бережливого запуска» упрощает проведение оценок сроков и бюджета создания последующих очередей Системы, в связи с тем, что функциональность и продолжительность МДВ существенно меньше, чем функциональность Системы в целом. Оценка полученных результатов в ходе создания МДВ также позволит уточнить прогноз затрат в рамках реализации последующих очередей Системы, в том

числе Совокупную стоимость владения Системой в случае ее реализации в архитектуре, подтвержденной созданием МДВ.

48) Создание МДВ обеспечит:

1. предоставление ключевых возможностей Системы клиентам за минимальные сроки;
2. сокращение сроков выпуска последующих релизов Системы за счет сокращения мероприятий по вводу Системы в эксплуатацию, установленных руководящими документами, после ввода в эксплуатацию первого релиза Системы (МДВ);
3. получение обратной связи от клиентов Системы, что может быть использовано для корректировки работ по созданию и развитию Системы.

2.5.2. Разработка технического проекта на Систему в особых случаях

49) В отдельных случаях при создании критически важных, дорогостоящих Систем допускается в качестве первой очереди осуществить ее проектирование. Проектирование Системы ведется на основе Технического задания на проектирование Системы и, при необходимости, Концепции создания и развития Системы.

50) В результате работ по проектированию Системы должны быть разработаны Технический проект на Систему и ее части (подсистемы), а также иные документы по решению Ведомства, необходимые для разработки Системы, например, проекты технических заданий на создание Системы и/или ее части (подсистемы). При этом в рамках разработки проектных и архитектурных решений на Систему должны быть определены ее функциональные и технологические (например, подсистема информационной безопасности) подсистемы, а также порядок (очередность) создания и внедрения этих подсистем.

51) В рамках проектирования рекомендуется проведение исследовательских работ по формированию альтернативных вариантов архитектуры Системы (гипотез), выбору рациональных вариантов, например, по критерию эффективность/стоимость с учетом заданных ограничений, а также технологий создания и соответствующей инфраструктуры (программно-аппаратных средств) Системы.

52) Полученные результаты исследований и проектирования должны уточняться в рамках создания и развития Системы. Независимо от того, насколько хорошо Система изначально определена и спроектирована, реальные потребности клиентов и выбор технологический решений являются неопределенными и, соответственно, развивающимися. Поэтому понимание того, как Система должна быть реализована, должно адаптироваться с течением времени. Исходя из этого при разработке Технического проекта рекомендуется придерживаться следующих правил:

1. Проектирование осуществляется на основе требований к Системе, исходящих от заинтересованных сторон, в том числе – клиентов, эксплуатирующего систему персонала, должностных лиц Ведомства.
2. В ходе проектирования рекомендуется использование практик, сохраняющих как можно дольше рассмотрение возможных вариантов реализации Системы (гипотез) в процессе её создания и принятие обоснованных (например, с точки зрения лучших экономических результатов) технических решений только после проверки гипотез.
3. Рекомендуется отказ от детализации проектных решений на начальных этапах создания Системы и постоянное уточнение решений по архитектуре Системы в процессе ее создания и развития в зависимости от изменений реальных потребностей клиентов. На начальном этапе создания Системы нужно обеспечить разработку решений таким образом, чтобы можно было ответить на критические вопросы об объеме, стоимости, графике работ, определить используемые технологии, а также технические риски создания Системы.
4. Предлагается использовать принцип минимизации объема документации предполагающем, что документация Технического проекта должна определять ключевые архитектурные решения наиболее эффективным способом (в том числе, в виде моделей).
5. Рекомендуется проведение моделирования архитектуры Системы с целью определения общего для Команд разработки понимания ключевых решений по ее построению. При этом необходимо стремиться использовать максимально простые и распространенные нотации, гарантирующие что модели будут поняты не только разработчиками, но и всеми заинтересованными в создании Системы сторонами (рекомендуется использование «Типового соглашения о моделировании архитектуры государственной информационной системы»).
6. Проектные решения должны учитывать существующий технологический задел Ведомства, включая созданные ранее информационные Системы Ведомства, особенности их организации и эксплуатации, техническую инфраструктуру, а также нормативные документы Ведомства, определяющие принятые практики и шаблоны создания архитектурных решений, а также ограничения на архитектурные решения.
7. При разработке архитектурных решений все участники процесса создания Системы должны привлекаться к обсуждению архитектурных решений, что обеспечит понимание и принятие архитектуры всеми разработчиками, а также увеличит вероятность того, что в случае необходимости Команды разработки

будут готовы изменить архитектурные решения в процессе создания (развития) Системы.

8. Для создания отказоустойчивой Системы может быть использован подход, позволяющий Командам разработки независимо создавать, тестировать и развертывать свои части Системы (при этом также предполагается, что архитектурные решения позволят реализовывать элементы Системы независимо).
9. В ходе разработки архитектурных решений для выявления потенциальных угроз безопасности информации, обрабатываемой в Системе, рекомендуется выполнять моделирование угроз безопасности информации. По результатам моделирования угроз могут быть скорректированы (уточнены) требования по безопасности и архитектурные решения.

53) Технический проект на Систему и технико-экономическое обоснование согласовываются и утверждаются в порядке, определяемом Постановлением № 676, и используются для выполнения работ на последующей очереди.

54) Если для выполнения работ по проектированию Системы привлекаются внешние исполнители, рекомендуется заключение отдельного Государственного контракта.

3. СТАДИЯ ПЛАНИРОВАНИЯ

55) Стадия планирования проводится Ведомством с целью выполнения всех подготовительных работ, необходимых для дальнейшего создания и/или развития Системы. При этом Ведомство берет на себя обязательства по организации процесса создания или развития Системы, в том числе с учетом основных рекомендаций настоящего документа.

56) До начала стадии планирования Ведомство формирует **Рабочую группу по созданию Системы** в составе, утверждаемом решением должностного лица Ведомства, имеющего соответствующие полномочия. Рабочая группа, необходимая для выполнения планирования, должна включать следующие категории:

1. **Владелец Системы —** должностное лицо Ведомства, которое несёт финансовую и операционную ответственность за создание и развитие Системы в течение всего ее жизненного цикла и обеспечение требуемых результатов оказания государственных услуг и выполнения государственных функций с использованием Системы. Владельцы Системы наделяются полномочиями по согласованию концепции, бюджета и технического задания на Систему внутри Ведомства, определяют приоритеты реализации функциональности в соответствии с концепцией и ТЗ, утверждают результаты финальной приемки каждого релиза Системы, а также принимают решение о вводе релиза Системы в эксплуатацию.
2. **Группа ответственных за Систему** — должностные лица и специалисты Ведомства, отвечающие за сопровождение процессов создания и развития Системы, разработку необходимых документов в ходе создания и развития Системы, в том числе формирование и контроль реализации требований к Системе, соответствующих потребностям клиентов Системы, зафиксированных в ТЗ на Систему и в рамках установленного бюджета.

57) Группа ответственных за Систему, в общем случае, может включать следующих должностных лиц Ведомства:

а. руководитель Группы ответственных за Систему — директор департамента (начальник управления), курирующего создание Системы — определяет основные проектные и архитектурные решения и руководит приемкой Системы (версий Системы);

б. начальники управлений, отделов, ответственных за предоставление услуг, выполнение функций государственного управления и контроля — участвуют в разработке Концепции, ТЗ, а также в испытаниях Системы и приемке выполненных работ;

в. представитель (представители) управления (департамента) информационной безопасности Ведомства, ответственные, в том числе, за внедрение и использование системы менеджмента информационной безопасности (СМИБ) в соответствии с рекомендациями ГОСТ Р ИСО/МЭК 27002-2021;

г. представитель (представители) управления информационных технологий и обеспечения эксплуатации Ведомства;

д. иные должностные лица и специалисты по решению руководителя Ведомства.

58) Стадия планирования, выполняемая созданной Рабочей группой, включает:

а. разработку проекта Концепции создания Системы, в соответствии с Методическими рекомендациями по формированию концепции создания государственной информационной системы, в том числе включающей:

- основания для создания Системы;
- предпосылки, предшествующие решению о создании Системы;
- роль и место Системы в государственном управлении Российской Федерации;
- анализ документов, регламентирующих разработку Системы;
- анализ решений аналогичного назначения;
- описание всех Очередей создания системы;
- цели и задачи создания Системы, целевые показатели результативности создания Системы;
- описание возможных вариантов построения Системы и обоснование выбранного варианта реализации;
- формирование требований к системе защиты информации в соответствии с ГОСТ Р 51583-2014, и иным законодательством в сфере защиты информации, обрабатываемой в государственных информационных системах, а также проведение классификации Системы в соответствии с требованиями по защите информации;
- план мероприятий по созданию Системы (проект);
- обеспечение правовой основы создания Системы;

б. разработку технико-экономического обоснования (ТЭО) реализации Системы, включающего в себя оценку финансовых, трудовых и материальных ресурсов, необходимых для реализации требований к Системе, включая оценку указанных ресурсов для создания Системы, ввода ее в эксплуатацию, эксплуатации и в случае, если установлен срок эксплуатации Системы, оценку необходимых ресурсов для

вывода Системы из эксплуатации и дальнейшего хранения содержащейся в ее базах данных информации;

в. разработку проекта правового акта, являющегося основанием для создания Системы;

г. обеспечение проведения оценки целесообразности проведения мероприятий по информатизации и (или) их финансирования Министерством цифрового развития, связи и массовых коммуникаций Российской Федерации в соответствии с постановлением Правительства Российской Федерации от 1 июня 2004 г. № 260 «О Регламенте Правительства Российской Федерации и Положении об Аппарате Правительства Российской Федерации»;

д. подготовку и предоставление на утверждение уполномоченному должностному лицу Ведомства проекта Концепции, ТЭО и правового акта, являющегося основанием для создания Системы;

е. разработку Технического задания на Систему, включая требования по защите информации и персональных данных, обрабатываемых с Системе, а также модели угроз безопасности информации;

ж. согласование Технического задания с уполномоченным федеральным органом исполнительной власти, осуществляющим функции по выработке и реализации государственной политики и нормативно-правовому регулированию в сфере информационных технологий (в соответствии с Постановлением № 676);

з. согласование Технического задания на Систему (Очередь системы) и модели угроз безопасности информации с федеральным органом исполнительной власти в области обеспечения безопасности и федеральным органом исполнительной власти, уполномоченным в области противодействия техническим разведкам и технической защиты информации;

и. представление на утверждение уполномоченному должностному лицу Ведомства Технического задания на создание или развитие Системы (и ее части), включая требования к защите информации и модели угроз безопасности информации, обрабатываемой в Системе в порядке, установленном Постановлением № 676;

к. определение очередности реализации сформированных в техническом задании Групп требований к Системе, плановые сроки реализации и критерии приемки каждой группы требований;

л. выполнение мероприятий по определению Исполнителя[4] в соответствии с законодательством Российской Федерации.

59) После определения Исполнителя и заключения Государственного контракта, концессионного или иного соглашения с Исполнителем, в Рабочую группу включаются следующие его представители:

а. **Архитектор Системы** — технический специалист (или команда специалистов), которые разрабатывают и уточняют проектные (технические) решения, определяют архитектуру Системы и осуществляют архитектурный надзор в процессе создания и развития Системы.

б. **Системная команда** — команда для решения специальных задач по созданию инфраструктуры разработки, включающей необходимые программно-аппаратные средства для интеграции разработанной функциональности в составе Системы, проведения всех видов тестирования, демонстрации Систем и решений, развертывания новых версий Системы, а также предоставления доступа к инфраструктуре разработки для Рабочей группы.

60) Архитектор Системы, в соответствии с требованиями, указанными в техническом задании, выполняет разработку решений по архитектуре Системы[5] и документации на Систему, в том числе технической документации в объеме, необходимом для описания полной совокупности проектных решений (в том числе по защите информации) и достаточном для дальнейшего выполнения работ по созданию Системы, а также описания проектных (технических) решений, обеспечивающих реализацию одного или нескольких процессов предоставления государственных услуг, государственных функций, включая контрольно-надзорную деятельность, подлежащих автоматизации и/или цифровой трансформации и реализуемых посредством Системы. Конкретный перечень, требования к составу и оформлению документации на Систему определяется Техническим заданием.

61) Группа ответственных за Систему осуществляет приёмку и обеспечивает согласование и утверждение документации на Систему или ее части в соответствии с порядком, установленным Постановлением № 676.

---

[4] «Исполнитель» — юридическое лицо, независимо от организационно-правовой формы, или индивидуальный предприниматель, определяемые в соответствии с законодательством Российской Федерации о контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд либо в соответствии с бюджетным законодательством, выполняющие работы по созданию или развитию государственной информационной системы либо оказывающее услуги по эксплуатации государственной информационной системы (Постановление № 676).

[5] Архитектурные решения могут уточняться и дополнятся в рамках выполнения Инкрементов.

62) Системная команда подготавливает инфраструктуру (программно-аппаратных средств) разработки Системы с учетом согласованных проектных (технических) решений и решений по архитектуре Системы.

4. СТАДИЯ ВЫПОЛНЕНИЯ

63) Стадия выполнения, в соответствии с итерационным подходом к разработке, осуществляется в виде серий итераций, каждая из которых включает следующие виды работ, выполняемые итерационными циклами (*Раздел 4.1.1*):

а. разработка или адаптация программного обеспечения Системы с использованием итерационного подхода. Использование итерационного подхода позволит проводить регулярные демонстрации промежуточных результатов разработки Системы, оперативно корректировать проектные решения в рамках требований технического задания на создание или развитие Системы, а также осуществлять выпуск релизов Системы непосредственно в ходе ее создания;

б. разработка и уточнение рабочей документации на Систему и ее части, необходимой для выполнения работ по вводу Системы в эксплуатацию и ее эксплуатации, а также методической документации и порядка эксплуатации Системы, содержащих сведения, необходимые для выполнения работ по поддержанию уровня эксплуатационных характеристик (качества) Системы (в том числе по защите информации), установленных в проектных решениях;

в. пусконаладочные работы;

г. регулярное проведение системной демонстрации промежуточных результатов создания или развития Системы (рекомендуется каждые две недели);

д. проведение предварительных испытаний Системы по завершении реализации Групп требований.

64) На основе ТЭО, разрабатываемого на стадии «Планирование», определяется состав и численность разработчиков, привлекаемых для создания Системы. В зависимости от полученных оценок рекомендуется выбор одного из вариантов организации разработки с использованием итерационного подхода:

1. **базовый уровень**, предполагающий разработку Систем Рабочей группой численностью до 150[6] человек (*Раздел 0*),
2. **расширенный уровень**, включающий дополнительные, по отношению к базовому уровню, рекомендации по разработке Систем Рабочей группой численностью более 150 человек (*Раздел 4.2*).

[6] Указанная оценка является верхнеуровневой, в общем она может колебаться в диапазоне от 50 до 150 человек.

4.1. Базовый уровень

4.1.1. Общие рекомендации по организации работ на базовом уровне

65) Базовый уровень представляет собой минимальный набор ролей, методик и практик, необходимых для создания или развития Системы с использованием итерационного подхода. На базовом уровне рекомендуется выполнение работ коллективом до 150 человек.

66) В соответствии с итерационным подходом к разработке на базовом уровне работы выполняются итерационными циклами с рекомендуемой продолжительностью 12 недель (~ 1 квартал) — Инкрементами. В рамках каждого Инкремента по согласованию с представителями Ведомства, входящими в состав Рабочей группы, реализуется функциональность Системы (определяемая Группой требований), представляющая ценность для потребителей государственных услуг и государственных функций (клиентов), предоставляемых через Систему.

67) Непосредственное выполнение работ по созданию и развитию Системы выполняют **Команды разработки** — самостоятельные и самоорганизующиеся кросс-функциональные команды численностью 5…11 человек, имеющие в своём составе все необходимые компетенции для выполнения работ. Принципы и возможные варианты формирования Команд разработки указаны в *Разделе 2.4*. За непосредственное формирование команд отвечает Исполнитель с учетом принятых у него стандартов и управленческих подходов к организации работ.

68) Для успешного применения настоящих Методических рекомендаций на базовом уровне рекомендуется соблюдение нижеперечисленных условий:

1. Использование принципов бережливого производства и итерационного подхода в разработке (*Приложение № 1*). Эти принципы гарантируют, что выполнение работ осуществляется непрерывно, с минимальными временными потерями и с наилучшим качеством.
2. Планирование Инкрементов с непосредственным участием всей Группы разработки Системы. Планирование определяет перечень приоритетных задач на очередной Инкремент и связывает стратегическое понимание создания (цели создания Системы) и развитие Системы с реализацией.
3. Постоянная синхронизация статуса работ и корректировка взаимосвязанных планов Команд разработки, привлекаемых к созданию и развитию Системы. Синхронизация позволяет обеспечивать взаимодействие одновременно нескольких Команд разработки, ведущих разработку различных составных частей Системы, что в свою очередь обеспечивает устойчивость процесса разработки.

4. Включение в Инкремент специальной итерации для осуществления планирования, проверки и корректировки результатов Инкремента, используемой в качестве резервного временного буфера для завершения ранее запланированных работ итерации, проведения необходимых для дальнейшей разработки Системы исследований, подведения итогов выполненных работ, а также для подготовки к планированию и проведению планирования последующего Инкремента.
5. Проведение регулярной демонстрации Системы должностным лицам Ведомства и иным заинтересованным сторонам в интересах получения обратной связи, выявления и решения проблем, а также адаптации хода разработки к изменяющимся условиям.
6. Обеспечение частых и качественных релизов Системы в соответствии с требованиями ТЗ в интересах получения обратной связи от клиентов.

4.1.2. Ролевая структура Рабочей группы Системы

69) Для непосредственного выполнения работ на базовом уровне Рабочая группа *(Рисунок 9)* по созданию Системы должна включать:

1. **Команду разработки** — самоорганизующаяся кросс-функциональная команда, включающая группу специалистов из 5...11 человек, работающих в рамках Инкремента короткими фиксированными циклами (рекомендуется 2 недели) — Итерациями.
2. **Группу команд разработки** — несколько Команд разработки, совместно работающих над созданием и развитием Системы итерациями фиксированной длительности (рекомендуется — 12 недель) — Инкрементами.

70) Дополнительно в состав Рабочей группы помимо ролей, указанных в *Разделе 3* «Стадии планирования», включаются следующие роли:

1. **Менеджер процессов группы команд** — руководитель со стороны Исполнителя, отвечающий за организацию и содействие выполнению Группой команд Системы согласованных рабочих процессов, рекомендуемых настоящим документом, в интересах эффективной реализации требований ТЗ.
2. **Ответственный за реализацию** — член Команды разработки, ответственный за определение приоритетов и реализацию требований командой в рамках двухнедельных итераций (циклов) при сохранении концептуальной и технической целостности компонентов Системы и ее устойчивого функционирования.
3. **Менеджер процессов команды** — член команды разработки, основной обязанностью которого является содействие команде в следовании рабочим процессам в соответствии с требованиями настоящего документа.

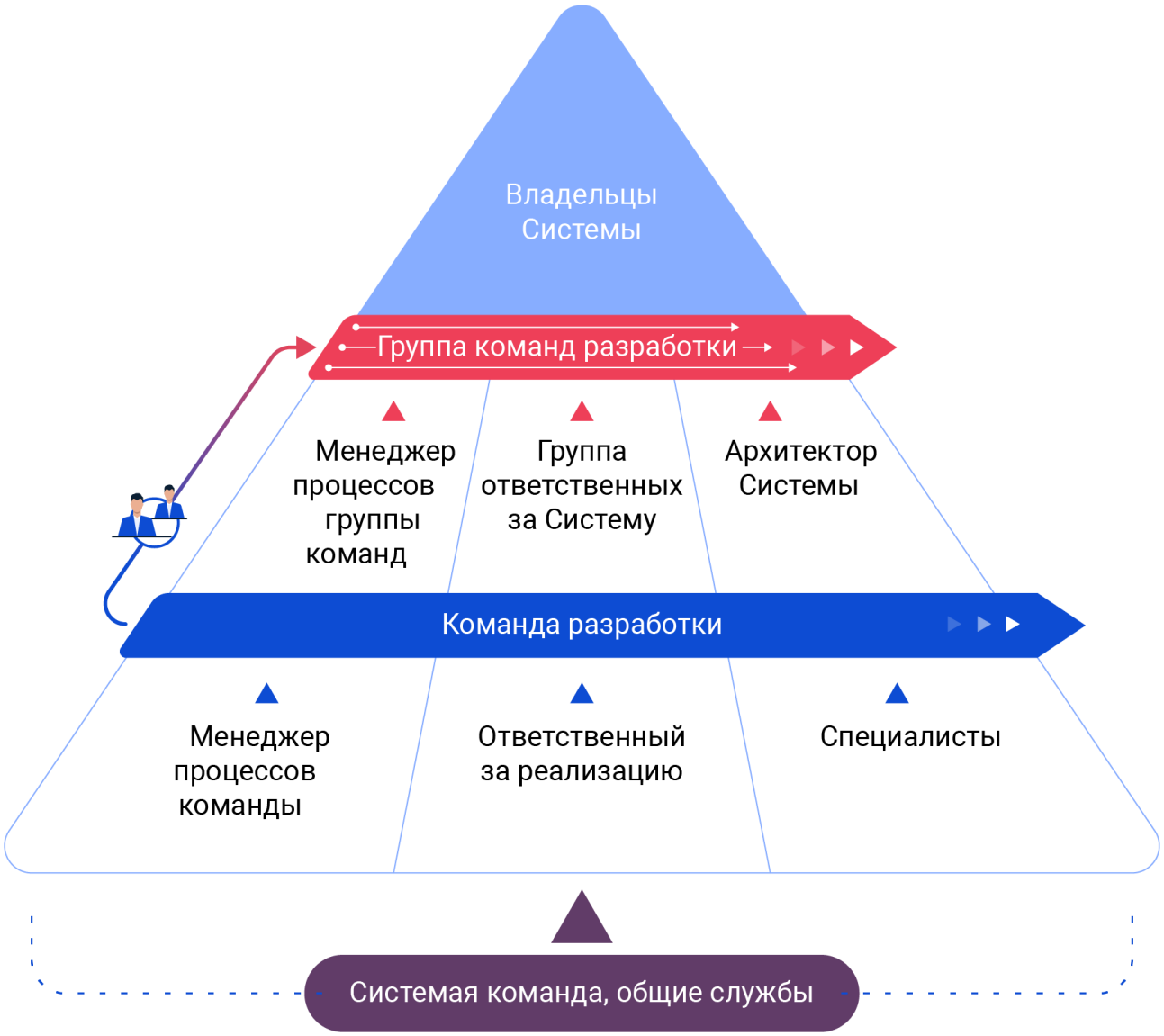

Рисунок 9. Структура Рабочей группы базового уровня

71) Формирование Рабочей группы, в соответствии с *nn.69)* и *70)*, обеспечивает Исполнитель.

72) Команды разработки должны иметь в своем составе специалистов, обеспечивающих во взаимодействии с Системной командой и другими Командами разработки проектирование, разработку, тестирование и развертывание функциональности Системы, предоставляющую ценность для клиентов.

73) Обязанности членов Рабочей группы определены в *Приложении № 3* к настоящему документу.

4.1.3. Выполнение работ по созданию и развитию Системы на базовом уровне на основе итерационного подхода

74) Типовая схема выполнения работ на основе итерационного подхода представлена на рисунке ниже *(Рисунок 10)*. В общем случае предполагается, что стадия выполнения работ выполняется серией Инкрементов, каждый из которых включает несколько итераций, реализующих:

а. планирование Инкремента,

б. выполнение Инкремента,

в. проверки результатов Инкремента,

г. корректировки выполняемых работ.

75) Рабочая группа может изменять типовую схему, исходя из практического опыта. При этом все Команды разработки, входящие в Группу, должны иметь одну и ту же частоту внутренних циклов выполнения работ.

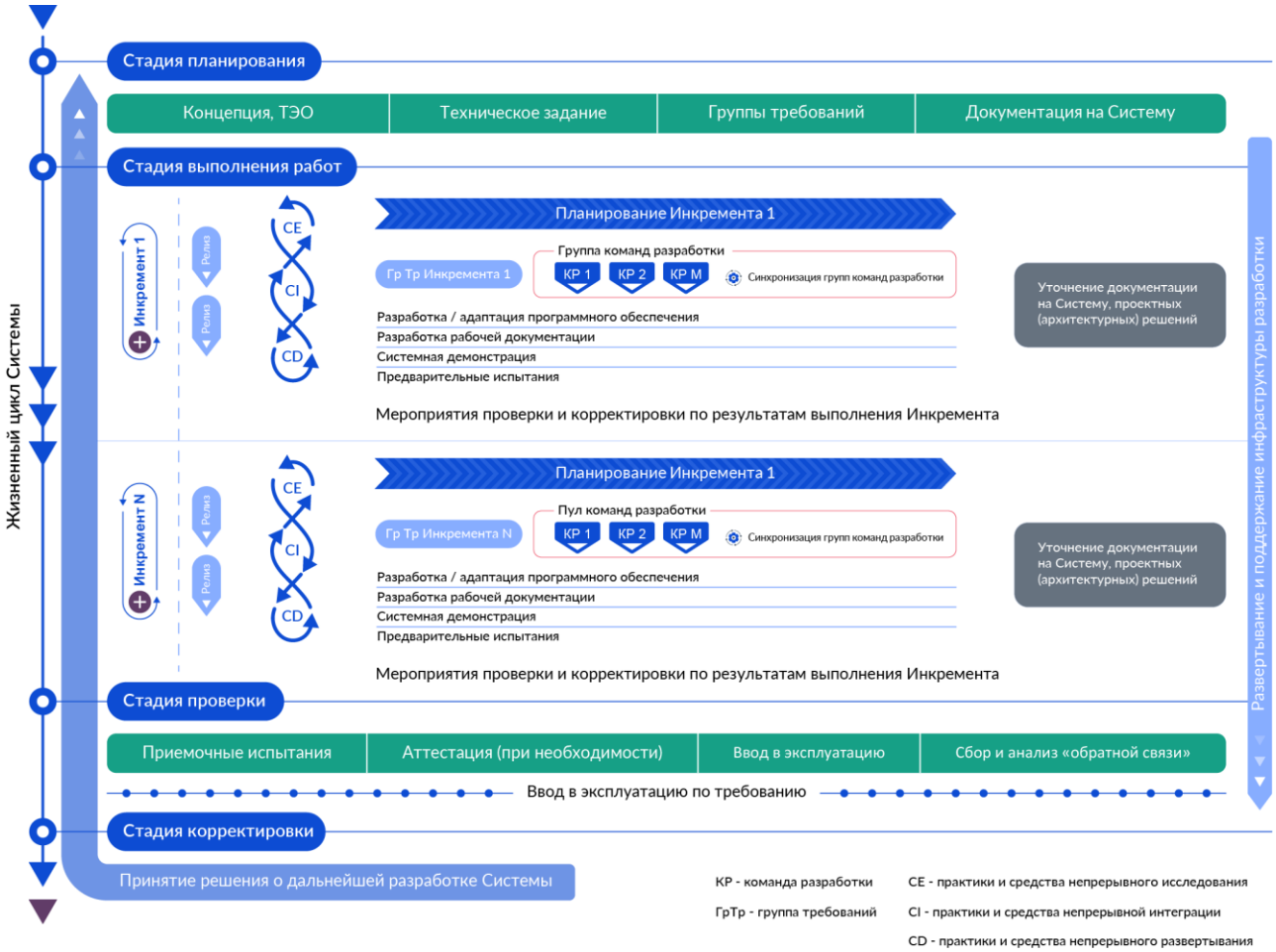

Рисунок 10. Типовая схема выполнения работ на базовом уровне

4.1.3.1. Планирование Инкремента

76) Планирование Инкремента — это регулярное мероприятие, которое служит для выработки единых целей, задач и согласования взаимосвязанных работ всех Команд разработки, участвующих в создании и развитии Системы. Планирование Инкремента включает подготовку к планированию и собственно проведение планирования.

77) В Планировании Инкремента участвуют Владельцы Системы, Группа ответственных за Систему, Команды разработки, архитекторы / инженеры Системы, системная команда и другие заинтересованные стороны. Все участники должны быть уведомлены заранее о проведении планирования Инкремента для подготовки к планированию.

78) Подготовка к планированию Инкремента осуществляется Группой ответственных за Систему с помощью Менеджера процессов Группы команд и с привлечением Архитектора Системы.

79) При подготовке к планированию:

1. Группа ответственных за Систему уточняет перечень задач, которые планируется выполнить в рамках предстоящей итерации, в том числе:
   - используя предстоящую к реализации в рамках планируемого Инкремента Группу требований, формируют цели (ожидаемые результаты) следующего Инкремента;
   - проводят работу с Ответственными за реализацию от Команд разработки по определению для каждой команды приоритетных функций предстоящего Инкремента;
   - определяют и оценивают основные факторы, риски и проблемы, влияющие на разработку Системы;
   - оценивают достаточность ресурсов Группы Команд разработки для выполнения предстоящего Инкремента.
2. Архитектор Системы с учетом предполагаемой к реализации функциональности Системы:
   - уточняет проектные (архитектурные) решения, обеспечивающие реализацию и интеграцию функциональности, реализуемой различными Командами разработки в предстоящем Инкременте, в том числе с учетом нефункциональных требований к Системе;
   - определяет необходимость создания вспомогательных технических и программных средств, необходимых для создания новых функций и возможностей Системы;

- определяет основные технологии, планируемые к использованию при разработке Системы, и нефункциональные требования.

3. Менеджер процессов Группы команд совместно с Группой ответственных за Систему:
   - определяет место проведения планирования, которое должно обеспечивать условия работы всех членов Рабочей группы разработки, участвующих в планировании;
   - обеспечивает доступ в режиме реального времени к информации и инструментам для поддержки распределенного планирования или удаленных участников;
   - обеспечивает готовность аудио и видео каналов для взаимодействия участников планирования, а также наличие иных презентационных материалов и технических средств.

80) По результатам подготовки формируются и предоставляются участникам планирования следующие входные данные и документы:

а. концепция создания и развития Системы, ТЗ, а также, при наличии, дополнения к ТЗ;

б. приоритеты реализации функциональных и нефункциональных требований в составе Группы требований в предстоящем Инкременте;

в. контекст выполнения планируемых работ, определяющий текущее состояние разработки Системы, применяемые технологии и средства для выполнения работ, ранее выявленные проблемы выполнения работ, внешние зависимости выполняемых работ от поставщиков и иных организаций (например, надзорных органов).

г. Перечень ошибок и запросов на изменения, выявленных в ходе эксплуатации Системы (частей Системы).

81) **Непосредственное планирование** Инкремента проводится, как правило, в течение двух дней.

82) **Первый день планирования** Инкремента включает следующие мероприятия:

1. Менеджер процессов группы команд представляет регламент процесса планирования и ожидаемые результаты.
2. Владельцы Системы описывают текущее состояние разработки Системы и дают оценку — насколько эффективно существующие решения удовлетворяют текущие потребности клиентов, а также определяют ожидаемые результаты (ценности

для клиентов), которые должны быть реализованы в ходе выполнения последующего Инкремента.

3. Группа ответственных за Систему представляет предложения по работам в рамках следующего Инкремента.
4. Архитектор / Инженер представляет уточненные архитектурные решения для реализации функциональности в предстоящем Инкременте, а также соответствующие изменения в методах разработки и в использовании практик DevSecOps, таких как непрерывная интеграция, непрерывное развертывание и автоматизация тестирования.
5. Команды разработки оценивают свои возможности и формируют проекты планов работ на предстоящий Инкремент, а также зависимости от выполнения работ других Команд разработки.
6. Команды разработки представляют участникам Планирования ключевые результаты планирования, которые включают в себя цели, трудоемкость, потенциальные риски, проблемы и зависимости от других команд, а также возможные решения по согласованию работ и решению проблем.
7. Проводится анализ и решение проблем, связанных с объемом работы, ресурсными ограничениями и зависимостями между командами. Члены Рабочей группы должны договориться об изменении объема работ и согласовать иные корректировки в планах каждой Команды разработки. При этом Менеджер процессов группы команд консультирует и обеспечивает совместную работу заинтересованных сторон до тех пор, пока не будут согласованы планы Команд разработки. В качестве инструмента достижения консенсуса рекомендуется использование досок зависимости, визуализирующих задачи каждой Команды разработки и связи с задачами, решаемыми другими командами *(Рисунок 11)*.

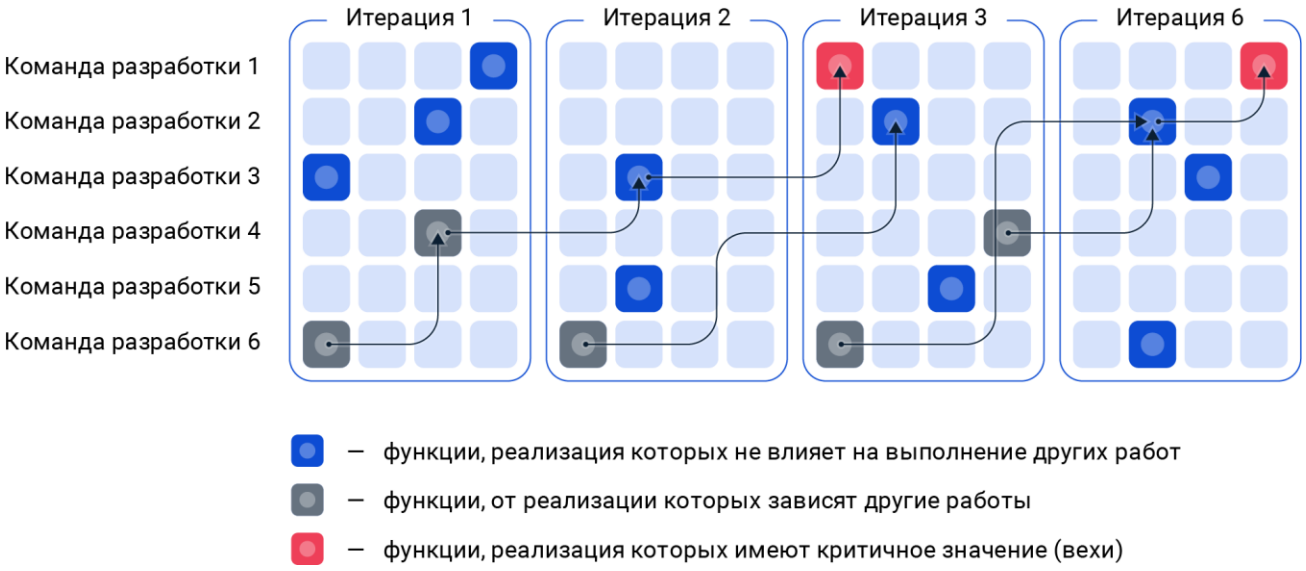

Рисунок 11. Доска зависимостей между командами разработки

83) **Второй день планирования** Инкремента включает следующие мероприятия:

1. Владельцы системы представляют изменения в объеме планирования и ресурсах по результатам договорённостей первого дня планирования.
2. Команды вносят коррективы в свои первоначальные планы, а также окончательно определяют свои цели на предстоящую итерацию и представляют свои планы всем участникам планирования.
3. Если Владельцы Системы считают план неприемлемым, командам предоставляется возможность их скорректировать.
4. Владельцы Системы определяют важности целей[7] *(п. 92)* для каждой Команды разработки на предстоящую итерацию.
5. Проводится обсуждение рисков и зависимостей, которые во время планирования выявили Команды разработки. Риски рассматриваются и классифицируются в одну из следующих категорий:
   - команды согласны с тем, что риск больше не является проблемой;
   - команды принимают риски и планируют мероприятия по управлению этими рисками в ходе выполнения Инкремента с целью снижения влияния риска;

[7] Важность целей определяется весовыми коэффициентами (оценками), для определения которых могут использоваться как неформальные, так и формальные методы, например, указанные в Приложении № 1.

- команды эскалируют риск на уровень Группы ответственных за Систему и/или Менеджера процесса группы команд — в случае если у команды нет полномочий и возможностей по управлению риском.

6. Команды голосуют о своей уверенности в достижении целей Инкремента с учетом выявленных рисков. При необходимости команды перерабатывают свои планы до тех пор, пока не будет достигнут высокий уровень доверия (~ 80%).
7. Менеджер процессов группы команд проводит краткую ретроспективу планирования.

84) **По результатам планирования** должны быть получены два основных результата:

1. Цели предстоящего Инкремента для каждой Команды разработки. При этом цели объединяют цели развития функциональности Системы и технические цели, которые Группа создания Системы намерена реализовать.
2. План работ для каждой Команды разработки с указанием дат реализации новых функций с учетом наличия зависимостей между работами других команд.

85) Все результаты процесса планирования фиксируются в целях Инкремента *(Таблица 2)* для каждой Команды разработки.

Таблица 2. Цели Инкремента для Команды разработки

Цели Инкремента № ___		**Важность цели**
1	Формирование отчетов о выполнении плана	
2	Периодический контроль выполнения работ	
3	Автоматическая регистрация выполнения работ	
4	Автоматическое уведомление об отклонениях от плана	
5	Ведение журнала оперативных событий	
6	Формирование отчетов о выполнении плана	
7	Экспорт отчетов	

86) Цели каждой Команды разработки часто напрямую связаны с предполагаемыми к реализации функциями. Однако такое сопоставление не всегда является однозначным, поскольку некоторые функции требуют совместной работы нескольких команд, как показано на рисунке ниже *(Рисунок 12)*. Некоторые функции (например, компонент ФТ 1) могут быть реализованы отдельной командой, тогда как другие (компонент ФТ 2) требуют совместной работы нескольких команд.

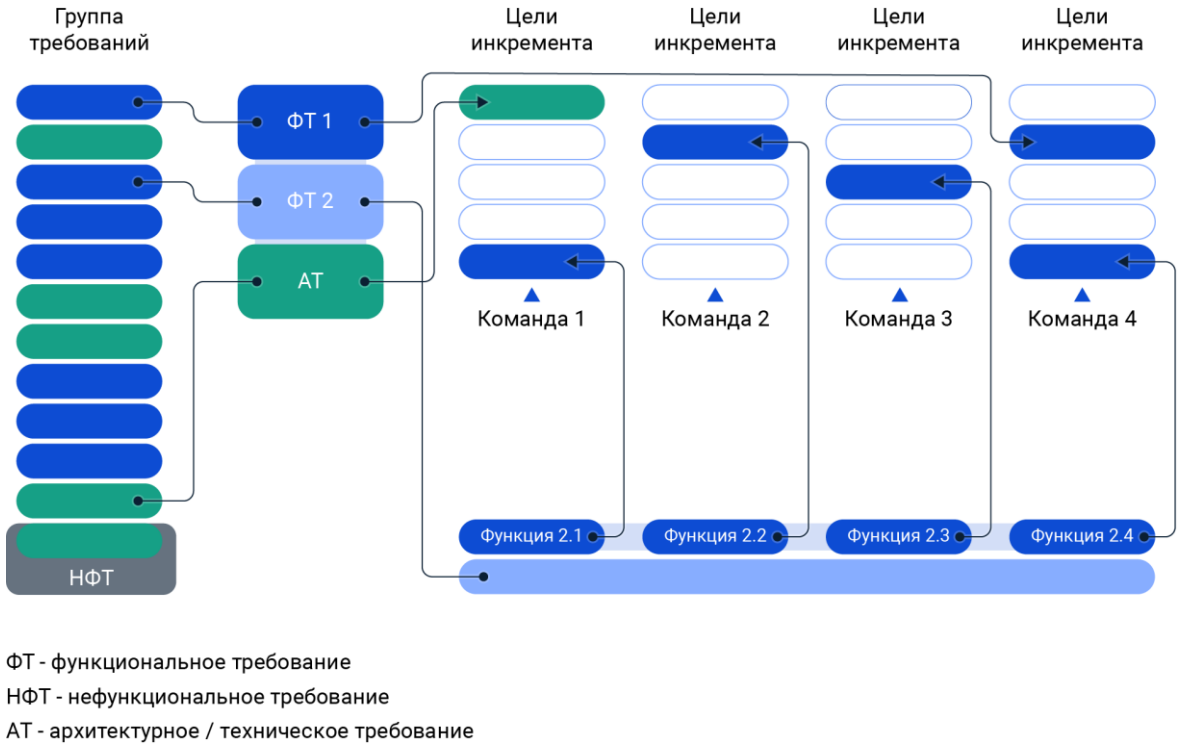

Рисунок 12. Распределение работ между Командами разработки

87) Дополнительно для команд в ходе планирования Инкремента могут быть определены технические архитектурные (технические) требования (АТ), которые обеспечивают реализацию функциональности, а также служат улучшению инфраструктуры разработки. Кроме этого, должны учитываться нефункциональные требования (НФТ) к безопасности, надежности, производительности, удобству обслуживания и использования, масштабируемости Системы.

88) В рамках выполнения Инкремента, как Команды разработки, так и Группа команд в целом, сталкиваются с проблемой — как сбалансировать выполнение работ по техническому обслуживанию, рефакторингу и развитию инфраструктуры разработки с работами по реализации новой функциональности, которая обеспечивает более непосредственную ценность для клиентов. Сосредоточение внимания исключительно на функциональности может работать некоторое время на достижение целей Системы, но через некоторое время вес нереализованных технологических решений замедлит скорость создания (развития) Системы в целом. Чтобы смягчить эту проблему, команды постоянно инвестируют в развитие архитектуры Системы, что позволяет избежать необходимости массовой замены инфраструктуры Системы из-за технологического устаревания.

89) Существуют обстоятельства, при которых по результатам планирования у команды остается низкая уверенность в достижении целей:

а. зависимости от другой команды или поставщика, которые не могут быть гарантированы;

б. команда не имеет опыта работы с функциональностью данного типа;

в. команда уже загружена почти на полную мощность, выполняя критические задачи, от которых зависит успех Инкремента в целом.

90) Такие цели (не рекомендуется более трёх) также могут быть включены в План работ команды на предстоящий Инкремент, и для их выполнения должны быть спланированы соответствующие ресурсы. При этом команды не фиксируют их в качестве своих обязательств, чтобы в случае невыполнения не ухудшать свои показатели по результатам выполнения Инкремента. Дополнительные цели, обязательства достижения которых не определены, называются **незафиксированными целями.**

91) В случае, если на момент проведения планирования отдельный компоненты (части) Системы введены в эксплуатацию для исправление возникших ошибок и проведения доработок ранее реализованных функциональных возможностей Системы при планировании Инкремента необходимо:

1. включить исправление выявленных ошибок непосредственно в цели Команд разработки;
2. определять лимит времени для исправления ошибок, при этом определять исправление ошибок как «незафиксированные цели».

92) После завершения определения целей Инкремента, Владельцы Системы с привлечением Группы ответственных за Систему определяют важность целей для каждой Команды разработки в рамках очного обсуждения с ними в ходе второго дня планирования, например, как показано на рисунке ниже *(Таблица 3)*. При этом следует выполнять следующие правила:

а. использовать шкалу от 1 (самая низкая) до 10 (самая высокая) для оценки каждой цели;

б. оценки не должны быть «нормализованы»[8] между командами, то есть у каждой команды может быть несколько самых приоритетных (рейтинг 10) пунктов;

в. принимать к учету как цели, которые обеспечивают прямую и непосредственную ценность, так и цели, которые обеспечивают создание будущей ценности, такие как, например, изменения в области инфраструктуры, сред разработки и инициатив в области качества;

[8] Под нормализацией в общем случае понимается преобразование всего набора данных к безразмерным единицам в рамках заданного диапазона, например, 1…10

г. включать в оценку незафиксированные цели.

Таблица 3. Конечные цели Команды разработки с незафиксированной целью и оценкой ценности

Цели Инкремента № ___		Важность цели
1	Формирование отчетов о выполнении плана	6
2	Периодический контроль выполнения работ	2
3	Автоматическая регистрация выполнения работ	10
4	Автоматическое уведомление об отклонениях от плана	8
5	Ведение журнала оперативных событий	3
6	Формирование отчетов о выполнении плана	7
7	Экспорт отчетов	2
Незафиксированные цели		
8	Реестр контрольных мероприятий	2
9	Реестр оперативных событий	3

4.1.3.2. Выполнение Инкрементов

93) Инкремент выполняется Группой команд разработки, при этом каждая Команда разработки, входящая в состав Группы команд, обеспечивает достижение целей, согласованных в ходе Планирования инкремента, а также реализацию зависимостей с другими командами, при этом план (порядок) выполнения работ (итераций) может корректироваться.

94) Команды разработки выполняют работы синхронно, итерациями фиксированной продолжительности (рекомендуется — 2 недели) с проведением синхронизации в рамках Группы команд разработки после завершения каждой итерации *(Рисунок 13)*. Организация процесса итерации определяется выбранным методом итерационной разработки. Например, для Команд разработки, использующих метод Скрам (*Приложение №1)*, рабочий процесс в рамках итерации включает:

а. планирование итерации, предполагающее формирование задач и целей на итерацию и определение приоритетов их выполнения;

б. ежедневные совещания с целью понять текущий статус работ, проанализировать проблемы и получить помощь от других членов Команды разработки;

в. обзор результатов итерации и демонстрацию реализованной функциональности в интересах получения обратной связи от заинтересованных сторон по результатам разработки запланированной функциональности Системы;

г. проведение ретроспективы итерации, включающей оценку процесса разработки, а также выявление проблем, которые необходимо решить на следующей итерации;

д. уточнение перечня реализуемых функций (бэклога) в рамках следующих итераций.

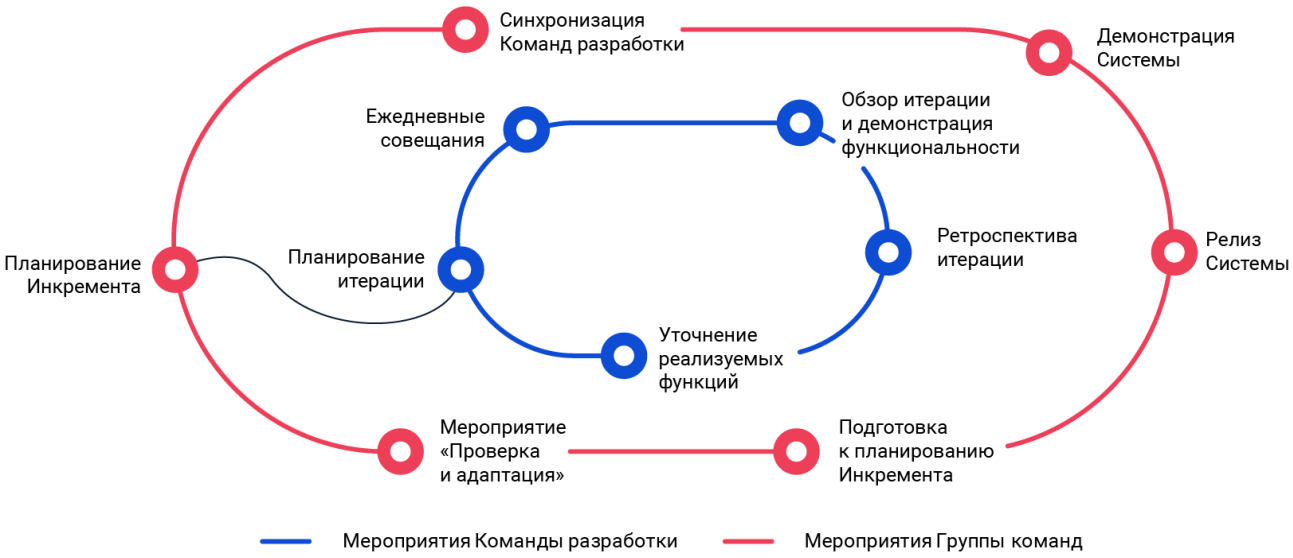

Рисунок 13. Мероприятия Группы команд разработки на базовом уровне

95) За контроль соответствия процессов выбранному методу итерационной разработки, а также наставничество над специалистами Команды разработки отвечает Менеджер процессов.

96) В ходе выполнения работ Команды разработки в рамках согласованных Планов работ осуществляют:

а. разработку и адаптацию программного обеспечения — реализацию функциональных и нефункциональных требований к Системе;

б. разработку и актуализацию рабочей документации, а также уточнение, при необходимости, архитектурных решений и технической документации на Систему (перечень, требования к составу и оформлению документации на Систему определяется Техническим заданием);

в. проведение всех видов тестирования — функционального, интеграционного, регрессионного, а также, при необходимости, нагрузочного;

г. выполнение иных мероприятий, предусмотренные согласованным в рамках планирования Инкремента Планом работ.

97) В качестве основного инструмента для выполнения работ Команде разработки рекомендуется использовать Конвейер непрерывной разработки, основанный на принципах DevSecOps (*Приложение №1*) и обеспечивающий повторяемые рабочие процессы и их автоматизацию, необходимые для создания новой функциональности Системы, начиная от разработки проектных решений до выпуска релиза в эксплуатацию и в последующем обеспечения эксплуатации. Конвейер состоит из четырех компонентов *(Рисунок 14):*

1. **Непрерывное исследование (CE)** — набор практик и необходимых технических и программных средств, обеспечивающий возможность Командам разработки постоянно повышать уровень знаний в своей предметной области, формировать технологический задел и определять рациональные пути создания и развития Системы, а также быстро адаптировать средства разработки и применяемые технологии к изменяющимся потребностям клиентов (ключевой практикой CE является создание МДВ или определение набора наиболее востребованных клиентами функций). Рекомендуемой методологией проведения непрерывного исследования является дизайн-мышление (*Приложение №1*).
2. **Непрерывная интеграция (CI)** — набор практик и средств автоматизации процессов разработки, сокращающих временные затраты на проверку и выпуск версий Системы путём регулярного объединения программного кода, разработанного разными командами в специальном репозитории — тестовом контуре, с последующей автоматической сборкой, тестированием и запуском Системы с реализованной новой функциональностью.
3. **Непрерывное развертывание (CD) —** технология автоматизации сборки версии Системы, тестирования и развертывания с последующим автоматическим (без ручных вмешательств) развертыванием в среде эксплуатации. При этом развертывание отделяется от релиза. Функциональные возможности развертываются в контуре эксплуатации непрерывно, но предоставляются конечным пользователям только по требованию (решению Владельца Системы).
4. **Релиз по требованию (непрерывная доставка)** — технология, обеспечивающая предоставление (по решению Владельца Системы) релиза Системы пользователям, с учетом последних изменений, внесенных Командами разработки.

98) Дополнительно используемые инструменты Команд разработки должны соответствовать требованиям ГОСТ Р 56939, ГОСТ Р 58412 и включать инструменты, обеспечивающие автоматические проверки информационной безопасности и мониторинг Системы на предмет атак и утечек данных на всех участках

производственного процесса, включая планирование, программирование, интеграцию, тестирование, выпуск релиза и развертывание.

99) Наличие инструментов DevSecOps обеспечивает Исполнитель, при этом Группа ответственных за Систему проводит контроль развернутых инструментов DevSecOps на этапе планирования создания и развития Системы, в том числе на соответствие используемого конвейера требованиям информационной безопасности (*Приложение № 1*).

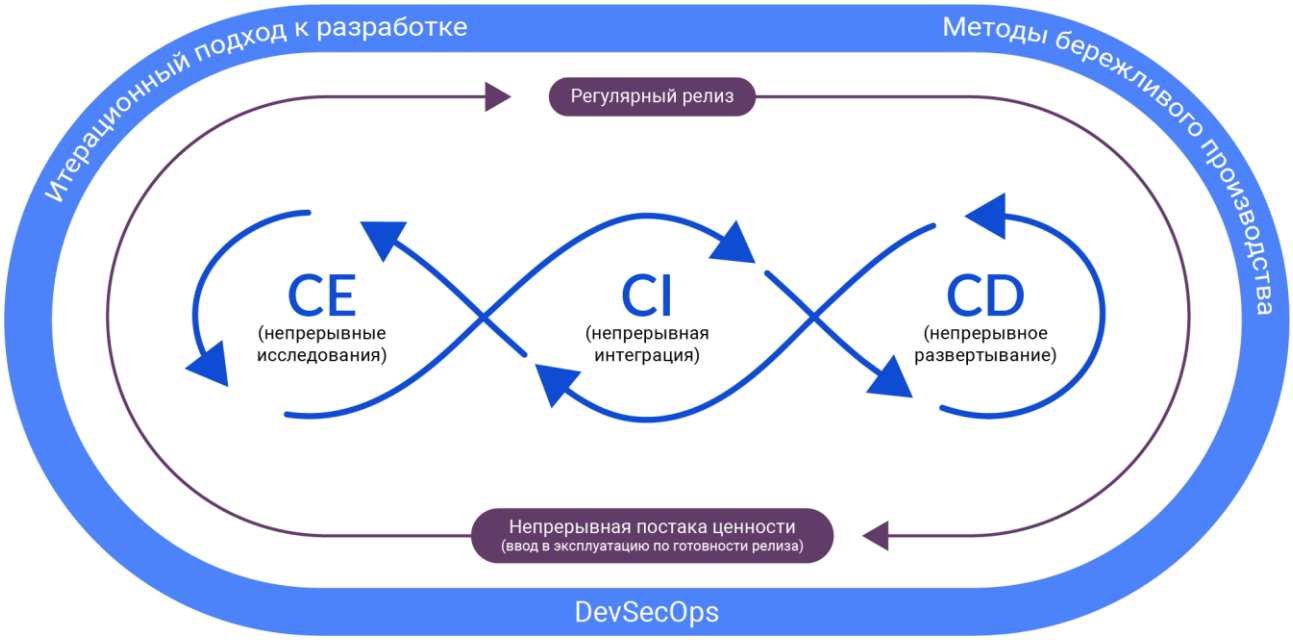

Рисунок 14. Конвейер непрерывной разработки

100) В ходе выполнения работ Командам разработки рекомендуется использование методов визуализации и управления задачами, например, с использованием досок Канбан (Приложение №1), обеспечивающих:

а. демонстрацию текущего статуса выполнения работ, выявление «узких» проблемных мест, в случае их возникновения, и определение возможностей для их устранения;

б. соблюдение ограничений на количество одновременно выполняемых работ с учетом ресурсных возможностей Команды разработки;

в. оценку производительности и непрерывное совершенствование производственного процесса.

Ведение досок Канбан возлагается на представителей Команд разработки, а контроль — на Менеджера процессов команд.

101) С целью минимизации влияния рисков и контроля влияния зависимостей в ходе выполнения работ проводятся регулярные мероприятия по синхронизации работ, выполняемых различными Командами разработки:

1. Менеджер процессов группы команд и представители каждой Команды разработки (как правило, Менеджеры процессов команды) регулярно проводят еженедельные встречи и рассматривают прогресс в достижении целей Инкремента, выявляют проблемы, а также проводят анализ принятых рисков зависимостей между командами. Результаты еженедельных встреч представляются Группе ответственных за Систему.
2. Группа ответственных за Систему с привлечением Ответственных за реализацию от каждой Команды разработки проводит еженедельные совещания с целью получить представление о том, насколько хорошо Группа команд разработки обеспечивает достижение целей Инкремента, обсудить проблемы с разработкой функций и выработать, в случае необходимости, корректировки. Для подготовки к мероприятиям по синхронизации необходимые материалы готовят и предоставляют Ответственные за реализацию.

102) Архитектор и специалисты Команд разработки, в соответствии с техническим заданием, разрабатывают рабочую документацию, включающую техническую документацию, необходимую для выполнения работ по вводу Системы в эксплуатацию и ее эксплуатации, методическую документацию и порядок эксплуатации Системы, содержащие сведения, необходимые для выполнения работ по поддержанию уровня эксплуатационных характеристик (качества) Системы (в том числе по защите информации), установленных в проектных (технических) решениях, в том числе:

а. перечень действий сотрудников при выполнении задач по эксплуатации Системы, включая перечень, виды, объемы и периодичность выполнения работ по обеспечению функционирования Системы;

б. контроль работоспособности Системы и компонентов, обеспечивающих защиту информации;

в. перечень неисправностей, которые могут возникнуть в процессе эксплуатации Системы, и рекомендации в отношении действий при их возникновении;

г. перечень режимов работы Системы и их характеристики, а также порядок и правила перевода Системы с одного режима работы на другой с указанием необходимого для этого времени.

103) Дополнительно, в случае если при реализации Инкремента вносились изменения (уточнялись) в проектные и архитектурные решения, Архитектор и специалисты

Команд разработки осуществляют доработку документации на Систему таким образом, чтобы она однозначно и адекватно отражала реализованные в рамках Инкремента решения.

104) Согласование и утверждение рабочей документации (а также, при необходимости, доработанной документации на Систему) осуществляет Группа ответственных за Систему под руководством (контролем) Владельца Системы в порядке, установленном Постановлением № 676.

105) В конце каждого Инкремента должна быть выполнена интеграция результатов работ, выполненных различными Командами разработки.

106) В интересах сокращения временных и ресурсных затрат на подготовку к интеграции в рамках Инкремента должна выполняться интеграция реализованной функциональности в рамках одной или нескольких итераций, выполняемых Командами разработки. Решение о частичной интеграции принимает Группа ответственных за Систему и согласовывает его с Ответственными за реализацию.

107) Для подтверждения успешности результатов выполненных работ Команды разработки проводят совместное интеграционное и регрессионное тестирование Системы с установленными обновлениями функциональности в соответствии с предварительно разработанными и согласованными с Группой ответственных за Систему программами и методиками тестирования.

108) В ходе выполнения работ рекомендуется использовать практики, повышающие эффективность тестирования (*см. Приложение № 1*), в том числе:

1. Практику раннего тестирования - вместо того, чтобы выполнять большое количество тестов в конце Инкремента, Команды разработки должны выполнять тесты на ранних стадиях разработки кода – в рамках двухнедельных итераций. Раннее тестирование применяется как к функциональным требованиям, так и к нефункциональным требованиям, к производительности, надежности и другим требованиям к качеству Системы, указанным в ТЗ.
2. Использование сокращенных тестов - поскольку тесты могут расти с течением времени (по мере роста объема кода), и они начинают вносить задержки в работу команд, допустимо использование сокращенных тестов.
3. Использование эффективных инструментов производственных конвейеров (DevSecOps) - для повышения скорости тестирования может быть обеспечено использование более производительного оборудования в тестовых контурах. Для этого Команды разработки должны взаимодействовать с Системной командой, чтобы сбалансировать скорость и качество тестирования.

109) В рамках эксплуатации отдельных релизов Системы (в том числе МДВ) Группой ответственных за Систему должны быть организованы прием и регистрация обращений клиентов, консультации клиентов, а также исправление ошибок и проведение доработок ранее реализованных функциональных возможностей Системы.

110) Решения возможных проблем функционирования Системы должно осуществляться в рамках принятых при вводе Системы в эксплуатацию соглашений об уровне обслуживания. При этом Группа ответственных за Систему совместно с Менеджером процессов группы команд и Архитектором Системы в рамках выполнения Инкремента должны обеспечить:

1. сотрудничество между Командами разработки и иными заинтересованными сторонами, входящими в состав Рабочей группы по всему потоку создания ценности для выявления и решения проблем по мере их возникновения;
2. разработку механизмов отработки ошибок (сбоев), чтобы обеспечить быстрое возобновление функционирования Системы в случае аварийных ситуаций, в том числе планирование и отработка аварийного восстановления Системы;
3. непрерывный мониторинг безопасности, предполагающий постоянное тестирование Системы на наличие недавно обнаруженных и зарегистрированных уязвимостей, а также обнаружение вторжений и атак на инфраструктуру.
4. мониторинг нефункциональных требований (НФТ) — чтобы избежать сбоев в обслуживании, такие системные атрибуты, как надежность, производительность, ремонтопригодность, масштабируемость и удобство использования, должны постоянно контролироваться.

4.1.3.3. Проверка результатов Инкремента

111) Проверка результатов Инкремента осуществляется путём проведения демонстрации реализованных всеми Командами разработки запланированных функциональных возможностей Системы, а также, в случае ввода части Системы в эксплуатацию - путём проведения установленных Постановлением №676 видов испытаний. Организует проведение демонстрации Менеджер процессов группы команд совместно с Группой ответственных за Систему.

112) Перед демонстрацией Системы должны быть выполнены пусконаладочные работы, включающие автономную наладку технических средств и разработанного (или адаптированного) программного обеспечения Системы, загрузку информации в ее базу данных, комплексную наладку и тестирование технических средств и программного обеспечения Системы, включая средства защиты информации.

113) **Демонстрация Системы** — важное событие, которое является механизмом получения Группой команд разработки обратной связи и обеспечивает комплексное представление новых функций, реализованное Группой команд разработки в рамках Инкремента. Демонстрация проводится перед всеми заинтересованными сторонами и дает Владельцам Системы, Группе ответственных за Систему и иным заинтересованным сторонам объективную оценку прогресса разработки Системы. К демонстрации Системы по решению Ответственного за Систему могут **привлекаться клиенты**, что позволит ускорить получение обратной связи о качестве о востребованности результата.

114) Демонстрация проводится в тестовом контуре, который максимально точно должен соответствовать контуру эксплуатации Системы.

115) В рамках подготовки к демонстрации:

1. Системная команда проводит установку и настройку тестового контура Системы, включая подготовку тестовых данных, не содержащих конфиденциальной информации и персональных данных.
2. Команды разработки выполняют следующие мероприятия:
   - разрабатывают и согласовывают Группой ответственных за Систему методики регрессионного и интеграционного тестирования, а также тестирования в интересах подтверждения выполнения требований по безопасности;
   - устанавливают обновления Системы, включающие реализованные как функциональные, так и нефункциональные требования, а также проводят интеграционное и регрессионное тестирование;
   - проводят тестирование в соответствии с разработанными и согласованными программами и методиками;
   - фиксируют и исправляют выявленные ошибки, при этом, по согласованию с Группой ответственных за Систему часть ошибок может быть исправлена рамках следующего Инкремента.

116) В ходе демонстрации Системы должна быть подтверждена реализованная в рамках Инкремента функциональность Системы, а также осуществляться проверка соответствия Системы наиболее критичным нефункциональным требованиям (например, требованиям надёжности, информационной безопасности). При этом для проведения проверок на соответствие специальным (нефункциональным) требованиям Ответственные за Систему могут привлекать Ведомства, обладающими необходимыми полномочиями.

117) По результатам демонстрации Системы проводится оценка текущего состояния разработки Системы. Результаты проверки (демонстрации) реализованной в рамках

Инкремента функциональности (компонента) Системы оформляются протоколом, в котором дается заключение о соответствии функциональности (компонента) Системы требованиям технического задания и возможности при необходимости передачи в эксплуатацию этого компонента. Протокол оформляется Группой Ответственных за Систему и представляется Владельцу Системы на утверждение. Результаты демонстрации Системы учитываются при планировании следующего Инкремента, а также при принятии решения о выпуске релиза и включаемых в состав релиза функций.

118) В процессе выполнения работ по созданию и развитию Системы Группа команд разработки должна обеспечивать готовность к развертыванию релиза Системы в контуре эксплуатации каждый раз, когда это требуется Ведомству. При этом в состав релиза могут включаться функции, реализованные и протестированные в рамках предшествующих Инкрементов.

119) В большинстве ситуаций завершение Инкремента и планируемый момент выпуска релиза Системы могут совпадать, но в некоторых случаях может потребоваться выпускать релиз Системы более или менее часто (например, через полтора Инкремента или через пол Инкремента).

120) Решение о развертывании релиза Системы, составе релиза и предоставлении доступа к реализованной функциональности клиентам принимает Владелец Системы на основе предложений, разработанных Группой ответственных за Систему.

121) Развёртывание релиза Системы предполагает:

а. выполнение пусконаладочных работ;

б. проведение предварительных испытаний Системы;

в. устранение выявленных ошибок.

122) Выполнение пусконаладочных работ предполагает наладку технических и программных средств, загрузку (заполнение, например, баз данных) Системы первоначальной информацией, необходимой для проведения испытаний и начала функционирования Системы, комплексную наладку технических средств и программного обеспечения Системы, включая средства защиты информации.

123) Проведение предварительных испытаний включает:

а. разработку программы и методики предварительных испытаний, в соответствии с которыми осуществляется проверка Системы на работоспособность и соответствие техническому заданию на ее создание;

б. проверку Системы на работоспособность и соответствие техническому заданию на ее создание;

в. устранение выявленных при проведении таких испытаний неисправностей и внесение изменений в документацию и рабочую документацию на Систему;

г. оформление протокола испытаний и акта о приемке Системы в опытную эксплуатацию.

124) По решению Владельца Системы и в случаях, определенных Техническим заданием, часть Системы, реализующая функциональность одной или нескольких групп требований и обеспечивающая определенные возможности клиентам Системы, может быть введена в эксплуатацию. С этой целью проводятся установленные Постановлением №676 виды испытаний, описанные в *Разделе 5* настоящего документа. При этом допускается:

а. проводить приемочные испытания в соответствии с в отношении отдельных частей Системы, с учетом проверки полноты и качества выполнения ими функций в составе эксплуатируемой Системы,

б. в ходе проведения приемочных испытаний выполнять мероприятия по аттестации в Системы на соответствие требованиям информационной безопасности.

125) Результаты проверки (испытаний) частей Системы оформляются протоколом, в котором дается заключение о соответствии Системы и ее части требованиям технического задания и возможности эксплуатации Системы с учетом реализованной в рамках Инкремента новой функциональности.

126) Выполнение мероприятий по вводу в эксплуатацию отдельных релизов Системы, в дальнейшем обеспечит существенное сокращение времени на ввод в эксплуатацию последующих версий Системы, а также Системы в целом. Например, мероприятия по повторной аттестации Системы при вводе в эксплуатацию релизов могут не проводиться в случае, если их использование в составе эксплуатируемой Системы не нарушает установленных к ней требований о защите информации.

4.1.3.4. Корректировка выполняемых работ по результатам Инкремента

127) Корректировка выполняемых работ по созданию и развитию Системы по результатам Инкремента проводится в рамках мероприятия «Проверка и Адаптация», организуемого Менеджером процессов группы команд с привлечением Группы ответственных за Систему.

128) Проверка и адаптация выполняется по завершению Инкремента и является временем (ориентировочно 2 итерации) для анализа выполненных работ, решения проблем и принятия мер по улучшению, необходимых для увеличения скорости, качества и надежности следующего Инкремента.

129) Участниками мероприятия «Проверки и Адаптация» должны быть, при возможности, все члены Рабочей группы.

130) Мероприятие «Проверка и адаптация» (ПА) включает:

1. рассмотрение количественных и качественных показателей результатов выполненных работ;
2. проведение ретроспективы;
3. выполнение дополнительных мероприятий по решению представителей Группы ответственных за Систему и Ответственных за реализацию.

131) Во время **рассмотрения количественных и качественных показателей** рассматриваются все цели, которые Команды разработки определили в ходе планирования Инкремента. В ходе мероприятия «Проверка и адаптация» Группа ответственных за Систему совместно с представителями каждой Команды разработки оценивает результаты, достигнутые для каждой из поставленных в ходе планирования Инкремента целей *(Таблица 4)*, на основе которых могут быть выполнены оценки результатов выполнения Инкремента в целом, например, по сумме оценок для каждой цели (за исключением оценок незафиксированных целей).

132) Одним из основных показателей, принимаемых к учету для оценки результатов Команд разработки является показатель предсказуемости результатов Команд разработки, определяемый как отношение фактических и запланированных оценок выполнения Инкрементов каждой Командой разработки (в %) на интервале нескольких Инкрементов *(Рисунок 15)*. Такой же показатель может быть использован для Группы команд в целом как среднее значение показателей Команд разработки, входящих в Группу команд.

133) Надежные Команды разработки должны работать в диапазоне предсказуемости 80–100 %, что в свою очередь обеспечивает предпосылки для надежного планирования. При этом незафиксированные цели не учитываются в обязательствах команд, но учитываются в фактическом достижении цели.

Таблица 4. Оценка результатов квартального цикла

Цели Инкремента № ___		Важность цели	
		План	Факт
1	Формирование отчетов о выполнении плана	6	5
2	Периодический контроль выполнения работ	2	2
3	Автоматическая регистрация выполнения работ	10	7
4	Автоматическое уведомление об отклонениях от плана	8	8
5	Ведение журнала оперативных событий	3	3
6	Формирование отчетов о выполнении плана	7	6
7	Экспорт отчетов	2	2
Незафиксированные цели			
8	Реестр контрольных мероприятий	2	0
9	Реестр оперативных событий	3	3
	ИТОГО	**43**	**36**
	% достижения цели	**84%**	

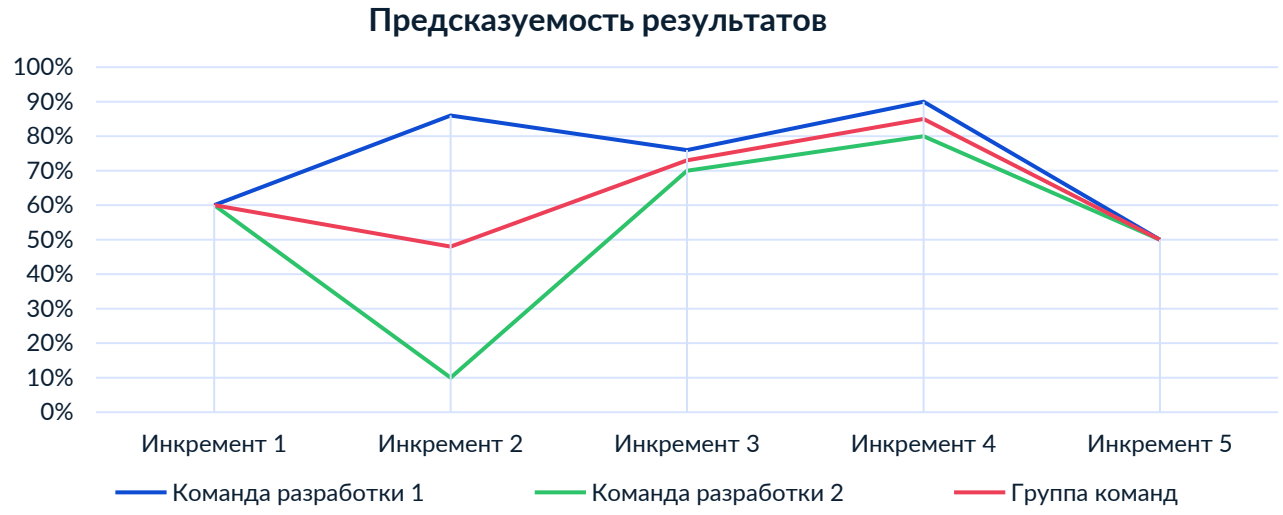

Рисунок 15. Показатель предсказуемости работы Команды разработки

134) **Ретроспектива** проводится с целью определения проблем, с которыми столкнулась Рабочая группа в рамках текущего Инкремента и проведения их дальнейшего обсуждения и решения в ходе ретроспективных семинаров. При этом каждая проблема может обсуждаться с привлечением Команд в полном составе или с привлечением специально сформированных групп из представителей разных Команд разработки. Такой подход к организации проведения семинара позволяет обеспечить

различные взгляды на проблему и учесть мнение тех, кто больше всего мотивирован для решения проблемы.

135) Ретроспективный семинар организует и проводит Менеджер процессов группы команд с привлечением Менеджеров процессов команд разработки. Владельцы Системы и Группа ответственных за Систему принимают участие в ретроспективных семинарах в интересах обеспечения решения проблем, которые находятся вне сферы ответственности Команд разработки.

136) В ходе ретроспективного семинара проводится выявление источников выявленных проблем, ранжирование источников проблем и определение инструментов и способов их решения, а также формируются предложения по учету сформированных рекомендаций в ходе планирования следующего Инкремента.

137) Дополнительно в рамках мероприятий проверки и адаптации по решению Группы ответственных за Систему и Ответственных за реализацию могут проводиться:

1. Работа над технической инфраструктурой и инфраструктурой поддержания рабочих процессов, в том числе:
   - развитие новых инструментов конвейера непрерывной разработки;
   - внедрение и поддержка инструментов автоматизации тестирования;
   - внедрение инструментов управления работами.
2. Обучение (развитие навыков) Рабочей группы, а также проведение исследовательских работ, необходимых для последующего создания и развития Системы.
3. Устранение ошибок, выявленных в ходе тестирования новой функциональности Системы, разработанной в рамках текущего Инкремента.

4.2. Расширенный уровень

4.2.1. Общие рекомендации по организации работ на расширенном уровне

138) Расширенный уровень представляет собой набор ролей, методик и практик, необходимых для создания или развития Систем (представляющих собой крупномасштабные решения)[9] в формате итерационной разработки, требующих ресурсных возможностей больших, чем 150 человек.

139) Увеличение численности разработчиков, как правило, обусловлено увеличением сложности и масштаба разрабатываемых Систем, состоящих из взаимосвязанных функциональных подсистем, каждая из которых имеет самостоятельное значение для предоставления государственных услуг клиентам и реализации государственных функций.

140) Рекомендации по использованию расширенного уровня являются опциональными и используются по решению Ведомства. При принятии решения Ведомство должно учитывать, что необходимость использования расширенного уровня обусловлена следующими обстоятельствами:

1. Мероприятий по организации работ, принятых на базовом уровне, может оказаться недостаточно для координации большого числа Групп разработки. Также может возникнуть необходимость в корректировке принятых на базовом уровне процессов с целью снижения временных и ресурсных издержек. Например, планирование Инкремента с привлечением всей многочисленной Рабочей группы (более 150 человек) может оказаться экономически нецелесообразным.
2. С ростом сложности создаваемых Систем возрастает цена ошибки за некорректные проектные и архитектурные решения. Сбой в работе крупных решений может иметь неприемлемые социальные или экономические последствия. Поэтому требуются дополнительные меры контроля результатов и процессов создания таких Систем, а также проверки их на соответствие отраслевым и нормативным стандартам и предоставления объективных доказательств соответствия требованиям ТЗ.

4.2.2. Ролевая структура

141) Ролевая модель разработки на расширенном уровне *(Рисунок 16)* основана на ролевой модели базового уровня, при этом:

[9] Крупномасштабные системы обеспечивают поддержку нескольких потоков ценностей для потребителей Государственных услуг и имеют совокупную стоимость владения более 100 млн. руб.,

1. Несколько Групп команд разработки объединяются в Пул команд разработки.
2. Роли, установленные для базового уровня, предполагают выполнение задач, аналогичных задачам базового уровня в рамках разработки функциональных подсистем. На уровне разработки функциональных подсистем ролевая структура включает:
   - Ответственных за подсистему;
   - Архитектора подсистемы;
   - Группу команд разработки (подсистемы);
   - Менеджера процессов Группы команд.
3. На расширенном уровне в состав Рабочей группы вводятся дополнительные роли, обеспечивающие создание Системы:
   - Группа ответственных за Систему, включающих в том числе Ответственных за подсистему — должностных лиц Ведомства, отвечающих за предоставление клиентам определенной функциональности Системы;
   - Главный архитектор Системы, которому функционально подчинены архитекторы подсистем;
   - Менеджер процессов пула команд разработки, координирующий деятельность менеджеров процессов Группы команд.

142) **Пул команд разработки** — это организационная конструкция, обеспечивающая создание и развитие Системы на расширенном уровне. Пул команд разработки координирует работы нескольких Групп команд разработки, каждая из которых обеспечивает разработку отдельных функциональных подсистем Системы. Пул команд разработки в общем случае может включать следующие типы команд (см. раздел 2.4):

1. Группы команд разработки, ориентированные на создание ценности и решающие задачи разработки (или адаптации) программного обеспечения подсистем.
2. Группы команд специальных подсистем, обеспечивающие создание подсистем, требующих глубоких специальных навыков и опыта, например, подсистем информационной безопасности, и тем самым снижающие нагрузку на Группы команд разработки ценности.
3. Группы команд платформы, ориентированные на создание и поддержку платформы (группы команд платформы), решающих задачи разработки и поддержки новых технологий и технологических платформ, на базе которых создается Система, и предоставляющих услуги другим командам.

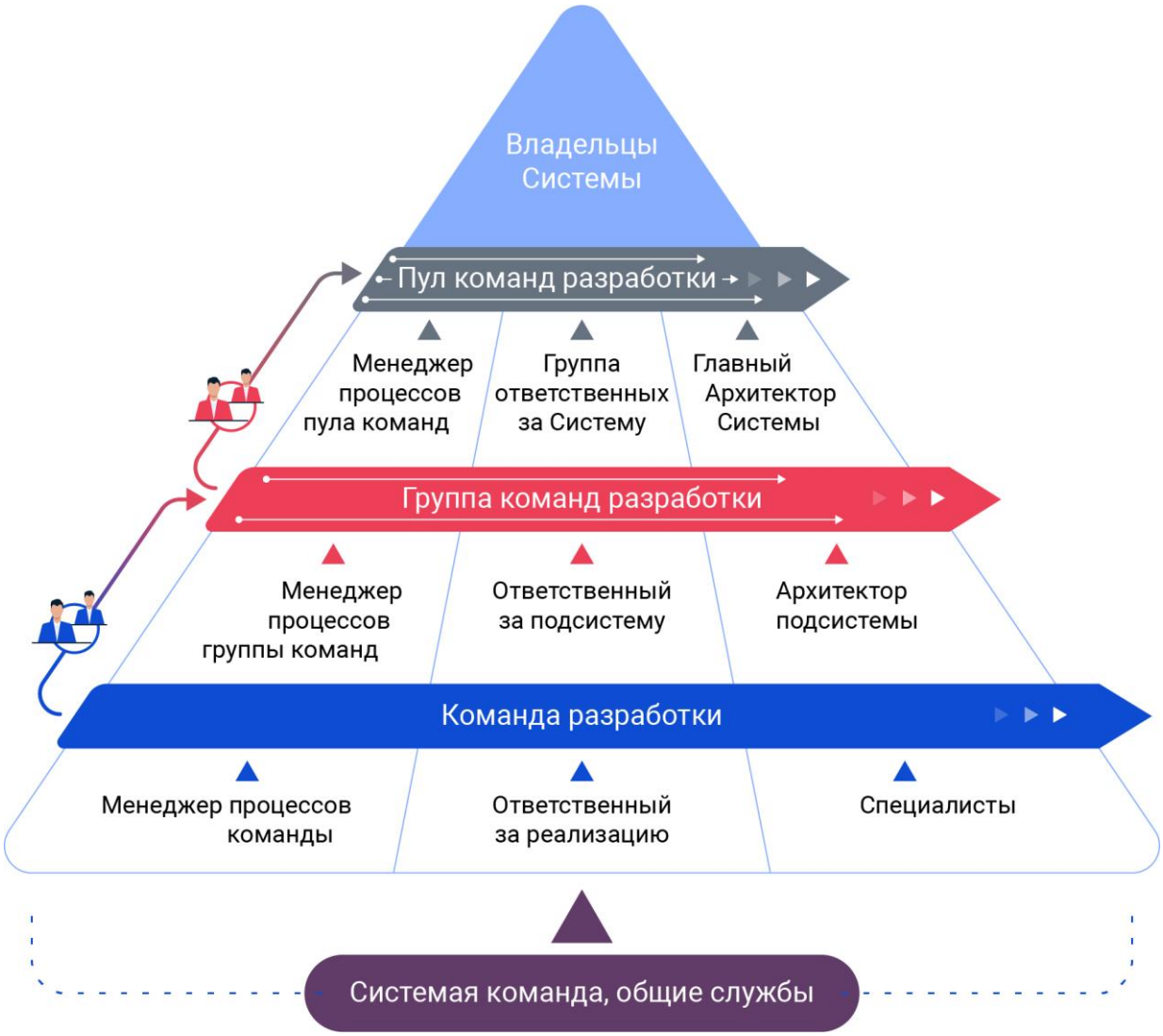

Рисунок 16. Ролевая структура на расширенном уровне

143) **Главный архитектор Системы** определяет общее техническое и архитектурное видение разрабатываемых проектных решений и организует работу команды, включающей Архитекторов подсистем.

144) **Группа ответственных за Систему** отвечает за создание Системы, управляет и координирует работу Ответственных за подсистемы.

145) **Менеджер процессов пула команд** является руководителем, организующим работу Менеджеров процессов Группы команд и отвечающим за организацию и содействие выполнению Пулом команд разработки согласованных рабочих процессов, рекомендуемых настоящим документом.

146) Дополнительно в состав Рабочей группы могут быть включены:

1. **Системная команда** обеспечивает инфраструктуру разработки и развертывания Системы.

2. **Общие службы** предоставляют специальные навыки — например, специалисты по информационной безопасности, финансовые специалисты, юристы.

4.2.3. Особенности выполнения работ на расширенном уровне

147) Типовая схема выполнения работ на расширенном уровне представлена на *Рисунок 17*. Типовая схема выполнения работ на расширенном уровне. По аналогии с Базовым уровнем предполагается, что стадия выполнения работ выполняется серией Инкрементов, каждый из которых включает несколько итераций, реализующих:

а. планирование Инкремента;

б. выполнение Инкремента;

в. проверку результатов Инкремента;

г. корректировку выполняемых работ.

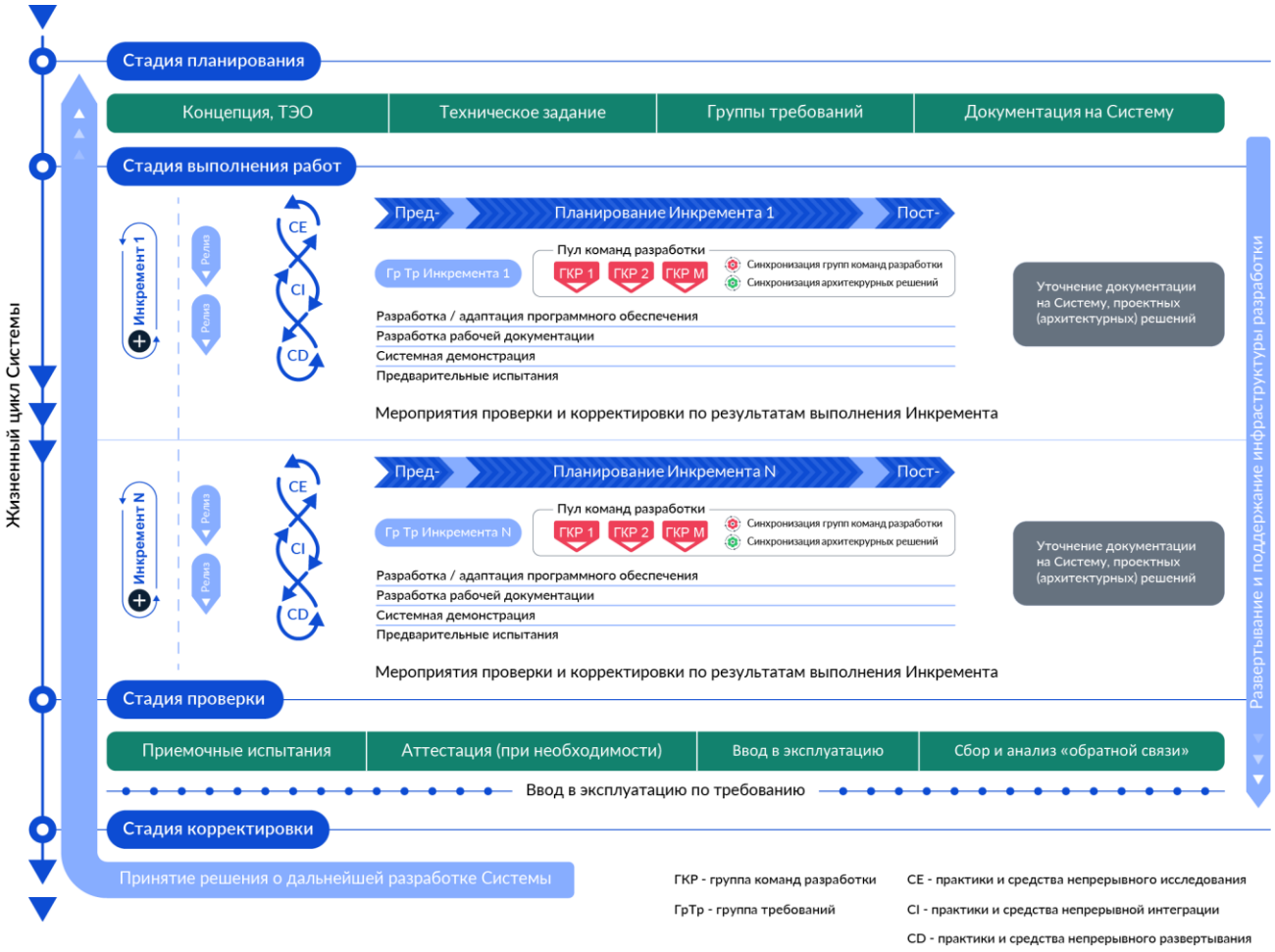

Рисунок 17. Типовая схема выполнения работ на расширенном уровне

4.2.3.1. Особенности планирование Инкремента на расширенном уровне

148) Планирование Инкремента является критической точкой синхронизации Инкрементов для каждой Группы команд разработки в рамках Пула команд. Особенность планирования для нескольких Групп команд разработки обусловлена сложностью организации взаимодействия и сотрудничества в рамках единого мероприятия большого количества людей, а также высокими затратами на проведение очного мероприятия с присутствием всех членов Рабочей группы (как, например, на при планировании на базовом уровне). Для разрешения этой сложности при планировании Инкремента на расширенном уровне рекомендуется два дополнительных мероприятия:

1. Мероприятие **«пред-планирование»** Инкремента используется для координации входных целей, этапов, а также контекста разработки решения для сессий планирования Инкремента для каждой Группы команд разработки.
2. Мероприятие **«пост-планирование»** Инкремент используется для интеграции результатов планирования Группы команд в рамках единого плана Инкремента. В конце мероприятия пост-планирования должно быть достигнуто соглашение о наборе целей Инкремента на уровне Системы, которые должны быть реализованы и продемонстрированы всеми Группами команд разработки к концу Инкремента.

149) За выполнение всех мероприятий пред- и пост- планирования отвечает Менеджер процессов пула команд, который привлекает Менеджеров процессов Групп команд и Менеджеров процессов Команд разработки. О мероприятиях информируются Ответственные за Систему и Владелец Системы.

150) Мероприятия пред-планирования итерации обеспечивают несколько преимуществ:

а. открытое и продуктивное общение и достижение компромиссов лицом к лицу;

б. синхронизация Групп команд разработки в рамках Группы решения и целей Инкремента Решения;

в. выявление зависимостей и содействие координации между Группами команд разработки;

г. обеспечение уточнения архитектуры на уровне Решения;

д. соответствие потребностей (целей) разработки Решения в рамках предстоящего Инкремента возможностям Групп команд разработки.

151) Непосредственное планирование Инкремента происходит синхронно для всех Групп команд разработки в одно и то же время, аналогично тому, как данное мероприятие проводится на базовом уровне *(Раздел 4.1.3.1)* с привлечением Владельца Системы, Группы ответственных за Систему, Ответственных за подсистемы, Команды

разработки, Главного архитектора Системы и Архитекторов подсистем, представителей Групп команд разработки (Команд разработки) и других заинтересованных сторон. При этом при подготовке к планированию Инкремента на расширенном уровне необходимо обеспечить:

а. детальный план проведения планирования Инкремента, включая распределение обязанностей (ролей) в процессе непосредственного планирования, а также время на синхронизацию между Менеджерами процессов Групп команд и Менеджерами Процессов команд разработки в ходе планирования;

б. техническое обеспечение проведения онлайн-мероприятий (если они запланированы), включая работоспособность, наличие достаточного количества лицензий для программного обеспечения онлайн-трансляций, видеоконференций и иных инструментов для обеспечения онлайн-работы.

152) Мероприятия планирования Инкрементов на расширенном уровне обеспечивают согласованность и создают единый план для всех Групп команд разработки, а также способствуют управлению зависимостями между ними. Результатом этих мероприятий является обобщенное формирование целей Инкремента для согласования с заинтересованными сторонами на уровне Пула команд разработки.

4.2.3.2. Особенности выполнения Инкрементов на расширенном уровне

153) Реализация итерационного подхода к разработке на расширенном уровне предполагает использование единой для всех Групп команд разработки частоту Инкрементов и синхронизацию для управления разработкой. При этом следуют учитывать следующие особенности выполнения работ:

1. В рамках Системы, как правило, интегрируются подсистемы, созданные и предоставляемые различными Исполнителями, в условиях специфических нормативных и технологических ограничений;
2. При развитии Системы, объединяющей несколько подсистем, необходимо, с одной стороны, соблюдение баланса между новыми возможностями и технологиями и наследуемой функциональностью, реализованной с использованием разработанных ранее архитектурных решений и технологий, — с другой стороны;
3. Необходимо создание конвейера непрерывной разработки, единого для всех Групп команд разработки, а реализация процессов непрерывного и скоординированного развития функциональности должна обеспечиваться как для отдельных подсистем, так и Системы в целом.
4. Создание и развитие Систем на расширенном уровне предполагает:

- регулярное уточнение проектных решений, в том числе изменение архитектуры решения для обеспечения масштабирования и удобства обслуживания Системы в будущем, а также обеспечение требований информационной безопасности;
- постоянное решение проблем, связанных с соблюдением нормативных требований (прежде всего требований информационной безопасности), касающихся разработки как отдельных подсистем, так и Системы в целом.

154) В ходе создания Системы на расширенном уровне должна обеспечиваться синхронизация работ Групп команд разработки, включающая *(Рисунок 18):*

а. координацию целей и задач на уровне Групп ответственных за подсистемы;

б. постоянную координацию архитектурных решений, используемых различными Группами команд разработки;

в. интеграцию подсистем, создаваемых различными Группами команд разработки.

Рисунок 18. Мероприятия Пула команд по выполнению работ на расширенном уровне

4.2.3.3. Проверка результатов Инкремента на расширенном уровне

155) По аналогии с базовым уровнем на расширенном уровне основным мероприятием для изучения решений и получения объективной оценки и обратной связи является демонстрация Системы.

156) Демонстрация Системы происходит в конце каждого Инкремента. Во время демонстрации представители Пула команд разработки представляют Группе

ответственных за Систему и Владельцу Системы интегрированный результат всех Групп команд разработки.

157) Каждая демонстрация Системы должна обеспечивать исправление всех выявленных ошибок и несоответствий техническому заданию, а также обеспечивать снижение рисков разработки Системы за счет получения обратной связи.

158) Пул команд разработки должен обеспечивать возможность развернуть Систему или отдельные подсистемы в контуре эксплуатации в любое время, когда это потребуется Ведомству по решению Владельца Системы.

159) Мероприятия по вводу в эксплуатацию на расширенном уровне аналогичны мероприятиям на базовом уровне.

4.2.3.4. Корректировка выполняемых работ на расширенном уровне

160) Мероприятие «Проверка и адаптация» на расширенном уровне для Пула команд разработки осуществляется аналогично проведению проверки и адаптации в контексте одной Группы команд на базовом уровне *(Раздел 4.1.3.4)*.

161) С целью сокращения временных и материальных затрат на проведение мероприятия «Проверка и адаптация» рекомендуется в состав участников мероприятий включать ключевых представителей Групп команд разработки, обладающих всеми необходимыми полномочиями для решения проблем.

Мероприятие «Проверка и адаптация» организует Менеджер процессов пула команд с привлечением Группы ответственных за Систему.

162) По аналогии с базовым уровнем в конце мероприятия «Проверка и адаптация» проводится ретроспективный семинар для улучшения процессов создания и развития Системы с привлечением представителей Групп команд разработки, Групп ответственных за Систему, ответственных за подсистемы, Менеджера процессов пула команд, Менеджеров процессов групп команд.

Ретроспективный семинар организует и проводит Менеджер процессов пула команд с привлечением Менеджеров процессов групп команд.

5. СТАДИЯ ПРОВЕРКИ ВЫПОЛНЕННЫХ РАБОТ

163) Стадия проверки выполненных работ проводится Группой ответственных за Систему с целью определения соответствия результатов выполненных работ по созданию Системы требованиям Технического здания, а также планирования и ввода Системы в эксплуатацию, мониторинга показателей функционирования Системы и сбора обратной связи от клиентов.

164) Реализация функциональности Системы с использованием итерационного подхода предполагает возможность ввода в эксплуатацию (после проведения испытаний) отдельных релизов Системы, включающих функциональность по результатам реализации отдельных групп требований. Такой подход обеспечивает оперативное предоставление Системы клиентам, получение обратной связи от клиентов и формирование предложений по развитию Системы.

165) Мероприятиям по проверке предшествуют пусконаладочные работы и предварительные испытания Системы, выполняемые на стадии «Выполнение» *(см. пункт 118).*

166) В рамках стадии проверки проводятся следующие виды испытаний в соответствии с Постановлением № 676:

а. опытная эксплуатация Системы или ее части;

б. приемочные испытания Системы или ее части.

167) Опытная эксплуатация Системы или ее части проводится начиная с завершения реализации первой Группы требований и до завершения разработки (адаптации) программного обеспечения и включает:

а. разработку программы и методики опытной эксплуатации;

б. проведение опытной эксплуатации Системы в соответствии с программой и методикой опытной эксплуатации;

в. оформление акта о завершении опытной эксплуатации, включающего перечень недостатков, которые необходимо устранить до начала эксплуатации Системы;

г. проведение дополнительной итерации по доработке программного обеспечения Системы и дополнительной настройки технических средств в случае обнаружения недостатков, выявленных при опытной эксплуатации Системы.

168) Приемочные испытания Системы или ее части включают:

а. разработку программы и методики приемочных испытаний;

б. испытания Системы на соответствие техническому заданию на ее создание в соответствии с программой и методикой приемочных испытаний;

в. анализ результатов устранения недостатков, указанных в акте о завершении опытной эксплуатации;

г. проведение итерации устранения замечаний, выявленных в ходе приемочных испытаний.

169) Приемочные испытания системы завершаются на основании результатов проверки Системы, а также на основании протоколов испытаний частей (отдельных компонентов) Системы в рамках выполнения Инкрементов *(см. пп. 124) - 126)*

170) Ввод Системы или ее части в эксплуатацию включает следующие мероприятия, выполняемые в соответствии с требованиями Постановления № 676:

а. подготовка правового акта Ведомства о вводе Системы в эксплуатацию, определяющего перечень мероприятий по обеспечению ввода Системы в эксплуатацию и устанавливающего срок начала эксплуатации;

б. разработка и утверждение организационно-распорядительных документов, определяющих мероприятия по защите информации в ходе эксплуатации Системы, разработка которых предусмотрена нормативными правовыми актами и методическими документами федерального органа исполнительной власти в области обеспечения безопасности и федерального органа исполнительной власти, уполномоченного в области противодействия техническим разведкам и технической защиты информации, а также национальными стандартами в области защиты информации;

в. аттестация Системы по требованиям защиты информации, в результате которых в установленных законодательством Российской Федерации случаях подтверждается соответствие защиты информации, содержащейся в Системе, требованиям, предусмотренным законодательством Российской Федерации об информации, информационных технологиях и о защите информации;

г. оформление прав на использование компонентов Системы, являющихся объектами интеллектуальной собственности;

д. ввод Системы или ее части в эксплуатацию.

171) Основанием для начала эксплуатации Системы (очереди Системы) является наступление срока, установленного правовым актом Ведомства о вводе Системы в эксплуатацию.

172) В случаях, указанных в Постановлении №676 ввод в эксплуатацию в составе эксплуатируемой системы Релиза Системы по результатам реализации группы

требований к Системе осуществляется на следующий рабочий день с даты утверждения протокола проверки (испытаний) релиза без необходимости принятия правового акта, предусмотренного *пунктом 170) а.*

173) Эксплуатация не допускается в случаях, указанных в Федеральном законе "Об информации, информационных технологиях и о защите информации", а также в Постановлении №676.

174) Эксплуатация Системы организуется Ведомством с привлечением собственных подразделений и/ли внешних организаций (Исполнителей) в соответствии с требованиями, указанными в Государственном контракте, Концессионном соглашении и в соответствии с внутренними регламентами Ведомства, определяющим порядок эксплуатации Системы и требования к уровню сервиса. При этом Ведомство и/или внешний Исполнитель в ходе эксплуатации Системы руководствуется рабочей документацией, согласованной и утверждённой Ведомством и, при необходимости, иными уполномоченными организациями.

175) Ведомство обеспечивает поддержку безопасности Системы в соответствии с аттестатом соответствия путем реализации требований по защите информации в ходе эксплуатации Системы и проведения периодического контроля уровня защиты информации, результаты которого оформляются протоколами и отражаются в техническом паспорте на объект информатизации (Систему).

176) В общем случае, эксплуатация Системы должна включать прием и регистрацию обращений клиентов, консультации клиентов, а также исправление ошибок и проведение доработок ранее реализованных функциональных возможностей. При этом в случае, если в рамках эксплуатации проводится дальнейшее развитие Системы рекомендуется привлечение к исправлению ошибок текущего исполнителя (*см. пункты 121) - 122)*, что позволит избежать существенных издержек на организацию взаимодействия по сравнению с случаем, если внесение исправлений и разработка новой функциональности будут выполняться разными организациями (при этом следует учитывать, что Исполнитель после закрытия Государственного контракта несет гарантийные обязательства). Как вариант, с целью обеспечения преемственности экспертизы в случае привлечения внешних Исполнителей рекомендуется рассмотреть вопрос о заключении с ними Контрактов жизненного цикла.

177) После ввода Системы или ее части в эксплуатацию должен быть организован мониторинг показателей качества оказания государственных услуг, государственных функций, включая контрольно-надзорную деятельность с использованием Системы (далее — мониторинг), в соответствии со Стандартом мониторинга оказания сервисов (утвержден Протоколом заседания президиума Правительственной комиссии по

цифровому развитию, использованию информационных технологий для улучшения качества жизни и условий ведения предпринимательской деятельности от 29 декабря 2021 г. № 50).

178) По результатам мониторинга выявляются предложения по развитию Системы, а также замечания к функционированию Системы, устраняемые Исполнителем, например, в рамках гарантийного срока в соответствии с законодательством Российской Федерации.

179) В рамках эксплуатации Системы осуществляется сбор обратной связи от клиентов, и проводится анализ потребностей клиентов с целью разработки предложений по развитию Системы.

180) По результатам мероприятий по проверке выполненных работ Ведомство принимает решение о дальнейшем развитии Системы или о выводе Системы из эксплуатации, при этом учитываются следующие факторы:

а. финансово-экономическая целесообразность дальнейшей эксплуатации Системы или ее развития;

б. снижение эксплуатационных характеристик (устаревания) используемых технических средств и программного обеспечения;

в. изменение правового регулирования вопросов оказания государственных услуг и выполнения государственных функций, а также наличие иных изменений, препятствующих эксплуатации Системы.

181) Поскольку отказ Системы может иметь неприемлемые социальные последствия и/или экономические издержки, Система должна подвергаться регулярному надзору со стороны контролирующих и регулирующих органов и удовлетворять требованиям нормативных документов, в том числе требованиям информационной безопасности. Владельцы системы могут организовывать проведение аудита Системы на соответствие нормативным требованиям с привлечением уполномоченных Ведомств (например, ФСТЭК, ФСБ).

182) Выполнение мероприятий по выводу Системы из эксплуатации осуществляется в соответствии с определяемыми Постановлением № 676 требованиями к порядку вывода Системы из эксплуатации и дальнейшего хранения содержащейся в ее базах данных информации.

6. СТАДИЯ КОРРЕКТИРОВКИ

183) В случае принятия решения о продолжении создания и развития Системы Ведомство проводит стадию корректировки. Стадия корректировки включает:

а. разработку предложений по развитию Системы на основе анализа результатов мониторинга и сбора информации обратной связи от клиентов;

б. определение требований к развитию Системы с учетом изменения потребностей клиентов и контекста использования Системы;

в. оценка Ведомством объемов финансирования, необходимого для развития Системы, внесение, при необходимости, изменений в ВПЦТ;

г. переход к очередному циклу планирования реализации мероприятий по развитию Системы.

184) Дополнительно на стадии корректировки рекомендуется проводить анализ проблем, с которыми столкнулась Рабочая группа в рамках выполнения работ по созданию Системы (Очереди системы) с целью выработки предложений по корректировке производственного процесса в рамках дальнейшего развития Системы.

ПРИЛОЖЕНИЕ № 1

к Методическим рекомендациям
по организации производственного процесса разработки
государственных информационных систем с учетом
применения итерационного подхода к разработке

МЕТОДЫ И ПРАКТИКИ, ИСПОЛЬЗУЕМЫЕ ДЛЯ СОЗДАНИЯ И РАЗВИТИЯ ГОСУДАРСТВЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Оглавление

1. ПЕРЕЧЕНЬ ОСНОВНЫХ МЕТОДОВ[1] И ПРАКТИК[2]

Методические рекомендации по разработке государственных информационных систем используют следующие методы и практики *(Рисунок 19)*, получившие широкое распространение при создании и развитии цифровых сервисов и продуктов[3]:

1. Системный подход;
2. Методы итерационной разработки (Скрам, Канбан, Экстремальное программирование);
3. Методики масштабирования итерационной разработки;
4. Практики DevSecOps и бережливого производства (Встроенное качество, Дизайн-мышление, Конвейер непрерывной доставки).

Рисунок 19. Основные методы и практики создания продукта

[1] Метод — систематизированная совокупность шагов, действий, которые необходимо предпринять, чтобы решить определенную задачу или достичь определенной цели (ГОСТ 56407 – 2015. Бережливое производство. Основные положения).
[2] Практика – накопленный опыт, испытанные приёмы и навыки в какой-либо области деятельности.
[3] Продукт - изделие и/или услуга, которые в настоящее время и в будущем будут предлагаться производителем и обладают определенной ценностью для существующих или потенциальных клиентов (ГОСТ Р 58537- 2019. Управление продукцией. Основные положения).

Указанные методы и практики в совокупности направлены на обеспечение **потока создания ценности для клиента**, под которым в общем случае понимаются все действия, как создающие, так и не создающие непосредственно ценность, которые позволяют продукции пройти все процессы — от разработки концепции до запуска в производство и от принятия заказа до доставки потребителю. Поток создания ценности предполагает **организацию производственного процесса вокруг создания ценности для клиента** и используется как интегральное понятие, включающее в себя материальные потоки (сырье, материалы, комплектующие, детали и сборочные единицы, готовую продукцию), информационные и финансовые потоки, направленные на создание и доставку готовой продукции потребителю в установленное время, в установленном месте, с установленной стоимостью, с последующим ее обслуживанием в процессе эксплуатации и вывода из эксплуатации (ГОСТ Р 56020-2020).

2. СИСТЕМНЫЙ ПОДХОД К ОРГАНИЗАЦИИ РАБОТ

Теоретическим обоснованием используемых методов и практик является Системный подход [12, 23, 46, 48, 56, 57][4] — методология изучения (рассмотрения) системы как целостного структурированного комплекса взаимосвязанных и взаимодействующих элементов для достижения поставленных целей в динамично изменяющемся окружении.

Создание Системы является сложной инженерно-практической и организационной задачей, при этом сложность характерна не только применительно к Системе, но и к организации работ по созданию (развитию) Системы, а также к используемой при этом методологии. В теории рассматриваются два подхода к определению понятия сложности — структурный и управленческий. С точки зрения структурного подхода [56, 57], сложность рассматривается как большое число объектов, которые взаимодействуют множеством способов с различной интенсивностью. С точки зрения управленческого подхода [48], сложной является система, в модели которой не хватает информации для эффективного управления (в контексте процесса достижения цели системы). При этом системная методология учитывает роль человеческого фактора в сложных социальных, организационных и технических системах и примиряет многочисленные и противоречивые цели у людей, чьи интересы затрагивает поведение системы.

В рамках настоящих Методических рекомендаций в рамках системного подхода использованы следующие направления:

1. **Адаптивность к изменениям**, предполагающая использование механизмов разработки и функционирования систем, позволяющих достигать целей в условиях изменения контекста использования системы и потребностей клиентов, для которых создается система (является, наряду с принципом обратной связи, теоретическим обоснованием итерационного (гибкого) подхода к разработке) [46, 58, 62].
2. **Использование обратной связи** о степени достижения цели разработки для обеспечения устойчивости производственного процесса [49, 50] (является теоретическим обоснованием необходимости корректировки проектных (архитектурных) решений в процессе разработки).

   Отсутствие обратной связи между взаимосвязанными и взаимодействующими элементами системы, сигнализирующей о текущем состоянии (или достигнутом результате) системы, организация эффективного управления системой проблематично, поскольку на основании этой информации управляющее

[4] [x] см. список использованных источников

воздействие может быть должным образом скорректировано. В общем случае механикам обратная связь[5] обеспечивает:

- восстановление нормальной работы системы, нарушенной внешними и внутренними факторами, т.е. обеспечение способности систем к саморегулированию и самоорганизации (адаптации).
- осуществление управления системой в условиях неполной информации о возмущающих воздействиях, носящих, как правило, случайный, априорно неизвестный характер.

3. **Интеграция элементов и обеспечение целостности,** предполагающие учет принципов системности и эмерджентности [46, 48] при разработке систем (является теоретическим обоснованием подходов к формированию моделей требований к системе, а также проектирования и определения структуры систем).

   Эмерджентность — свойство системы обладать признаками, отличными от признаков подсистем ее составляющих. Главная особенность систем состоит в том, что свойства целого не сводятся к совокупности свойств его частей, что система обладает принципиально иным качеством, которое существует, пока существует целое, являясь, таким образом, проявлением внутренней сущности системы (системообразующим фактором). При этом доказано (Эшби, Конструкция мозга, 1962), что у системы тем больше возможностей, чем сильнее согласованность поведения её частей (является теоретическим обоснованием необходимости регулярного проведения мероприятий синхронизации и «выравнивания» команд в процессе выполнения работ).

4. **Декомпозиция в виде иерархий** как представление систем в виде единого целого на одном уровне и как частей (подсистем) системы другого, более высокого уровня (является теоретическим обоснованием построения иерархии работ по созданию и развитию Систем и способов масштабирования команд разработки).

   Иерархии являются абстракцией структуры систем, предназначенных для изучения взаимодействия ее компонентов и их воздействия на систему в целом.

[5] Различают положительную и отрицательную обратную связь. Отрицательная обратная связь изменяет входной сигнал таким образом, чтобы противодействовать изменению выходного сигнала. Это делает систему более устойчивой к случайному изменению параметров. Положительная обратная связь, наоборот, усиливает изменение выходного сигнала. Положительная обратная связь ускоряет реакцию системы на изменение входного сигнала, поэтому её умышленно используют в технике в ситуациях, когда требуется ускорение реакции на изменение внешних параметров. В то же время положительная обратная связь может привести к неустойчивости системы.

Системный подход в целом предполагает описание систем и проблемы их построения в терминах взаимосвязанной иерархии[6].

5. **Координация и централизация** [46], предполагающие организацию управления в **сложных иерархических системах** через гибкое сочетание принципов автономности нижестоящих элементов иерархии и централизованного формирования координирующих механизмов для обеспечения стабильности производственных процессов (является теоретическим обоснованием практик организации взаимодействия и координации крупномасштабных команд разработки).

[6] Системный подход сложился как теория, заимствовавшая врожденные способности человека декомпозировать сложные события на составные элементы, одновременно анализируя взаимосвязи между элементами системы.

3. МЕТОДЫ ИТЕРАЦИОННОЙ РАЗРАБОТКИ

Получившая широкое распространение и применяемая в настоящее время классическая водопадная модель создания систем основана на последовательном выполнении этапов проекта от планирования до сдачи Системы в эксплуатацию [10]. При этом потребные привлекаемые ресурсы (бюджет) и сроки выполнения этапов проекта определяются в начале проекта, исходя из оценки реализации заданных требований. Ошибки в оценках бюджета и сроков порождают риски невыполнения проекта в рассчитанных рамках либо снижения качества выполненных работ.

Альтернативой классическому подходу стали методы итерационной разработки [9, 28, 53, 55, 61]. Первый ставший популярным гибкий метод — экстремальное программирование, основы которого подробно изложены в книге «Extreme Programming Explained» Кента Бека (1999 год) [31, 64]. Следующим гибким методом, получившим распространение, стал Скрам, описанный Кеном Шабером и Майком Бидлом в книге «Agile Software Development with SCRUM» (2002 год) [81].

В дальнейшем получило распространение введения идей бережливого производства в технологические разработки программного обеспечения Канбан как метода эволюционных и пошаговых изменений, использующего «вытягивающую» систему работ, визуализацию и другие инструменты. Канбан позволяет достичь оптимизации процесса с минимальным сопротивлением разработчиков к изменениям и при этом сохранить оптимальный темп для всех вовлеченных в работу сотрудников [27].

Конечным результатом применения методов итерационной разработки является устранение рисков низкого качества разработки. В итерационной разработке фиксируются сроки и ресурсы (команда), а изменяется состав требований, которые должны быть реализованы итерационно для последовательного достижения целей разработки *(Рисунок 20)* [74].

В конце каждой итерации должен достигаться конечный результат — версия цифрового продукта, в рамках которой реализован запланированный набор функций. Конечный результат предоставляется заинтересованным сторонам, возможно, конечным пользователям для получения обратной связи и принятия решения о завершении разработки или определения направлений развития продукта.

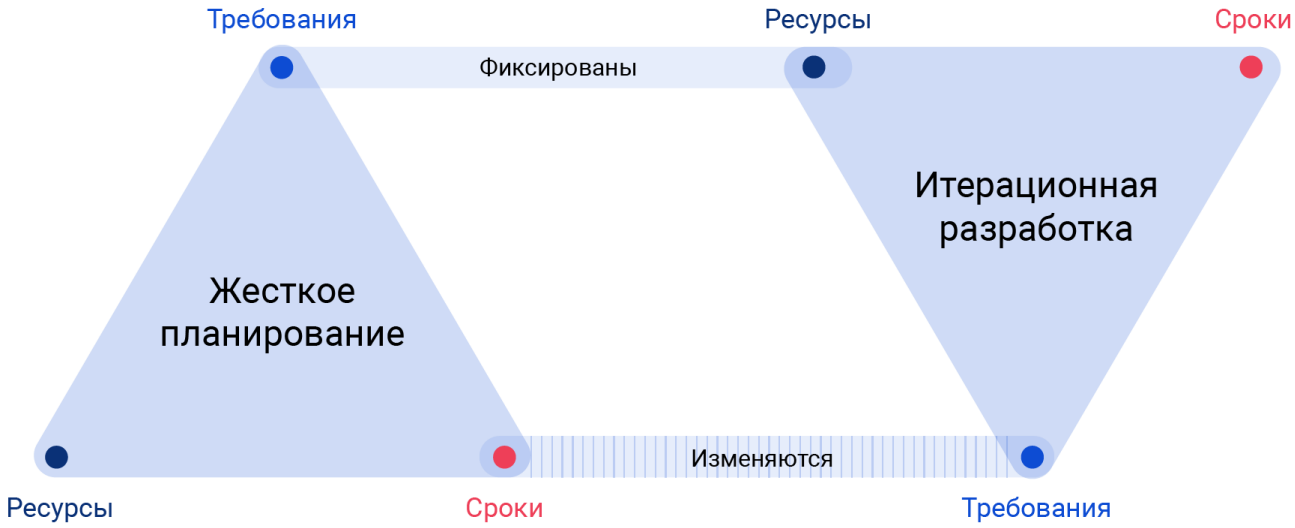

Рисунок 20. Смена парадигмы разработки

Настоящие Методические рекомендации предполагают, что Команды разработки в зависимости от их типа [79] используют один из методов итерационной разработки — Скрам, Экстремальное программирование, систему Канбан и/или их комбинации [63]. В частности, большинству Команд разработки рекомендуется использовать Скрам в силу простоты и универсальности этого метода. Для других команд, таких как например, Системные команды предпочтительно применять систему Канбан в качестве основного метода в силу быстро меняющихся приоритетов их работы и отсутствия необходимости регулярного планирования итераций.

3.1. Методология Скрам

Методология[7] Скрам [21, 30, 33, 51, 81] — это способ организации рабочего процесса разработки продукта, при котором задачи разработки разбиваются на мелкие блоки (подзадачи) и передаются кросс-функциональным автономным Командам разработки, которые работают методом итераций, то есть короткими непрерывными циклами, в течение которых команда проектирует, кодирует, тестирует и анализирует результаты реализации требований, в том числе — на соответствие потребности клиента. В Скрам для обозначения итерации используется термин «спринт».

Классический Скрам состоит из ролей, артефактов и процессов.

Роли Скрам:

a. кросс-функциональная команда;

[7] Методология - набор взаимосвязанных **методов и методик** и их практического применения (практик), оптимизированных для решения конкретных задач

б. ответственный за реализацию (владелец продукта);

в. менеджер процессов команды (скрам-мастер);

Основные артефакты Скрам:

а. бэклог продукта — источник требований и их изменений на протяжении всего процесса создания продукта;

б. бэклог спринта;

в. добавленная ценность продукта.

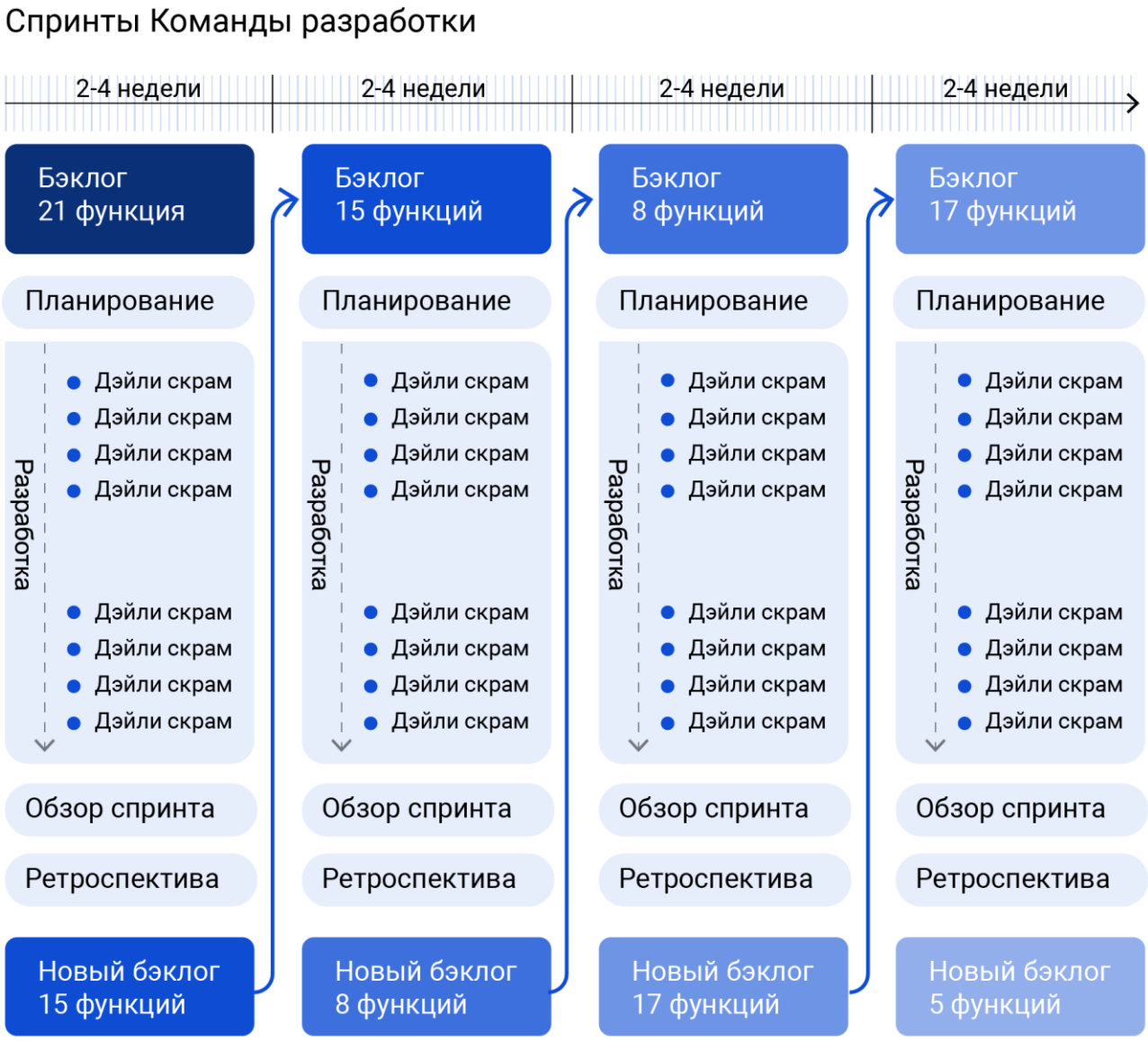

Рисунок 21. Артефакты и процессы Скрам

Рабочий процесс в Скрам проходит по определенной схеме, которая определяется **серией ограниченных по времени событий,** которые всегда происходят в одинаковом порядке *(Рисунок 21)* и включают:

а. планирование спринта;

б. ежедневные совещания на ногах (или «Дэйли скрам»);

в. подведение итогов спринта;

г. обзор итогов спринта, включая демонстрацию версии;

д. ретроспективу спринта;

е. уточнение бэклога продукта.

3.1.1 Планирование итерации (спринта)

Итерация начинается с планирования — события продолжительностью до восьми часов, в котором владелец продукта формулирует **цель спринта** (одно-два предложения, описывающих, что должно быть сделано в ходе спринта) и представляет требования для планирования в виде **пользовательских историй**[8]. Затем команда для уяснения работы, которая будет выполнена в предстоящей итерации, выполняет следующие действия:

а. проводит обсуждения, а при необходимости уточнение пользовательских историй;

б. разбивает пользовательские истории спринта на **задачи**[9], занимающие один день или менее на разработку (это называется **декомпозицией**);

в. определяет для каждой задачи ответственного исполнителя из состава команды;

г. переводит наборы функций, которые команда может предоставить в предстоящей итерации, основываясь на их известной скорости, в бэклог спринта, в котором перечисляется все, что будет реализовано командой за время спринта;

д. фиксирует план спринта, включающий в том числе цели спринта и пользовательские истории.

Планирование пользовательских историй рекомендуется выполнять с использованием подхода «вертикальных срезов», предполагающего в ходе разработки внесение изменений в каждый архитектурный слой продукта. При этом реализация **функциональных требований** дополняется реализацией обеспечивающих **нефункциональных требований** *(Рисунок 22).*

[8] Пользовательские истории (англ. User Story) — способ описания требований к разрабатываемому продукту, сформулированных как одно или более предложений на повседневном или деловом языке пользователя.

[9] Многие команды делят истории на задачи, оценивая их в часах, чтобы лучше улучшить свое понимание предстоящей работы.

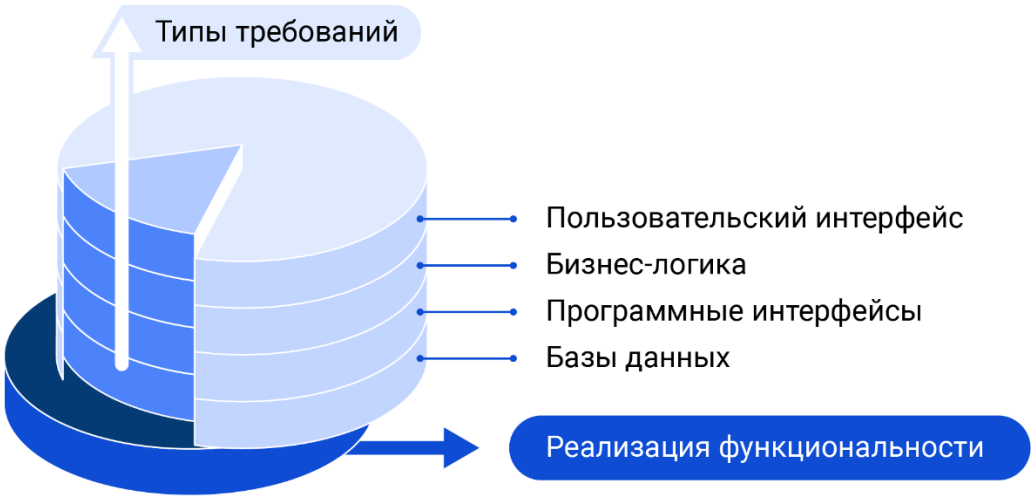

Рисунок 22. Использование вертикальных срезов включает работу во всех архитектурных слоях для реализации функций

3.1.2 Пользовательские истории

Пользовательская история [24, 43] — это наименьшая единица функциональности в методиках гибкой разработки. Пользовательская история представляет собой описание функций простыми, общими словами, без технической специфики, составленное с точки зрения конечного пользователя. Она пишется с целью разъяснить, как именно функциональная возможность принесет ценность пользователю. В Скрам пользовательские истории добавляют в спринты и отслеживают на диаграммах сгорания в течение спринта.

3.1.3 Визуализация работы

Визуализация работы [65] обеспечивает **прозрачность** создания продукта (системы). Наиболее употребимыми средствами визуализации работы в рамках методологии Скрам являются информационные доски и диаграммы сгорания.

3.1.3.1 Информационные доски

Команды используют «Информационные доски» для понимания и отслеживания прогресса во время выполнения итерации. Информационная доска представляет распределение специалистов команды по задачам, визуализирует реализуемые функции и прогресс их реализации на протяжении всей итерации, как показано на рисунке ниже *(Рисунок 23).*

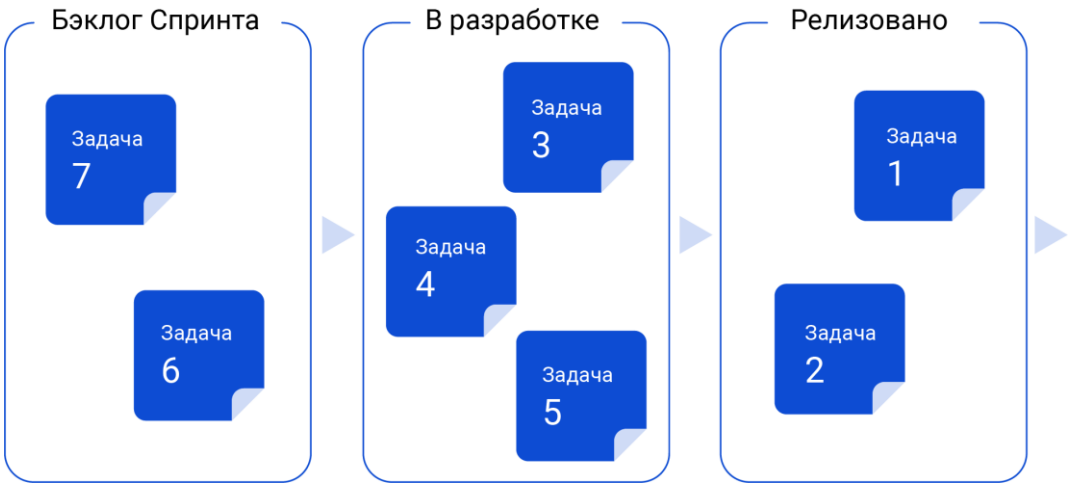

Рисунок 23. Пример информационной доски команды

Во время выполнения работы команда создает и тестирует одну-две функции каждые несколько дней. При этом рекомендуется следить за количеством задач, находящихся одновременно в разработке или тестировании, чтобы постоянно балансировать объем работы с их доступными ресурсами команды и увеличить ее пропускную способность.

Многие команды интегрируют лучшие практики Скрам и Канбан, чтобы облегчить поток работы через итерации. В этом случае распределение задач превращается в более структурированную доску Канбан.

3.1.3.2 Диаграммы сгорания

Диаграмма сгорания показывает количество оставшейся работы в начале каждой итерации. Она служит эффективным визуальным индикатором быстроты продвижения команды к ее цели. Пример диаграммы сгорания представлен на рисунке ниже *(Рисунок 24)*.

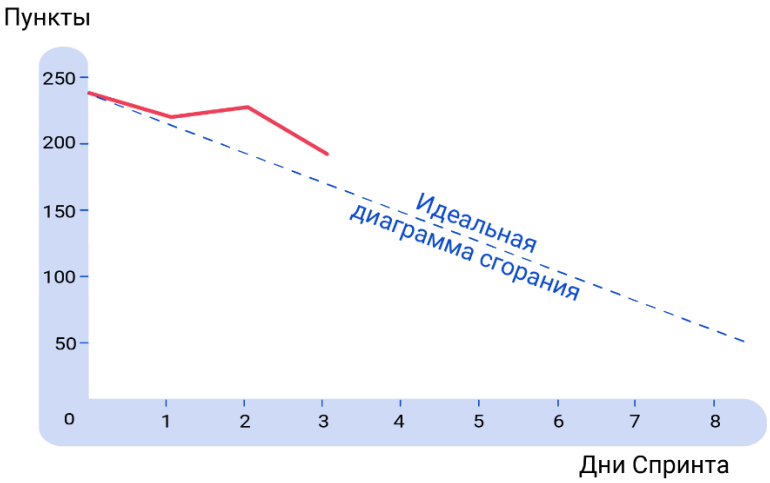

Рисунок 24. Диаграмма сгорания

На вертикальной оси откладывается количество оставшихся пунктов (пользовательских историй) в Бэклоге. На горизонтальной оси откладываются дни Спринта. Данные диаграммы необходимо ежедневно обновлять, чтобы в реальном времени показывать прогресс и издержки в работе над Спринтом. При этом следует учитывать, что диаграмма «сгорает» только по реализации полной пользовательской истории.

Рекомендуется, чтобы оставшаяся работа уменьшалась более-менее равномерно по ходу Спринта (см. идеальную прямую на рисунке). Это ведет к уменьшению числа параллельных задач, стимулирует команду помогать друг другу ради скорейшего перехода задач в «Готово» и в итоге ведет к повышению эффективности Команды.

При таком представлении одна диаграмма просто и ясно показывает два самых важных показателя, которые можно использовать для отслеживания проекта: объем оставшейся работы и чистый темп продвижения команды, не зависящий от изменений объема Бэклога.

3.1.4 Ежедневные совещания на ногах

Каждый день команда проводит официальное мероприятие — ежедневное совещание на ногах («Дэйли скрам») — чтобы понять текущий статус работ, оценить, насколько команды приблизились к цели спринта, проанализировать проблемы и получить помощь от других членов команды. Во время этого мероприятия каждый член команды описывает, что он сделал вчера для достижения целей Спринта, над чем он собирается работать сегодня и любые блокирующие обстоятельства (проблемы), с которыми он сталкивается при достижении целей итерации.

За организацию и проведение ежедневного совещания на ногах отвечает Ответственный за процессы команды (Скрам-мастер). Он обеспечивает его регламент и полноту. Совещание на ногах должно занимать не более **15 минут** и завершаться распределением текущих задач.

После проведения совещания на ногах командное общение не заканчивается, так как члены команды взаимодействуют непрерывно на протяжении всей итерации. В интересах облегчения такого общения необходимо, чтобы команда была размещена в одном месте, если это возможно.

3.1.5 Обзор итогов Спринта

В конце каждого Спринта команда (ориентировочно в течение 1 дня) проводит обзор итерации и ретроспективу спринта. Во время обзора команда демонстрирует каждую реализованную функцию (выполненную историю), обращая особое внимание на увеличение ценности для пользователей. Обзор спринта не является официальным докладом о положении дел, скорее, это обзор ощутимых результатов итерации.

Основная задача проведения обзора Спринта заключается в получении обратной связи от заинтересованных сторон по результатам разработки запланированного Бэклога Спринта проведенной итерации *(Рисунок 25)*.

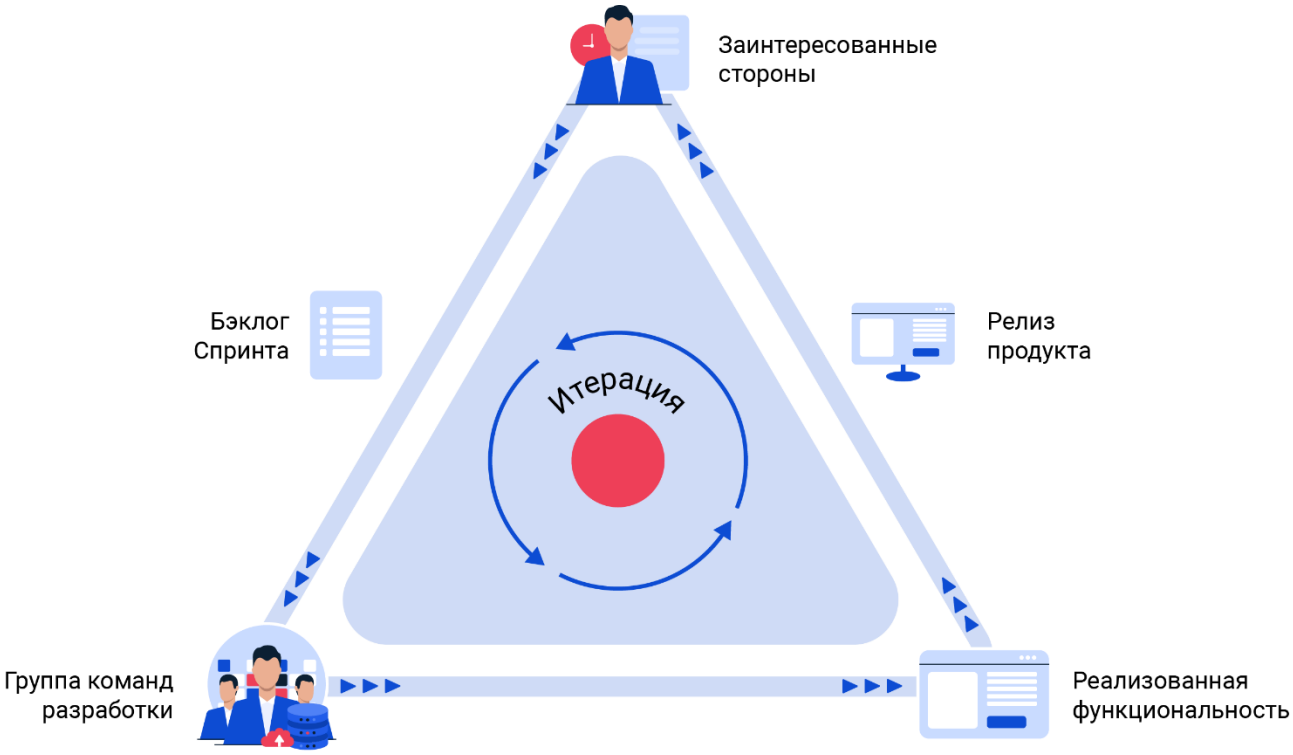

Рисунок 25. Получение обратной связи в итерации

3.1.6 Проведение ретроспективы Спринта

Команда проводит короткую ретроспективу традиционно после обзора спринта — спустя небольшое количество времени, чтобы получить обратную связь по результатам демонстрации.

Организует ретроспективу Ответственный за процессы команды (скрам-мастер) с привлечением Ответственного за реализацию (Владельца продукта).

Обычно **ретроспектива занимает от 30 минут до 3 часов** (в зависимости от продолжительности Спринта). Традиционно на первом этапе ретроспективы проводится сбор данных, который заключается в ответах каждого из членов команды на вопросы:

1. Что было сделано хорошо?
2. Что можно улучшить?
3. Какие улучшения будем делать?

При этом количество улучшений, которые команда берет в реализацию, не должно превышать 2-3, и команда должна обязательно в том или ином виде составить план улучшений для контроля их исполнения.

На втором этапе желательно обсудить следующие темы:

а. скорость команды и ее изменение по сравнению с предыдущими спринтами;

б. качество работ, в том числе нереализованные истории пользователей и причины опоздания, дефекты и их причины;

в. качество процессов команды (нарушения, отступления).

3.1.7 Уточнение Бэклога продукта

Уточнение Бэклога Продукта проводится не для пересмотра того, что уже взято в работу в текущем Спринте, а для будущих Спринтов. Хорошей практикой считается иметь в Бэклоге продукта детально проработанные элементы как минимум на два Спринта вперёд. В этом случае Планирование Спринта существенно упрощается, поскольку Ответственный за реализацию (Владелец продукта) и Команда начинают планирование с понятным, прошедшим этап анализа и корректно оцененным набором пользовательских историй. Если же уточнение Бэклога не было проведено (или было проведено недостаточно хорошо), Планирование Спринта может занять много времени, вызвать большое количество вопросов и потребовать неоднократных уточнений в ходе планирования.

3.2. Практика экстремального программирования

Экстремальное программирование (XP) — одна из гибких методологий разработки программного обеспечения [31, 68].

Ключевые особенности XP включают следующее:

1. Команда из 7...9 программистов работает в одном месте с представителем заказчика;
2. Разработка происходит в виде частых сборок или итераций, обеспечивающих дополнительную функциональность, которые могут быть или не быть выпущенными;
3. Требования задаются в виде пользовательских историй, каждая из которых представляет собой фрагмент новой функциональности, необходимой пользователю;

4. Программисты работают в парах, следуют строгим стандартам кодирования и проводят собственное модульное тестирование, что обеспечивает высокое качество кода;
5. Представитель Заказчика участвует в приемочных испытаниях;
6. Требования, архитектура и проектные спецификации появляются и изменяются в ходе проекта.

Двенадцать основных приёмов экстремального программирования могут быть объединены в четыре группы.

1. Короткий цикл обратной связи:
   - Разработка через тестирование;
   - Игра в планирование;
   - Заказчик всегда рядом;
   - Парное программирование;
2. Непрерывный, а не пакетный процесс:
   - Непрерывная интеграция;
   - Рефакторинг;
   - Частые небольшие релизы;
3. Понимание, разделяемое всеми:
   - Простота проектирования;
   - Метафора системы (аналог архитектуры) — даёт команде представление о том, каким образом система работает в настоящее время, в каких местах добавляются новые компоненты и как они должны взаимодействовать с существующими компонентами;
   - Коллективное владение кодом или выбранными шаблонами проектирования;
   - Стандарт оформления кода;
4. Социальная защищённость программиста, в том числе 40-часовая рабочая неделя.

Важное достоинство экстремального программирования состоит в том, что оно позволяет очень быстро откликаться на изменения, поскольку заказчик каждый день, а то и каждый час видит, как создается продукт и что получается. Внимание к технической дисциплине гарантирует высокое качество кода. Многие технические приемы XP (включая непрерывную интеграцию, непрерывное тестирование, рефакторинг и даже парное программирование) вошли в общую практику и

используются командами, которые не применяют XP, в т. ч. и такими, которые до сих пор работают по старинке.

Серьезная проблема экстремального программирования заключается в том, что сложно заранее предсказать эффективность команды и срок готовности продукта. К тому же, как оказалось, методика с большим трудом масштабируется на крупные проекты и на территориально распределенные команды.

Ниже рассмотрены отдельные, наиболее важные приемы экстремального программирования.

3.2.1 Игра в планирование

Команда экстремального программирования, как и Скрам-команда, работает на основе журнала пожеланий продукта и на каждой итерации практикует игру в планирование, цель которой — выбрать истории с наивысшим приоритетом. Команда совместно оценивает истории, добиваясь, чтобы каждый член команды был согласен с оценкой и готов сделать все для выполнения запланированной работы. Часто это происходит в виде коллективной игры — **покера планирования**, в которой стоимость истории оценивается по абстрактной шкале, в баллах.

После того как команда придет к единому мнению о стоимости каждой истории и относительных приоритетах, начинается работа; каждый этап продолжается обычно одну или две недели. Чаще всего команда контролирует ход работ и отчитывается о выполнении с помощью диаграммы сгорания, на которой показано, сколько баллов осталось до завершения работы, или обратной диаграммы сгорания, на которой показано, сколько баллов уже выполнено.

3.2.2 Заказчик всегда рядом

Обычно XP-команда сидит в одном помещении с представителем заказчика, который отвечает на вопросы и вправе принимать текущие решения о функциональности и пользовательском интерфейсе продукта. Как правило, команда соглашается с тем, что «карточка истории — это лишь повод для беседы» с заказчиком. Такие команды не записывают требования или истории во всех деталях, все участники работы над историей демонстрируют сделанное находящемуся рядом заказчику — быть может, несколько раз в день.

От заказчика ожидают заинтересованности и даже активного изменения своего мнения об истории после ознакомления с предъявленной работой, а команда должна четко указать стоимость внесения изменений, если такое происходит. Трудности XP-команды часто связаны с тем, что находящемуся рядом заказчику запрещено принимать решения без консультации с другими членами его организации, а это резко снижает скорость команды.

3.2.3 Парное программирование

В отличие от Скрам, когда команда может «пересматривать и адаптировать» технические приемы так, как считает удобным для себя, XP-команды привержены строгой дисциплине разработки ПО. Наиболее известны и понятны два приема: парное программирование и разработка через тестирование.

При парном программировании каждая история выбирается двумя разработчиками, которые обычно сидят за одним компьютером и пишут код совместно. У пары одна клавиатура на двоих, и они попеременно играют роль «водителя» и «штурмана». «Водителем» называется тот, кто в данный момент владеет клавиатурой и вводит код. «Штурман» держит в голове структуру будущей программы и думает о структуре кода, контекстах приложения и прочих требованиях.

Обычно пара регулярно меняется ролями, через каждые 15–60 минут, чтобы оба привыкали к смене контекста. Парное программирование позволяет разработчикам отделять контекст от таких несущественных деталей реализации, как синтаксис языка и детали API. Кроме того, при таком подходе на каждую строку кода смотрят две пары глаз, и в идеале один из членов пары держит в голове тестопригодность, удобство сопровождения и другие нефункциональные свойства кода, в т.ч. и безопасность.

Инженеры по безопасности могут (и должны) объединяться в пары с разработчиками при работе над функциями, каркасами и прочим кодом, имеющим прямое отношение к безопасности.

3.2.4 Разработка через тестирование

Основная идея разработки через тестирование состоит в том, чтобы сначала написать автоматизированный тест, а только потом — код, который этим тестом проверяется. Разработчики уже давно применяли автоматизированное тестирование, но только XP-разработчики довели эту практику до логического предела, выступая за стопроцентную тестопригодность.

Обычно этот подход выражают тремя словами: «красное, зеленое, рефакторинг». Разработчик пишет набор тестов, выражающих, что должен делать код. Затем создаются методы-заглушки, необходимые, чтобы тесты компилировались. При запуске тестов должен зажечься красный свет, означающий, что сборка не проходит из-за сбойного теста.

Затем разработчик пишет код таким образом, чтобы тест прошел. После этого тест прогоняется снова, и на этот раз зажигается зеленый свет, означающий, что тест прошел. Далее разработчик анализирует написанный код, пытаясь улучшить его, устранить дублирование и упростить структуру. Этот процесс называется рефакторингом, под этим понимается изменение внутреннего устройства кода без

изменения его поведения. Разработчик может вносить структурные изменения в код, будучи уверенным в его работоспособности, поскольку набор тестов отловит все ошибки и несовместимости.

Разработка через тестирование отлично работает в сочетании с парным программированием, поскольку оно поощряет «пинг-понговый» стиль работы, когда один программист пишет тест и передает его другому, а тот должен реализовать соответствующую функциональность, написать следующий тест и «вернуть мячик» назад.

Поскольку тесты должны быть акцентированы на том, что делает код, а не как он это делает, они дают возможность обсуждать вопросы проектирования, именования методов и инвариантов методов на этапе создания теста, и именно тесты определяют эти аспекты кода.

3.2.5 Метафора системы

XP включает множество практик, например, концепцию метафоры системы, согласно которой все члены команды должны пользоваться единым языком описания системы, поскольку это способствует коллективному владению кодом и общему пониманию задач.

Метафора системы нашла отражение уже в первом XP-проекте, где система формирования платежной ведомости была реализована как поточная линия, а различные поля платежной ведомости соответствовали рабочим местам на этой линии. В настоящее время эта практика уступила место предметно-ориентированному проектированию, когда от команды разработчиков требуется понимать и использовать язык предметной области, что способствует распространению знаний о ней среди членов команды.

3.3. Система Канбан

Система Канбан[10] [26, 27, 40] является методом оптимизации потока ценности с помощью процесса, который использует метод визуализации, основанный на

[10] Компания «Тойота» применила эту систему, основанную на «своевременном» производстве, в своих цехах, чтобы лучше соотнести внушительные складские запасы и реально используемые в производстве материалы. Для отслеживания объемов производства в цехе (и взаимодействия с поставщиками) в режиме реального времени использовалась специальная карточка Канбан, которую рабочие передавали между командами. Когда в корзине заканчивались используемые на участке производства материалы, на склад передавали карточку с указанием необходимого материала, нужного количества и т.д. На складе уже стояла новая корзина с этим материалом: ее отправляли в цех, а складские работники отсылали поставщику свою карточку Канбан. У поставщика корзина с этим материалом тоже была готова, и он отправлял ее на склад.

принципе вытягивания, предполагающим, что участник производственного процесса не начинает работу без сигнала от следующей в технологической цепочке операции и/или другого участника процесса.

Теоретически Канбан опирается на устоявшуюся теорию потока, включая, но не ограничиваясь: системное мышление, принципы бережливого производства, теорию очередей (размер партии и размер очереди), изменчивость и контроль качества. Постоянное совершенствование системы Канбан с течением времени, основанной на этих теориях, является одним из способов, с помощью которых команды могут попытаться оптимизировать доставку ценности. Теория, на которой основан Канбан, также разделяется другими существующими ценностно-ориентированными методологиями и структурами.

Методика Канбан включают 3 базовых правила:

1. Визуализируйте производство:
   - разделите работу на задачи, каждую задачу напишите на карточке и поместите на стену или доску;
   - используйте названные столбцы, чтобы показать положение задачи в производстве;
2. Ограничивайте работу, выполняемую одновременно на каждом этапе производства;
3. Измеряйте среднее время на выполнение одной задачи и постоянно оптимизируйте процесс, чтобы уменьшить это время.

Рабочие задачи визуально представляются на доске Канбан, что позволяет участникам команды видеть состояние каждой задачи в любой момент времени. Доска Канбан должна стать достоверным источником информации о работе. Доски нужны, чтобы визуализировать работу команды, стандартизировать процесс, а также находить и устранять блокирующие элементы и зависимости.

Доска Канбан обычно содержит следующие элементы:

1. **Ограничения на количество одновременно выполняемых работ** устанавливают максимальное количество элементов, которые могут существовать в отдельном состоянии рабочего процесса.
2. **Столбцы** представляют собой ряд шагов (состояний), представляющих действие, которое в совокупности определяет рабочий процесс группы.
3. **Карточки** представляют отдельные рабочие элементы, такие как функции, задачи, включая реализацию вспомогательных обеспечивающих средств.

4. **Дорожки** группируют и выделяют связанные рабочие элементы для дальнейшего определения класса рабочего процесса. Типичное использование дорожек включает в себя разделение работ для разных классов обслуживания. Классы обслуживания в Канбан имеют две основные цели: категоризация рабочих элементов в соответствии с их приоритетом и описание отдельных политик для определенного типа рабочих элементов. В общем случае можно рассматривать следующие классы обслуживания:

а. Фиксированная дата. Описывает рабочие элементы, которые необходимо доставить в определенную дату или до нее. Эти элементы должны быть специально идентифицированы и активно управляться для снижения риска отставания от графика. Среди прочего, этот класс обслуживания может гарантировать, что команды Канбан на основе потоков выполняют свои обязательства по зависимости с другими командами.

б. Высокоприоритетные внеплановые запросы. Этот класс предназначен для неожиданной, но срочной работы с высокой стоимостью задержки и поэтому требует немедленного внимания. В результате предметы на этой дорожке могут быть втянуты в разработку, даже если это нарушает текущие ограничения на количество выполняемых работ. Команды разработки должны иметь политики, ограничивающие использование этого класса услуг (например, в системе одновременно может быть только один элемент ускорения). Кроме того, команды могут установить политику «роения»[11], чтобы убедиться в прогрессе выполнения высокоприоритетных запросов.

в. Стандарт. Представляет базовый класс обслуживания, применимый к рабочим элементам, которые не являются ускоренными, не имеют фиксированной даты и не должны нарушать ограничения га количество одновременно выполняемых работ.

5. **Политики** определяют способ управления работой, например, критерии определения выполненной работы, выхода или входа для перемещения рабочего элемента из одного состояния в другое или определения правил для классов служб.

[11] Все члены команды разработки работают вместе, в одно и то же время, над реализацией одной задачи и могут использовать для этого все множество своих навыков.

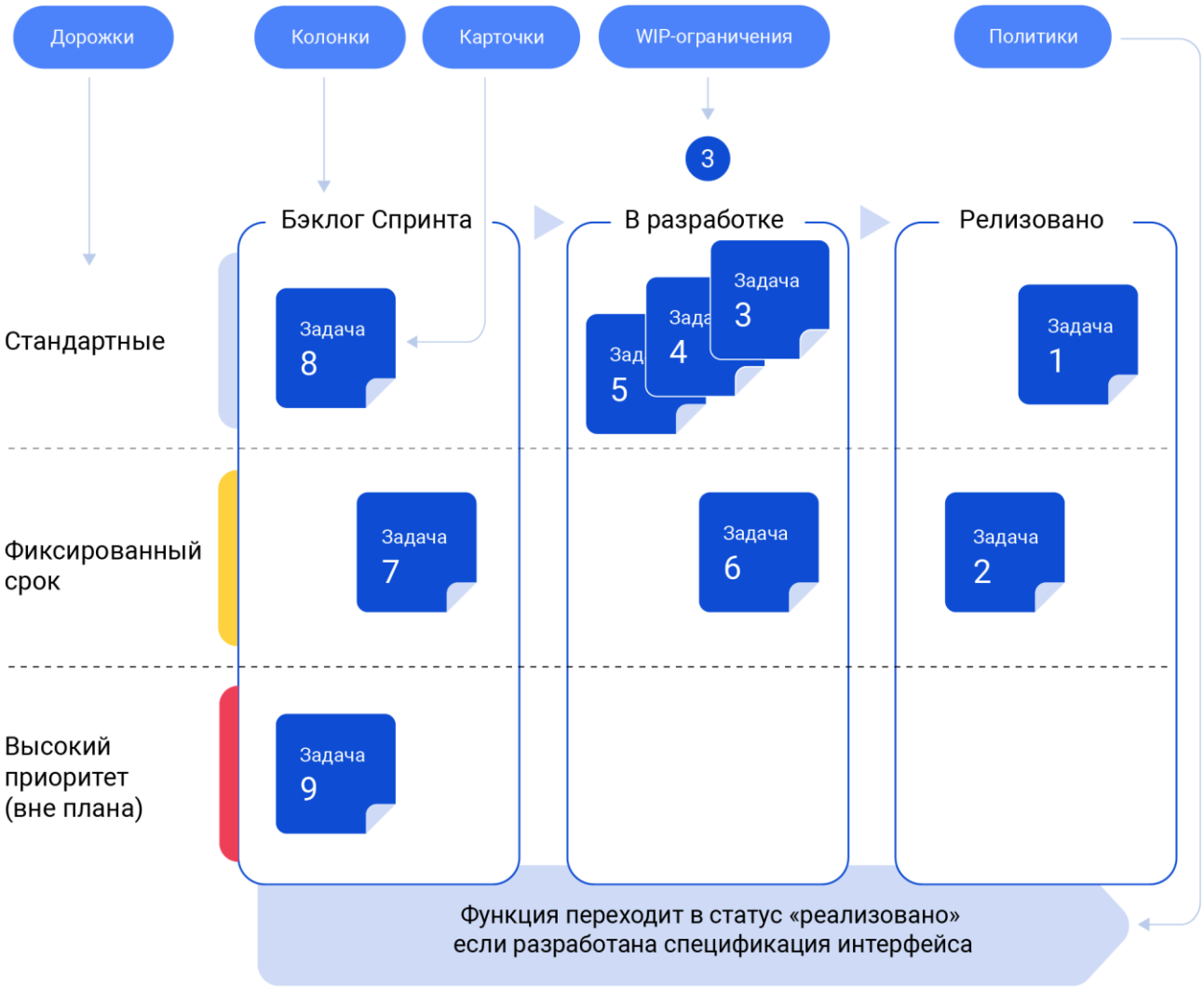

Рисунок 26. Элементы доски Канбан

На стандартной Канбан-доске *(Рисунок 26)* процесс может состоять из трех шагов: «Запланировано», «В работе» и «Сделано». Однако доску можно настроить в соответствии с процессом, принятым в той или иной команде, в зависимости от ее размеров, структуры и целей.

На рисунке ниже *(Рисунок 27)* показан пример доски Канбан команды, которая определяет четыре шага реализации функциональности (Пользовательских историй): Анализ, Разработка, Интеграция и Тестирование, а также для шага Разработка использованы буферы «В работе» и «Готово», чтобы упростить управление изменениями потока работ.

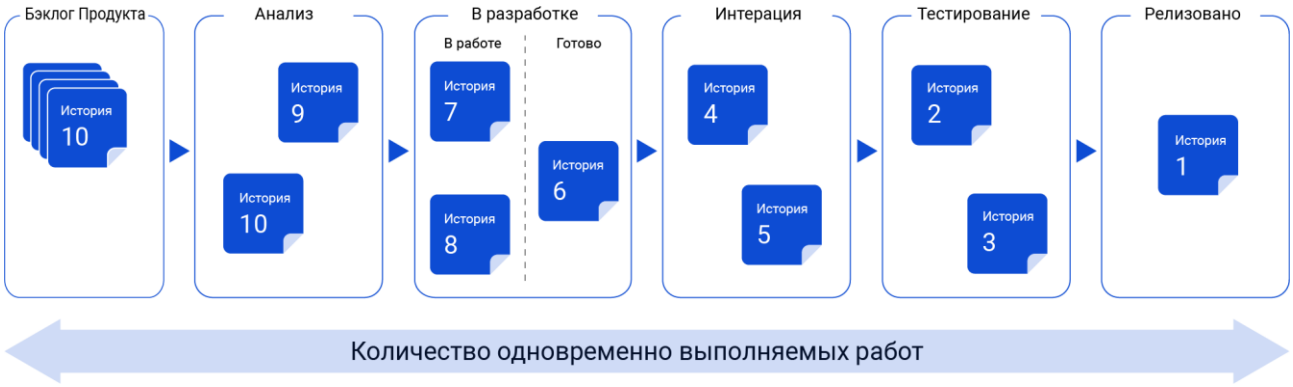

Рисунок 27. Пример «расширенной» Канбан — доски

Каждая рабочая задача на доске представлена в виде отдельной карточки, на которой отображается важная информация о конкретной рабочей задаче: имя ответственного за выполнение задачи, краткое описание выполненной работы, оценка необходимого времени и т.д. Когда все члены команды видят состояние каждой рабочей задачи в любой момент времени, а также всю связанную с ней информацию, повышается концентрация членов команды, обеспечивается полная прозрачность процесса разработки.

Ниже рассмотрены некоторые особенности процесса в рамках Канбан.

3.3.1 Сделать работу видимой

Канбан используется таким образом, чтобы каждая колонка соответствовала «рабочему месту» и показывала входную очередь, работы в процессе и выходную очередь. В команде разработчиков ПО могут существовать, например, такие рабочие места: анализ, разработка, тестирование и развертывание. Это позволяет визуализировать поток работ в команде, так что одного взгляда достаточно, чтобы понять, где накапливаются работы и возникают «затыки».

Канбан строго ограничивает количество незавершенных работ на каждом рабочем месте. Если количество работ превышает возможности рабочего места, то оно не может начать новую работу, пока не закончит предыдущую. Рабочее место с заполненной входной очередью не дает предыдущему рабочему месту переместить работу в выходную очередь. Это вызывает цепной эффект по всем предшествующим рабочим местам. И наоборот, когда последнее рабочее место завершает работу и готово принять следующую, все рабочие места передают законченные ими работы дальше.

3.3.2 Постоянная обратная связь

В рамках Канбан процесс не может продвигаться быстрее, чем самое медленное рабочее место. Если команда пытается работать быстрее, чем это рабочее место, то

она только впустую растрачивает ресурсы или создает задержки где-то в другом месте. Для решения этой проблемы используется метод, основанный на подаче работы к каждому рабочему месту точно в срок. Это значит, что каждое рабочее место запрашивает (или вытягивает) работу у предыдущего рабочего места в момент, когда оказывается готово взять следующую работу.

Используя постоянную обратную связь о потоке задач и наглядно представляя незавершенные работы, команда получает информацию о пропускной способности. Такая полная прозрачность гарантирует, что заинтересованные стороны могут увидеть текущее состояние системы, понять, сколько времени потребуется новым запросам для прохождения по ней, и соответственно расставлять приоритеты запросов на новые работы.

3.3.3 Сокращение продолжительности цикла

Продолжительность цикла — ключевой показатель для Команд разработки, использующих Канбан. Под продолжительностью цикла понимается время прохождения рабочей задачи цикла, от начала работы над задачей до завершения ее реализации. Оптимизировав продолжительность цикла, в будущем команда сможет с уверенностью предсказывать срок поставки задач.

Если теми или иными навыками обладает несколько человек, продолжительность цикла сокращается, если же только один — в рабочем процессе появляется «узкое» место. Благодаря обмену знаниями участники команды могут выполнять разнообразные задачи, что еще больше оптимизирует время цикла. Это также означает, что в случае возникновения узкого места в работе вся команда сможет взяться за него и восстановить нормальное течение процесса.

3.3.4 Гибкость планирования

Канбан позволяет командам концентрироваться только на текущей работе. По завершении рабочей задачи команда забирает следующую задачу с верха бэклога. Ответственный за реализацию (Владелец продукта) может менять приоритет задач в бэклоге, не мешая работе команды, поскольку изменения происходят за пределами текущих рабочих задач. Если Менеджер процессов Команды разработки следит за тем, чтобы наверху бэклога были самые важные задачи, Команда разработки будет гарантированно реализовывать самую важную функциональность в первую очередь. При этом Менеджеру процессов Команды разработки рекомендуется привлекать команду разработчиков к изменениям в бэклоге.

3.3.5 Непрерывная доставка функциональности.

Канбан и практика непрерывной поставки функциональности *(Раздел 4.2 данного Приложения)* идеально дополняют друг друга, поскольку обе методики основаны на своевременной (и последовательной) поставке ценности для пользователей. Поскольку Команды разработки, использующие Канбан, сконцентрированы на оптимизации процесса поставки продукта клиентам, чем быстрее команда сможет подготовить релиз продукта, тем более конкурентоспособным будет этот продукт.

3.4. Масштабирование кросс-функциональных команд

Небольшие команды показывают чаще хорошие результаты, чем большие, поэтому необходимо по возможности вести разработку компактными командами. Хотя такие команды являются кросс-функциональными, не всегда реально для команды от 5 до 11 человек обеспечить создание Систем, включающих в себя различные технологические платформы и спектр дисциплин, таких как аппаратное обеспечение, программное обеспечение и системная инженерия. Как правило, для реализации Системы требуется гораздо больше команд, которые должны работать согласованно.

Для организации разработки больших проектов или портфеля проектов необходимо масштабировать практики итерационной разработки на следующий уровень. На сегодняшний день наибольшее распространение получили следующие методологии масштабирования:

Scaled Agile Framework (SAFe) [84] — интегрирует возможности бережливого производства, методов итерационной разработки и DevOps в комплексную операционную методологию, которая позволяет разрабатывать инновационные цифровые продукты и услуги быстрее, с более высоким качеством и уровнем предсказуемости.

LeSS [71, 72, 73, 74] — методология, основанная на Скрам и применяемая к множеству команд, работающих совместно над одним продуктом. На сегодняшний день разработано два LeSS-фреймворка — Smaller LeSS (2..8 команд) и LeSS Huge (более тысячи человек, более 8 распределенных команд разработки).

Scrum@Scale[12] — способ масштабирования Скрам согласно базовым принципам Scrum и теории сложных адаптивных систем. В рамках Scrum@Scale все сотрудники организации поделены на несколько взаимозаменяемых команд разработки, которые могут сотрудничать в составе одной экосистемы вокруг одного общего набора целей.

[12] Scrum@Scale разработана совместными усилиями организации Scrum Inc. и ассоциации Scrum Alliance под руководством доктора **Джеффа Сазерленда**, одного из создателей Скрам.

Указанные методологии масштабирования включают общие организационные шаблоны *(Таблица № 1)* [88]:

- включают долгосрочные команды на основе единого планирования и стратегии их построения и развития;
- используется практика разработки с итерационным подходом (Скрам, Канбан и т.д.);
- в рамках выполнения работ регулярно проводится демонстрация разработанной функциональности системы (продукта) заинтересованным сторонам;
- регулярно проводятся ретроспективы с целью улучшения рабочих процессов;
- используются принципы клиентоцентричности и нацеленности на поставку ценности клиентам;
- осуществляется управление зависимостями между командами;
- масштабируются на практики управления портфелями;
- обеспечивается предоставление Релиза по запросу;
- осуществляется управление рисками;
- используется конвейер DevOps.

Таблица № 1.Организационные шаблоны методологий масштабирования

Основные характеристики методологий	**SAFe**	**LeSS**	**Scrum@Scale**
Команда команд	**ART (SRT)**	**Группа**	**Scrum of Scrum**
Долгосрочное планирование и стратегия			
Итерационная практика (Скрам, Канбан и т.д.)			
Регулярная демонстрация системы (продукта)			
Проведение ретроспективы			
Клиентоориентированность / ценность			
Управление зависимостями между командами			
Управление портфелем			
Предоставление Релиза по запросу			
Управление рисками			
Использование конвейера DevOps			

— определено и предписано
— определено как рекомендация
— нечетко определено

В рамках Методических рекомендаций, с учетом указанных методологий для решения задачи масштабирования кросс-функциональные команды предлагается объединить в Группы команд разработки *(Рисунок 28)*, а также обеспечить согласование решаемых задач в среде совместной работы, основанной на конвейере непрерывной разработки *(Раздел 4.2 данного Приложения)*. В рамках Группы команд все команды **совместно** проводят все мероприятия производственного цикла, как

показано на рисунке ниже, что позволяет избежать фокусировки Команд разработки исключительно на локальных проблемах отдельной команды. Такое согласование позволяет кросс-функциональным командам **независимо** и **скоординировано** выполнять разработку функциональности Системы. Для дальнейшего масштабирования (на расширенном уровне) Группы команд разработки объединяются в Пулы команд.

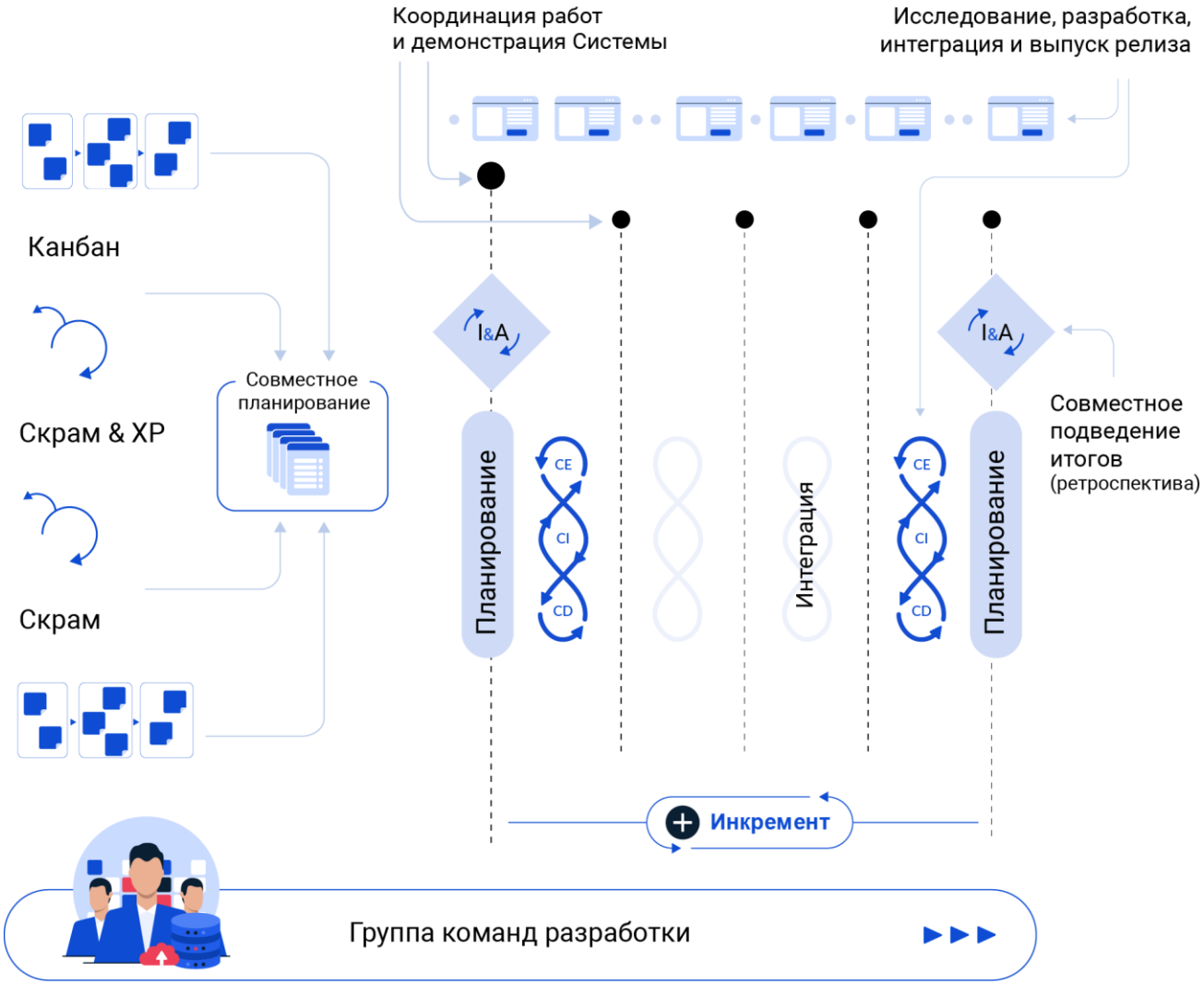

Рисунок 28. Масштабирование кросс-функциональных команд

4. ПРАКТИКИ DEVSECOPS

4.1. Общие положения DevSecOps

Разработке и развитию информационных систем присуще противоречие: процессы создания информационных технологий зависят от команд с противоположными целями и стимулами. Команды разработки должны быстро вносить изменения, в том числе по результатам оценки удовлетворенности конечных пользователей текущей версией Системы, а операционные группы (техническая поддержка) должны регулировать поток изменений, чтобы поддерживать стабильность уже созданной функциональности. Кроме этого, специалисты по информационной безопасности должны внедрить политики, предотвращающие внесение изменений в уязвимости, которые могут привести к утечке данных *(Рисунок 29)*.

Для устранения указанного противоречия широкое распространение получил набор технических практик DevOps[13], обеспечивающий связи, интеграцию, автоматизацию и тесное сотрудничество в Командах разработки, необходимый для планирования, разработки, тестирования, развертывания, выпуска и эксплуатации Системы [14, 19, 20, 32, 54, 59, 76, 87].

Рисунок 29. Схема DevSecOps

[13] DevOps — акроним, состоящий из двух слов: Development (Разработка) и Operations (Эксплуатация).

DevOps содержит набор ролей, инструментов и рекомендаций, а также «философию», которая состоит из следующих установок:

- коммуникация между командами разработки и технической поддержки должна быть прозрачной и эффективной;
- фокус на продукты, а не проекты;
- большая часть этапов работы должна быть автоматизирована;
- работа над продуктом не заканчивается после релиза (выпуска версии);
- создание продукта считается завершенным только тогда, когда он снят с производства.

DevOps предполагает разделение полномочий между участниками процесса при высокой степени автоматизации процессов сборки, тестирования и развертывания. Применение данного подхода позволяет создавать масштабируемые программные продукты, наращивая по мере увеличения числа программных модулей количество команд и перераспределяя их полномочия, а также делегируя задачи производства и контроля качества сравнительно небольшим командам специалистов.

Важным конкурентным преимуществом DevOps является включение специалистов технической поддержки в команды разработки и возможности устранять ошибки и расширять функциональность программных систем в режиме непрерывного развития (до вывода из эксплуатации). С этой целью используется конвейер непрерывной разработки, включающий такие инструменты как CI (Continuous Integration — постоянная интеграция, оперативное расширение функциональности) и CD (Continuous Delivery — постоянная поставка, выпуск релиза), а также обобщенный инструмент CI/CD. Их дополняет Continuous Exploration (непрерывное исследование того, что нужно создать или изменить в Системе, в том числе через обратную связь с конечным пользователем). Согласованное использование указанных практик образует конвейер непрерывной разработки Системы, как базовый рабочий инструмент создания и развития Системы *(Раздел 4.2 данного Приложения)*.

Изначально DevOps не включал безопасность в качестве первоочередной задачи явно, как это было в отношении разработки и эксплуатации. Чтобы избежать риска того, что безопасность останется второстепенной, появился термин DevSecOps [69, 84], который подчеркивает важность надежных методов информационной безопасности непосредственно в процессе создания (развития) Системы. Исследовательские проекты, посвященные DevOps, показывают, что уровень безопасности организации улучшается, когда безопасность полностью интегрирована в инструменты разработки цифровых продуктов и сервисов [11].

4.2. Конвейер непрерывной разработки

4.2.1. Общие положения

Конвейер непрерывной разработки — инженерная практика, используемая в том числе в DevOps, в рамках которой команды производят программное обеспечение в короткие циклы, гарантируя, что программное обеспечение может быть надежно выпущено в любое требуемое время. Он направлен на создание, тестирование и выпуск программного обеспечения с большей скоростью и частотой. Такой подход помогает снизить стоимость, время и риск внесения изменений за счет дополнительных обновлений приложений в производственной среде.

Конвейер непрерывной доставки представляет рабочие процессы, действия и их автоматизацию, необходимые для реализации новой функциональности Системы от разработки идей до выпуска версии, имеющей ценность для конечного пользователя [33, 34].

Создание и обслуживание конвейера непрерывной разработки предоставляет каждой группе команд возможность предоставлять пользователям новые функциональные возможности гораздо чаще, чем в традиционных (водопадных) процессах. В некоторых случаях «непрерывный» может означать ежедневные выпуски версий или даже выпуск несколько раз в день. В других случаях непрерывный может означать еженедельные или ежемесячные выпуски — все, что удовлетворяет требованиям и целям создания Системы.

Конвейер непрерывной разработки (КНР) определяет простые и повторяемые рабочие процессы и их автоматизацию, необходимые для реализации новой функциональности Системы, начиная от разработки идей до выпуска версии конечного пользователя. Как показано на рисунке ниже, конвейер состоит из четырех аспектов *(Рисунок 30):*

1. Непрерывное исследование (CE);
2. Непрерывная интеграция (CI);
3. Непрерывное развертывание (CD);
4. Выпуск по требованию.

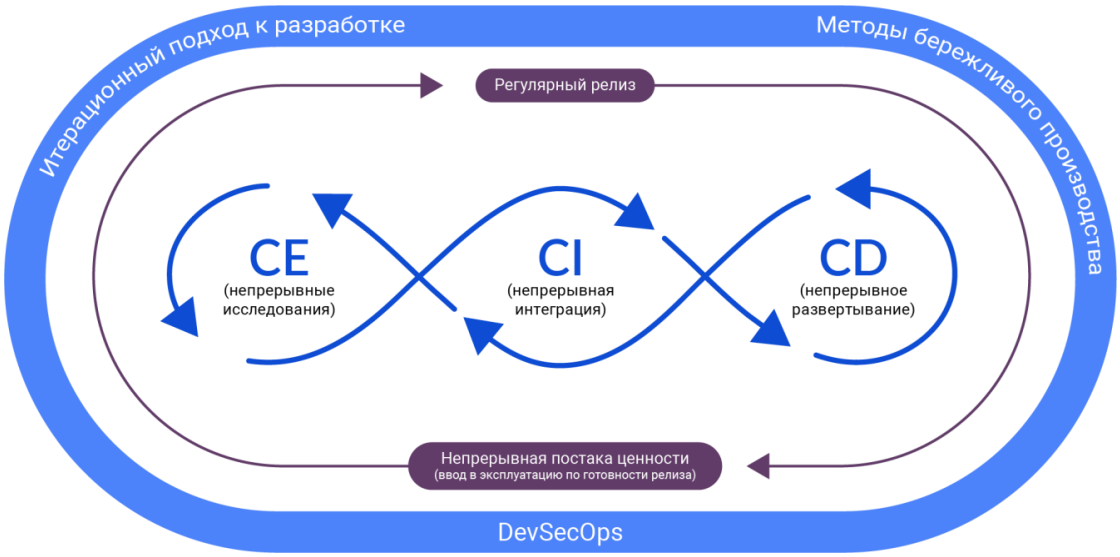

Рисунок 30. Конвейер непрерывной разработки

Внутренние циклы обратной связи направлены на улучшение процессов, в то время как внешняя обратная связь направлена на улучшение качества создаваемого продукта (Системы).

Каждый Пул кросс-функциональных команд создает и поддерживает или совместно использует конвейер с активами и технологиями, необходимыми для обеспечения ценности решения как можно более независимо. Первые три элемента конвейера (CE, CI и CD) работают вместе, чтобы поддержать поставку небольших партий новой функциональности, которые выпускаются для удовлетворения запросов пользователей.

Методика формирования и последовательных улучшений КНР в ходе выполнения работ представлена в Приложении к настоящему документу.

Непрерывное исследование (CE) фокусируется на понимании проблем/потребностей клиента и разработке предложений по решению для удовлетворения этой потребности. При этом используются практики дизайн-мышления (см. раздел 6). Исследование начинается с идеи или гипотезы о ценности для клиентов, как правило, в ответ на отзывы клиентов или исследования их запросов. Затем идеи анализируются и дополнительно исследуются, что приводит к пониманию того, что необходимо в качестве минимально-полезных наборов функций. Это, в свою очередь, обеспечивает пространство решений для изучения того, как должна быть разработана архитектура Системы, а также какие функции могут удовлетворить потребности клиента. При этом создается обратная связь с клиентом, позволяющая разработчикам выяснить, как программное обеспечение должно функционировать, когда еще есть много времени для внесения изменений. И с точки зрения устойчивости системы, создаваемые поэтапно, с минимально-полезными наборами функций, обычно легче

поддерживать, поскольку поэтапная реализация может быть растянута на срок жизни системы.

Непрерывная интеграция (CI) фокусируется на реализации функций из Бэклога и последующем их уточнении. После того, как конкретные функции спроектированы, Команды разработки реализуют их и интегрируют в существующую функциональность Системы в контуре разработки для проведения комплексного тестирования.

Непрерывное развертывание (CD) предполагает передачу версий системы из контура тестирования в контур эксплуатации. Этот шаг делает функции доступными в рабочей среде, где Ведомство определяет подходящее время для их выпуска, релиза Системы для клиентов. Этот аспект также позволяет Рабочей группе откатывать или исправлять ошибки, когда это необходимо.

Выпуск по требованию — это возможность сделать реализованную функциональность доступной для клиентов сразу или в определенном порядке на основе понимания актуальных потребностей клиентов. Это также позволяет тщательно контролировать величину риска, связанного с каждым вводом релиза в эксплуатацию. Выпуск по требованию также включает в себя критически важные действия конвейера непрерывной разработки, которые сохраняют стабильность релиза в течение длительного времени после выпуска.

Хотя этапы реализации функциональности в конвейере описываются последовательно, конвейер работает в итерационном режиме, который позволяет Командам разработки формировать одну или несколько гипотез, строить решения для проверки каждой гипотезы, проводить исследование гипотез и извлекать уроки из реализации гипотез.

4.2.2. Метрики улучшения производственного процесса

Команды разработки, как правило, имеют конвейер разработки — в противном случае они не смогли бы выпустить какую-либо ценность вообще.

Первым шагом к улучшению потока ценностей является изучение текущего конвейера. Со временем существующий процесс должен быть расширен таким образом, чтобы обеспечить изменения в производственном цикле — от реализации новых функций до обслуживания Системы в ходе ее эксплуатации.

Для улучшения производственного процесса используют метрики:

1. Время процесса — это время, необходимое для выполнения работы за один шаг;
2. Время выполнения — это время, которое требуется с момента выполнения работы на предыдущем шаге до ее выполнения на текущем шаге;

3. Время задержки — это время, когда не происходит никакой работы. Понимание и устранение ненужных задержек имеет решающее значение для улучшения потока ценностей;
4. Процент завершенности и точности представляет собой процент работы, которую следующий шаг может обработать без необходимости доработки. Часто задержки вызваны низким качеством в восходящих (предыдущих) шагах. Процентная полная и точная метрика помогает определить шаги, где может происходить низкое качество и приводить к увеличению времени выполнения заказов, что приводит к задержкам доставки стоимости.

4.2.3. Определение возможностей для улучшения производственного процесса

Команды ищут возможность повысить эффективность каждого шага, следовательно, сокращая общее время выполнения работ. Это включает в себя определение времени процесса, а также качества (в процентах завершения и точности) каждого шага. Чем выше это число, тем меньше требуется доработок, и тем быстрее работа движется по системе.

Время задержки (время между шагами) часто является наиболее значительным начальным фактором. Время задержки представляет собой передачу, ожидание и другие отходы, не связанные с добавленной стоимостью. Этот процесс имеет две значительные задержки и значительный объем доработок на первом этапе процесса развертывания. Сокращение задержек, как правило, является самым быстрым и простым способом снижения общего времени выполнения заказа. Еще одной приоритетной областью для улучшения является любой шаг с низкими показателями затрат времени и ресурсов, поскольку сокращение доработки позволяет Командам разработки сосредоточиться на создании ценности (например, для программного решения, вместо исправления ошибок команда может сосредоточиться на новых функциях).

4.2.4. Визуализация производственного процесса

Визуализация производственного процесса предполагает расположение всех инструментов, деталей, производственных стадий и информации о результативности работы производственной системы таким образом, чтобы они были четко видимы, и чтобы каждый участник производственного процесса моментально мог оценить состояние производственной системы (ГОСТ Р 56020-2020).

Когда располагаешь автоматизированным конвейером к производственной среде, становится особенно важно знать, что именно по нему движется, и предотвращать вмешательство одной команды в работу другой.

Несмотря на тестирование и автоматизацию, ошибки все же возможны, и особенно часто они вызваны зависимостями между компонентами. Может случиться, что некоторый компонент зависит от компонента, разрабатываемого другой командой. В таких случаях не исключено, что первый компонент интегрируется слишком рано и результат доходит до производственной системы раньше, чем готов компонент, от которого он зависит. Практически в любой методике взаимодействию команд придается первоочередная важность, а самым распространенным механизмом является визуальное прослеживание работы, удовлетворяющее ряду требований:

Видимость. Любой член команды и смежных команд должен сразу видеть, над чем ведется работа и что находится на пути в производственную среду.

Актуальность и полнота. Чтобы информация была полезной и надежной, она должна быть полной и актуальной. Все элементы проекта — журнал пожеланий, список дефектов, уязвимости, незавершенные работы, контрольные события и скорость работы, время цикла, риски, текущее состояние сборочного конвейера — все должно находиться в одном месте и обновляться в реальном времени.

Простота. Это не система для отслеживания всех детальных требований к каждой части работы. Каждый элемент должен кратко представлять часть работы и показывать несколько основных атрибутов, владельца и текущее состояние.

В качестве инструментов визуального отслеживания могут выступать отрывные листочки или каталожные карточки, наклеенные на стену или на доску Канбан, трекеры историй и другие.

Чтобы заинтересованные стороны в Рабочей группе могли визуализировать и отслеживать текущую работу Команд разработки, предлагается использовать систему Канбан *(Рисунок 31).*

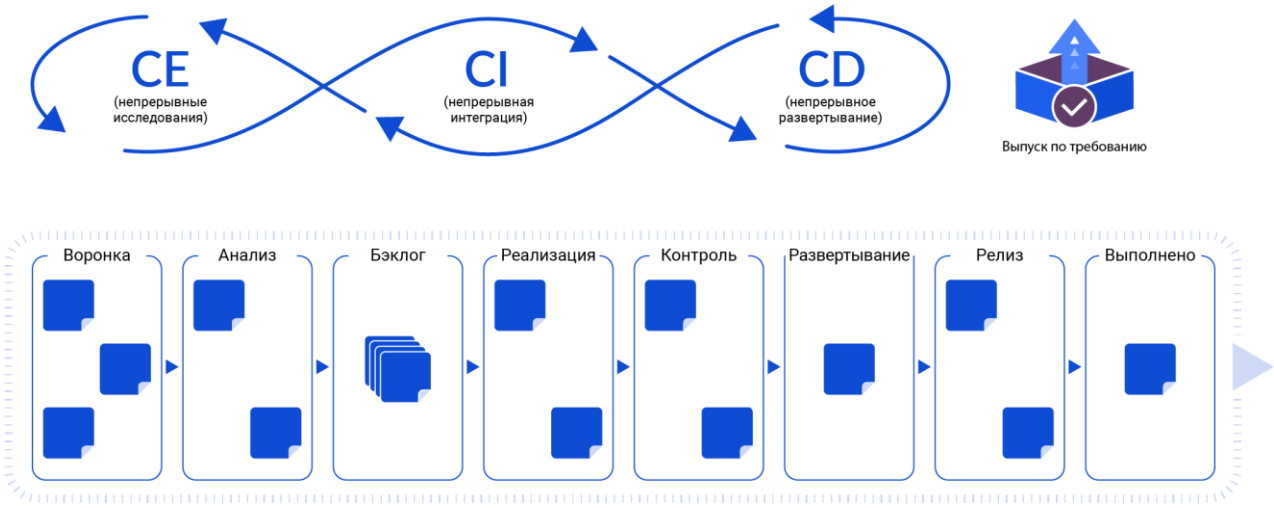

Рисунок 31. Отслеживание непрерывной поставки с помощью Канбан

Информационная доска Канбан, при этом, может состоять из следующих основных состояний:

1. **Воронка** — определение новых функций или улучшение существующей функциональности.
2. **Анализ** — уточнение функций с ключевыми атрибутами, включая гипотезу ценности для клиентов и критериев их реализации.
3. **Бэклог** — после анализа функции ранжируются по приоритету их реализации.
4. **Реализация** — основные функции из Бэклога разрабатываются и интегрируются в существующую функциональность Системы.
5. **Контроль** — тестирование реализованной функциональности.
6. **Развертывание** — по мере завершения работ реализованные компоненты развертываются в контуре эксплуатации, где они ожидают выпуска.
7. **Релиз** — развертывание в контуре эксплуатации и проверка гипотез ценности реализованной функциональности для клиента.
8. **Готово** — завершение работ при необходимости, предоставление реализованной функциональности клиентам.

4.3. Обеспечение безопасности

Как сказано выше, идея DevSecOps состоит в том, чтобы сделать безопасность обязательным требованием в процессе проектирования, разработки и выпуска релизов [14], включая:

а. организацию команды по безопасности, которая должна работать в тесном взаимодействии с разработчиками и специалистами по эксплуатации;

б. реализацию стратегии непрерывной безопасности, включая такие практики как безопасность на основе тестирования, мониторинг и реагирование на атаки, а также оценку рисков и непрерывное усиление безопасности;

в. включение в состав конвейера непрерывной разработки таких традиционных техник, как сканирование уязвимостей, обнаружение вторжений и журналов мониторинга безопасности.

г. Проверки и тесты безопасности должны быть автоматизированы способом, который позволяет легко и прозрачно вставлять их в технологические процессы разработки и сборочные конвейеры.

4.3.1. Организация команды безопасности

Из-за высокой скорости, с которой современные, практикующие DevOps команды передают системы в производственную среду, а также из-за темпа внесения изменений в уже используемые системы разработчики должны располагаться гораздо ближе к инфраструктуре. В рамках практик DevOps отсутствуют специальные мероприятия по передаче в отдельные группы эксплуатации и обслуживания, когда разработчики переходят к другому проекту, как было в каскадной модели. Модель работы команд основана на концепции оказания услуг — команда несет совместную, а иногда и полную ответственность за эксплуатацию и поддержку системы. Таким образом, команды могут быть связаны с системой на всем протяжении срока ее службы.

Это означает, что команда должна думать не только о тех, кто будет пользоваться системой, но и о тех, кто будет ее эксплуатировать и поддерживать: инженерах службы обеспечения инфраструктуры и сети, эксплуатационниках и службе поддержки клиентов. Все они становятся заказчиками и партнерами по принятию решений о том, как следует проектировать и реализовывать систему.

Организация взаимодействия команд разработки со специалистами по безопасности представляет, как правило, сложную управленческую задачу. Это обусловлено тем, что исторически процессы и практики в области безопасности создавались для крупных проектов, разрабатываемых по каскадной технологии с предварительной фиксацией требований, а не маленькими командами, работающими быстро и итеративно.

Специалисты в области безопасности испытывали большие трудности, пытаясь адаптировать существующие методы к процессам, где требования могут изменяться (а иногда не формулируются в письменном виде), где решения, относящиеся к проектированию и управлению рисками, принимаются командой, а не планируются и не спускаются сверху вниз, где ручное тестирование и контроль соответствия не поспевают за скоростью поставки.

Существует практика, когда специалисты по безопасности стремятся запрещать любые изменения, чтобы свести к минимуму изменения в моделях угроз. Если группа безопасности стремится уменьшить риск ценой отказа от изменений, вместо того чтобы помогать группе разработки в реализации требований безопасности, то возможно возникновение ситуаций, когда работы будут проводиться в обход ее. Это, в свою очередь может привести к тому, что системы станут менее безопасными, менее защищенными, не соответствующими нормативным требованиям.

Для решения указанной проблемы возможны следующие меры:

1. Ответственные за реализацию (владельцы продуктов) должны давать своим командам больше времени на надлежащую проработку вопросов безопасности, они должны понимать требования к безопасности и соответствию нормативным требованиям и назначать им соответствующие приоритеты.
2. Гибкие команды должны понимать и принимать меры обеспечения безопасности и брать на себя больше ответственности за безопасность создаваемых систем.
3. Команда безопасности должна не заниматься безопасностью самостоятельно, а содействовать работе команд разработки по вопросам безопасности. При этом основная задача команды безопасности состоит в том, чтобы создавать инструменты, документировать практические методы и обеспечивать возможность разработки и развертывания безопасных служб.
4. Команда безопасности не должна уступать в гибкости команде разработчиков. Она должна действовать быстро и итеративно, реагировать оперативно, учиться и совершенствоваться вместе с разработчиками.

4.3.2. Реализация стратегии непрерывной безопасности

Непрерывная безопасность состоит из трех областей *(Рисунок 32)*, каждая из которых сосредоточена на особом аспекте конвейера непрерывной разработки DevOps.

1. **Безопасность на основе тестирования**. Первыми шагами к безопасности программы являются определение, реализация и тестирование управления безопасностью. Безопасность на основе тестирования охватывает простые средства управления, такие как стандартная конфигурация серверов или настройки безопасности, которые должны быть реализованы в клиентских приложениях. Безопасности можно добиться последовательной реализацией основных мер безопасности и тестированием их соблюдения. В эффективной системе DevOps случаи ручного тестирования должны быть исключением, а не правилом. Тестировать безопасность следует так же, как и все приложения в CI- и CD-конвейерах: автоматически и повсеместно.
2. **Мониторинг и реагирование на атаки.** Второй фазой непрерывной безопасности выступают мониторинг и реагирование на угрозы, а также защита сервисов и критических данных, включая использование таких техник, как выявление мошенничества и вторжений, цифровая криминалистика, реагирование на инциденты.

3. **Оценка рисков и усиление безопасности.** Однако стратегия безопасности не может стать успешной, если основывается только на технических аспектах. Третья фаза непрерывной безопасности предполагает выход за пределы технологии и использование организационных аспектов безопасности, таких как управление рисками, внешнее и внутреннее тестирование безопасности, планирование действий в области безопасности в рамках планирования итераций разработкиистемы.

Рисунок 32 Три фазы обеспечения непрерывной безопасности

Следует отметить, что примерно половина уязвимостей находятся на уровне кода и являются результатом простых ошибок программирования, небрежности со стороны разработчиков, игнорирования требований, непонимания или неправильного применения языка, библиотек и фреймворков. Есть два основных подхода к обнаружению в коде ошибок, в т. ч. связанных с безопасностью:

1. **Тестирование.** Автоматизированное или ручное, включая сканирование на предмет уязвимости, когда система рассматривается как черный ящик.
2. **Инспекция кода.** Включая самоинспекцию, парное программирование и дружественную оценку, внешний аудит кода и автоматизированное сканирование кода.

Как для инспекции, так и для автоматизации возможно использование производственного конвейера на разных стадиях непрерывной интеграции или поставки.

Тесты и проверки, выполняемые до непрерывной интеграции:

- статическая проверка кода до записи в репозиторий с использованием встроенных средств проверки кода или подключаемых модулей;
- сканирование в поисках секретных данных перед объединением кода;
- инспекция кода.

Тесты и проверки безопасности в ходе непрерывной интеграции:

- проверки на этапе сборки, включая обнаружение новых компонентов и компонентов с известными уязвимостями;
- автономное тестирование, особенно негативные тесты ключевых функций;
- инкрементный статический анализ, соблюдение правил оформления и проверки использования запрещенных функций и других опасных практик;
- проверки безопасности «на дым»: быстрые и простые проверки, оформленные в виде сценариев.

Тесты и проверки безопасности в ходе непрерывной поставки:

- целенаправленное сканирование приложения;
- автоматизированные атаки;
- автоматизированные приемочные тесты функций безопасности (аутентификация, контроль доступа, управление удостоверениями, аудит и криптозащита).

Перед тем как добавлять тестирование безопасности в конвейер, необходимо убедиться, что конвейер настроен правильно, и что команда использует его корректно и последовательно, в том числе:

- все изменения записываются в репозиторий кода;
- члены команды регулярно и корректно производят записи в репозиторий;
- автоматизированные тесты выполняются быстро и одинаково;
- если тест завершается неудачно, то команда прекращает работу и немедленно исправляет ошибки до внесения новых изменений.

При этом если команда не использует автоматизированный сборочный конвейер в ходе развертывания релиза, или если она игнорирует ошибки тестирования в конвейере, то добавление проверок безопасности не принесет значимого эффекта.

4.3.3. Обеспечение безопасности конвейера непрерывной разработки

Автоматизация процессов сборки, интеграции и тестирования, реализуемая использованием конвейера непрерывное разработки, — одна из основ организации итерационной разработки. При этом задачи, решаемые с использованием конвейера, а также полномочия разработчиков при его использовании обусловливают риски несанкционированного доступа к компонентам разрабатываемой Системы.

Критичность защиты конвейера непрерывной разработки многократно возрастает, если команды разработки практикуют непрерывное развертывание, когда каждое изменение автоматически попадает в производственную систему после тестирования. Автоматизация сборки и развертывания увеличивает уязвимости производственной системы —так как она включает также среду и комплекты инструментов сборки. Если репозитории, сборочные серверы или системы управления конфигурацией скомпрометированы, то ситуация очень быстро становится крайне серьезной и для контура эксплуатации.

Если в результате компрометации получен доступ на чтение, то объектом кражи могут стать данные, исходный код и различные секреты, в т. ч. пароли и ключи прикладных интерфейсов. Если же получен доступ на запись или выполнение, то сброшены уже все маски: открывается возможность внедрить в приложения черные ходы или вредоносный код, перенаправить или перехватить рабочий трафик и, наконец, полностью уничтожить производственные системы.

Даже если скомпрометирована только тестовая система, этого может хватить противнику, чтобы внедриться в автоматизированный конвейер и причинить ущерб. Утратив контроль над сборочным конвейером, вы утратите также возможность реагировать на атаку, поскольку не сможете развернуть заплату или срочное исправление.

Защищать конвейер нужно не только от внешних угроз, но и от компрометации инсайдерами. Для этого все изменения должны иметь автора, быть прозрачными и прослеживаемыми от начала до конца, чтобы замысливший вред, и при этом информированный инсайдер не мог обойти средства контроля и внести изменения, оставшись незамеченным.

Необходимо создание модели угроз конвейеру, включающей слабые места в настройке и средствах контроля, а также пробелы в аудите и протоколировании. Рекомендуется выполнить следующие действия, чтобы обезопасить среду управления конфигурацией и сборочный конвейер:

1. Укрепить исполняющую среду управления конфигурацией, сборки и тестирования — разделить на сетевом уровне контуры разработки и эксплуатации, использовать контейнеры и виртуальные машины для обеспечения дополнительной изоляции;

2. Выяснить, какие действия производятся в облаке, и принять меры для их контроля — заставьте разработчиков применять строгую аутентификацию (включая и многофакторную), убедится, что частные репозитории действительно частные, следить за своими репозиториями на (в случае их использования) с помощью специальных инструментов монитринга, перед тем как записать код в репозиторий, просканировать или проинспектировать его на предмет отсутствия учетных данных;
3. Укрепить комплекты инструментов для сборки, непрерывной интеграции и непрерывной поставки — настройте отношения доверия между мастерами сборки и серверами и включите прочие имеющиеся механизмы защиты, логически разделите сборочные конвейеры разных команд (если один будет скомпрометирован, то хотя бы другие не пострадают), ограничьте доступ к инструментам и, включив средства обеспечения безопасности, вовремя устанавливайте обновления и исправления самих инструментов и всех необходимых подключаемых модулей;
4. Ограничить доступ к инструментам управления конфигурацией;
5. Защитить ключи и другие секретные данные, в том числе убедиться, что эти данные не зашиты в скрипты, незашифрованные конфигурационные файлы или в код;
6. Ограничить доступ к репозиториям исходных и двоичных файлов, в том числе запретить не аутентифицированный, анонимный или совместный доступ к репозиториям и установить правила контроля доступа;
7. Защитить платформы чатов;
8. Регулярно анализировать журналы управления конфигурацией, сборки и тестирования;
9. Использовать серверы-«фениксы» для создания подчиненных узлов сборки и тестирования; создавать такие среды с нуля всякий раз, как в них возникает надобность, и уничтожать по завершении работы. Поскольку создать хорошие синтетические тестовые данные трудно, в ряде случаев берут реальные данные из производственной системы или их подмножество и маскируют некоторые поля, стирая в них персональные данные. Если сделать это неправильно, то возможна утечка данных или другие нарушения законов о персональных данных и иных нормативно-правовых требований. Необходимо строго контролировать обращение со снимками системы и внедрить надежные (и тщательно проинспектированные) методики, гарантирующие, что персональная информация: имена, адреса, телефоны, адреса электронной почты, а также пароли, другие учетные данные и прочие секретные сведения — удалена и заменена случайными

данными или иным способом сделана бесполезной, — только потом данные можно использовать для тестирования. Все шаги этой процедуры должны фиксироваться, тогда вы сможете доказать клиентам и аудиторам, что информация была защищена;

10. Наблюдать за сборочными конвейерами так же, как за контуром эксплуатации. Компоненты конвейера должны подвергаться мониторингу как часть производственного контура. Эксплуатационную безопасность, в т.ч. сканирование на уязвимости, системы обнаружения и предотвращения вторжений, мониторинг нужно распространить на инструменты и инфраструктуру, в которой они используются. Также необходимо применять средства проверки целостности файлов, чтобы обнаруживать неожиданные или неавторизованные изменения конфигурации и данных.

4.4. Обеспечение выпуска релизов

Выпуск релизов Системы позволяет получить раннюю обратную связь от конечных пользователей и/или заказчика Системы. Сервисная архитектура, определение стандартных интерфейсов между компонентами Системы, позволяют независимо выпускать более мелкие изменения на уровне компонентов. Меньшие изменения обеспечивают более быстрые, частые и менее рискованные выпуски, но для обеспечения качества требуется выполнение ряда обеспечивающих выпуск релизов мероприятий *(Рисунок 33)*.

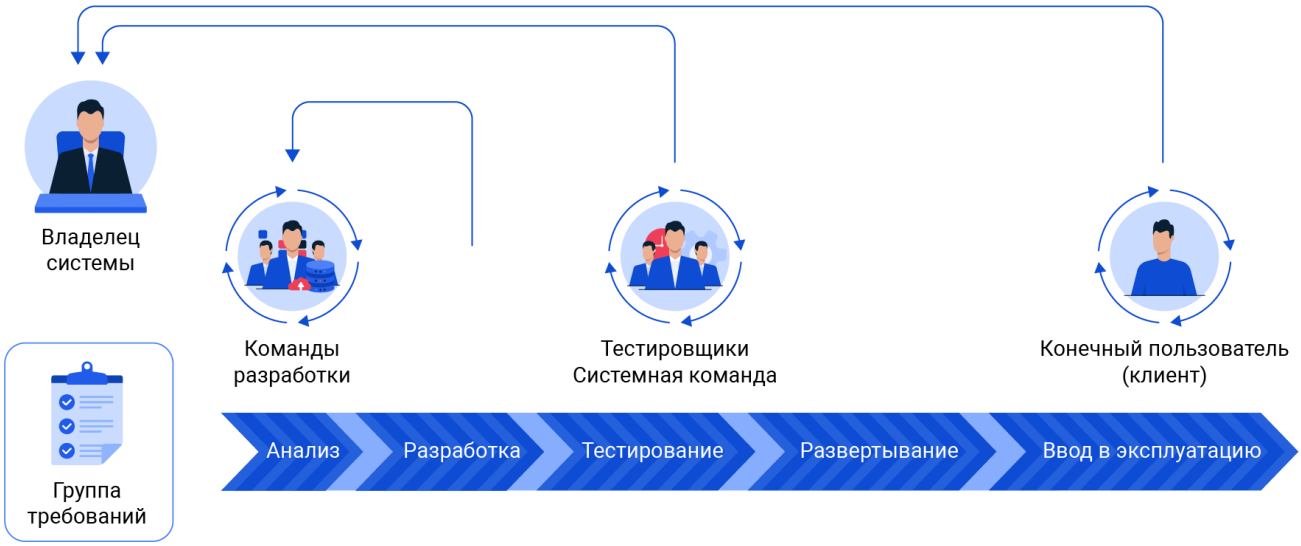

Рисунок 33. Типовой процесс выпуска релиза

В то время как использование принятых производственных процессов, качество программного кода и архитектуры гарантируют, что системные артефакты могут быть

легко реализованы и изменены, готовность релиза в целом подтверждает, что Система работает так, как ожидалось.

4.4.1. Фиксация выполненных работ

Эффективные команды постоянно изменяют функциональность системы и предоставляют ее клиентам, используя петлю обратной связи. Но для оптимизации времени цикла необходимо знать, какие данные собирать и, в частности, оценивать, произвела ли выполненная работа полезный эффект.

Масштабирование Системы приводит к тому, что использование конвейера непрерывной разработки несколькими командами требует определения выполненной работы, чтобы гарантировать, что необходимая работа выполняется в нужное время разными командами и группой команд в целом. Определение выполненных работ является важным способом обеспечения того, чтобы разработку версии Системы можно было считать завершенной. Данное определение в общем случае включает набор условий, которые должны быть выполнены всеми участниками процесса разработки в соответствии с принятым производственным циклом *(Рисунок 33*), включая разработку документации (*Раздел 4.4.2*) и проверку на соответствие нормативным требованиям *(Раздел 4.4.3).*

Многие команды вносят множество небольших изменений, которые необходимо постоянно проверять на наличие конфликтов и ошибок. Методы непрерывной интеграции и непрерывной поставки обеспечивают разработчикам быструю обратную связь. Каждое изменение быстро создается, а затем интегрируется и тестируется на нескольких уровнях, включая среду развертывания. Конвейер непрерывной интеграции и непрерывной доставки автоматизирует процесс перемещения изменений на всех этапах и обеспечивает необходимые реакции (уведомления) в случае непрохождения теста.

Фиксация выполненных работ в среде производственного конвейера информирует участников разработки и иных заинтересованных лиц в завершении запланированных работ.

4.4.2. Документирование итеративной разработки

Обеспечить возможность внесения изменений в документацию, согласованных с изменениями в Системе, можно с помощью включения этих работ в общий процесс итеративной разработки, начиная от планирования и завершая демонстрацией реализованной функциональности, испытаниями и выпуском релиза [86]. Это обеспечит согласованное инкрементальное наращивание и функциональности Системы, и создания документации на нее.

Проблемы поддержания документации в актуальном состоянии при итеративной разработке систем

Сложные комплексы программ и их компоненты, образующие Систему, непрерывно меняются в процессе всех этапов жизненного цикла Системы. Соответственно должны изменяться документы, отражающие актуальное состояние процессов реализованных компонентов и состояние процессов разработки. Для этого в рамках производственного конвейера должно быть реализовано итерационное, гибкое и точное изменение документов. Процессы документирования и поддерживающие их средства автоматизации должны быть адекватны методам разработки и применяемым средствам автоматизации. Процессы документирования должны быть поддержаны организацией контроля качества, регистрации изменений и утверждения версии каждого документа.

Общие типовые требования к номенклатуре, структуре и содержанию документов на Систему представлены в ГОСТ (например, ГОСТ 34.201-2020 «Виды, комплектность и обозначение документов», ГОСТ Р 59795-2021 «Требования к содержанию документов»), а конкретные требования могут быт указаны в контракте на создание (модернизацию) Системы.

При всех недостатках каскадной модели разработки, основанной на ГОСТ 19 и 34 серий и их аналогов, грамотно выстроенный процесс разработки позволяет порождать необходимую документацию в процессе перехода от этапа к этапу. Причем для специалиста, который только начинает знакомиться с системой, сложность возрастает линейно по мере погружения в предмет исследования — от общих проблем к более частным.

В рамках создания и развития Систем с использованием итерационного подхода происходит регулярное и частое обновление функциональности Системы. При этом изменения функциональности, видимые конечному пользователю, должны быть синхронизированы с внесением изменений в документацию. К примеру, конечный пользователь (клиент) может попросту не заметить новую функциональность, если комплект поставки не будет включать подробного описания новой функциональности и сценариев ее использования.

Применение средств производственного конвейера для автоматизированного тестирования и развертывания позволяет с достаточной скоростью и степенью точности фиксировать, а затем многократно тиражировать состояние программно-аппаратного комплекса Системы, что в теории должно давать 100%-ную гарантию работоспособности продукта. Однако, поскольку DevOps не содержит формальных и жестких рекомендаций к оформлению документации, это приводит к тому, что специалисты, участвующие в создании Системы, могут использовать несколько ссылок на репозитории проекта, базы знаний разработчиков, тестировщиков,

менеджеров по качеству, техподдержки и маркетинга, а также трекеры задач и ошибок, что представляет собой существенные объемы плохо структурированного текста.

При первом приближении кажется, что достижение максимальной полноты информации о производственном процессе должно способствовать разностороннему пониманию технологического процесса, однако на деле для консолидации специалистов в схожих по сути, но в то же время различных по подходам предметных областях (например, тестирование) требуется привлечение экспертов по формализации знаний, что существенно повышает стоимость разработки.

Предложения по разработке документации в рамках итеративной разработки

Процессы документирования Систем должны входить органически в весь жизненный цикл, поэтому организация и выполнение работ по созданию документов должны распределяться между специалистами, ведущими непосредственную разработку ПО, и специалистами, осуществляющими в основном разработку, контроль и издание документов. В общем случае специалисты по документированию (технические писатели) могут включаться в состав Команды разработки (при уровне их загрузки более 80%). При создании сложных крупномасштабных информационных систем целесообразно создание специальной обеспечивающей Команды разработки в составе Группы или Пула команд.

При планировании следует включать в состав работ по реализации функциональности, помимо разработки и/или адаптации программного обеспечения, тестирования, также и документирование реализованной функциональности. Таким образом, функциональность считается готовой только тогда, когда она в том числе описана в пользовательской документации. В итоге, после завершения реализации функциональности Система будет готова к испытаниям и вводу в эксплуатацию.

В ходе выполнения работ целесообразно использование традиционных инструментов, упрощающих следование гибким процессам. Одним из таких инструментов является доска Канбан (раздел 3.3), обеспечивающая визуализацию и контроль работ. При этом задачи по документированию должны отображаться на доске наряду с задачами по разработке и тестированию.

В интересах установления связи между реализуемыми требованиями и их документированием рекомендуется декомпозировать документацию на отдельные детали, каждая из которых представляет атомарную автономную документацию конкретного реализованного требования. Тогда станет возможной двусторонняя навигация: от конкретной пользовательской истории к ее описанию в пользовательской документации и обратно от конкретной части документации к тому требованию, которое было реализовано и породило данное описание. В итоге,

сценарий обновления документации превращается в механическую работу: найти и добавить в пользовательскую документацию описания требований, реализованных за время работы над последним выпуском, обновить в пользовательской документации описания измененных требований, удалить описания исключенных из выпуска функций.

Одна из проблем, указанных выше, заключается во внесении требуемых изменений при сборке пользовательской документации. Решение проблемы может заключаться в использовании специальных репозиториев, поддерживающих контроль версий и многопользовательскую работу в распределенных средах, а также средств автоматизации, обеспечивающих синхронизацию отдельных спецификаций (например, описания пользовательских историй) из репозиториев Команд разработки, а также сборки из этих отдельных спецификаций финальных документов, состав и структура которых определены Техническим заданием (или ГОСТ) с обеспечением контроля их качества, регистрации и утверждения.

4.4.3. Поддержка соответствия нормативным требованиям

Для систем, которые должны продемонстрировать объективные доказательства соответствия, выпуск версии имеет дополнительные условия. Исполнитель должен доказать, что Система соответствует своему назначению и не имеет непреднамеренных, вредных последствий.

Любой крупный отказ Системы обычно имеет неприемлемые социальные или экономические издержки. Поэтому Системы должны подвергаться регулярному надзору со стороны регулирующих органов и удовлетворять различным требованиям соответствия, в том числе информационной безопасности. Чтобы обеспечить качество и снизить риски, Ведомства полагаются на системы управления качеством, которые диктуют практику и процедуры и подтверждают безопасность и эффективность. Однако большинство систем основаны на традиционных подходах, которые часто предполагают проверку на соответствие в конце этапов реализации Системы, что могло приводить к невозможности ввода в эксплуатацию системы, к переработкам уже реализованных компонентов или системы в целом. В отличие от этого, бережливая система управления качеством делает деятельность по обеспечению соответствия частью регулярного производственного потока.

При подготовке релиза Системы необходимо учитывать следующие актуальные нормативные документы, определяющие требования к обеспечению безопасности государства в сфере, связанной с созданием, хранением, передачей и преобразованием информации [2, 3, 5, 7]:

1. Федеральный закон от 26 июля 2017 г. № 187-ФЗ "О безопасности критической информационной инфраструктуры Российской Федерации";

2. Приказ ФСТЭК России от 25 декабря 2017 г. № 239 "Об утверждении требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации" (в ред. приказа ФСТЭК России от 26 марта 2019 г. № 60);
3. ГОСТ Р 57628–2017 "Информационная технология (ИТ)";
4. ГОСТ Р 57628-2017, ГОСТ серии 15408, в части разработки профилей защиты и критериев оценки безопасности ИТ;
5. ГОСТ Р 51583-2014 в части создания АСЗИ, если для определяется законодательством Российской Федерации или решением ее обладателя;
6. НПА в части обеспечения безопасности обработки информации, содержащей государственную тайну;
7. Приказ ФСТЭК № 77 от 29 апреля 2021 г., определяющий порядок аттестации ГИС;
8. ГОСТ Р 56939, ГОСТ Р 58412 в части безопасной разработки ПО;
9. Федеральный закон от 27 июля 2006 года № 152-ФЗ "О персональных данных" (с изменениями от 2.07.2021).

Виды нарушений, касающихся самой информации и средств ее обработки, представлены в следующих статьях УК РФ: ст. 272. "Неправомерный доступ к компьютерной информации"; ст. 273. "Создание, использование и распространение вредоносных компьютерных программ"; ст. 274. "Нарушение правил эксплуатации средств хранения, обработки или передачи компьютерной информации и информационно-телекоммуникационных сетей"; ст. 274.1. "Неправомерное воздействие на критическую информационную инфраструктуру Российской Федерации".

Также в Кодексе об административных правонарушениях (КоАП РФ) предусмотрен перечень общественно опасных деяний, связанных с разглашением либо сокрытием информации.

5. БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

К 1980 году была сформирована производственная модель компании Toyota [82, 83], использующая принцип поставки «точно вовремя», на котором построено изготовление автомобилей в компании, которая стала называться бережливым производством.

Бережливое производство (от англ. lean production) — концепция управления производственным предприятием [34], ориентированная на создание продукта (услуги) путем формирования непрерывного потока создания ценности с охватом всех процессов организации и их постоянного совершенствования через вовлечение персонала и устранение всех видов потерь [15, 25].

Бережливый подход к созданию программного обеспечения рассматривает методы итерационной разработки (см. раздел 3) как методы, доказавшие свою эффективность [50, 66, 74]. Однако данный подход предоставляет более широкую перспективу, позволяющую гибким методам быстро развиваться.

Во-первых, бережливый подход позволяет держать под контролем всю цепочку создания ценности (от концепции до реализации), а также выявлять все случаи потерь и задержек, имеющих место до и после разработки кода.

Во-вторых, он создает управленческую среду, делающую возможным развитие гибких подходов к разработке программного обеспечения.

В-третьих, бережливый подход предоставляет набор доказавших свою эффективность принципов, которые каждая организация может использовать для внедрения предлагаемых средств, приемов и методов в своей уникальной среде и с учетом своих возможностей.

Бережливая разработка программного обеспечения, как методология разработки программного обеспечения, использующая методы концепции бережливого производства, основана на следующих принципах:

а. исключение потерь;

б. встроенное качество;

в. акцент на обучении и создании знания;

г. предельно отсроченное принятие решений;

д. предельно быстрая доставка продукта;

е. мотивация команды;

ж. оптимизация целого.

5.1. Исключение потерь

Процесс разработки программного обеспечения, основанный на принципах бережливости, фокусируется на сокращении сроков выполнения заказа путем выявления и ликвидации всех потерь.

Все, что мы делаем, если это не работает на то, чтобы наиболее полным образом удовлетворить требования заказчика, является источником непроизводительных затрат. Поскольку непроизводительная затрата не увеличивает потребительскую ценность, первый шаг в ликвидации подобных затрат состоит в том, чтобы понять, что собой представляет потребительская ценность. При этом нужно учитывать, что с одной стороны — клиент часто сам не до конца понимает, чего он хочет., а с другой — после того как клиент начинает работать с созданным программным обеспечением, представление о том, что ему нужно, обычно меняется.

Потерями считается всё, что не добавляет ценности для потребителя. В частности, излишняя функциональность, ожидание (паузы) в процессе разработки, нечёткие требования, бюрократизация, медленное внутреннее сообщение. Исходя из понятия ценности для клиента, к потерям при создании программного обеспечения следует отнести:

1. Задержку выполнения заказа.
2. Частично выполненную работу. В производстве источником непроизводительных затрат являются запасы (деталей, узлов и материалов), которыми необходимо управлять (перемещать, складировать, вести их учет и т.п.). Поэтому одна из целей производства — поддерживать запасы на минимальном уровне. Запасами в процессе разработки программного обеспечения является частично выполненная (или незавершенная) работа. Незавершенному ПО присущи все пороки, свойственные запасам в производстве (неиспользуемый программный код теряется, устаревает, содержит проблемы с качеством и «омертвляет» вложенные средства). Более того, большинство рисков, связанных с созданием ПО, кроется как раз в незавершенной работе.
3. Переработка созданного ранее программного обеспечения. Значительная доля потерь в разработке программного обеспечения связана с необходимостью переделывать то, что уже было сделано. Например, нередко приходится переделывать уже созданное ПО из-за того, что пока программное обеспечение создавалось, изменились требования клиента. Это достаточно типичная ситуация, и от 30 до 50% требований, сформулированных задолго до начала разработки ПО,

к концу процесса обычно изменяются. Также необходимость переделывать уже сделанную работу часто возникает, когда тестирование производится некачественно или со значительной задержкой. Если при разработке в программный код был внесен дефект, который сразу не был обнаружен и устранен, созданный после этого код (писавшийся с учетом имеющегося дефекта), когда дефект будет устранен, может оказаться неработоспособным, и его придется переделывать.

4. Повторное выполнение ранее сделанной работы. Указанные выше случаи часто являются только предтечей еще более крупных потерь, связанных с необходимостью повторного выполнения уже сделанной работы, например, когда имеет место отложенная (или небрежная) интеграция.
5. Избыточные функциональные возможности. Только около 20% функциональных возможностей в типичном пользовательском программном обеспечении используются регулярно, и около двух третей возможностей используется редко. Это не относится к используемым неявно функциям, например, обеспечению информационной безопасности. Речь идет о функциях, особой необходимости, которых нет изначально. Добавление избыточных функциональных возможностей в программное обеспечение обходится чрезвычайно дорого. Они увеличивают сложность программного кода, затрудняя его последующее совершенствование.

Право на существование имеют только те функциональные возможности, стоимость создания которых ниже производимой ими потребительской ценности. Выпуск продукта, обладающего минимально необходимым набором функциональных возможностей, ни больше, ни меньше, демонстрирует, что компания по-настоящему понимает, чего хотят пользователи.

Независимо от того, разрабатывается ли программное обеспечение на заказ или создается в качестве продукта для продажи на рынке, идеальный подход состоит в том, чтобы разделить это программное обеспечение на минимально-полезные наборы функций и поставлять их по одному набору за раз, начиная с наиболее нужных (или обеспечивающих наибольшую отдачу). Минимально-полезный набор функций — это тот, который помогает пользователям выполнить лучше некоторую полезную часть их работы. Ввод в эксплуатацию у заказчика (deployment) минимально-полезных наборов функций (в рамках проекта создания программного обеспечения на заказ) позволяет заказчикам начать использовать это программное обеспечение гораздо раньше. Поскольку эти наборы функций начинают давать прибыль раньше, компания может инвестировать меньше денег и, как правило, получать больше прибыли в течение срока использования системы. С технической точки зрения разработка минимально-полезных наборов функциональных возможностей (когда это делается

правильно) целесообразна, поскольку в этом случае программный код создается порциями и упрощается с каждым новым набором.

5.2. Встроенное качество

Встроенное качество является основным принципом бережливого производства программного обеспечения и направлено на снижение задержек выполнения работ, связанных с переделками и исправлением дефектов [50].

Встроенное качество включает следующие основные аспекты:

1. Создание потока функциональности;
2. Обеспечение качества архитектуры;
3. Обеспечение качества кода;
4. Обеспечение системного качества;
5. Обеспечение качественного релиза (раздел 4.3).

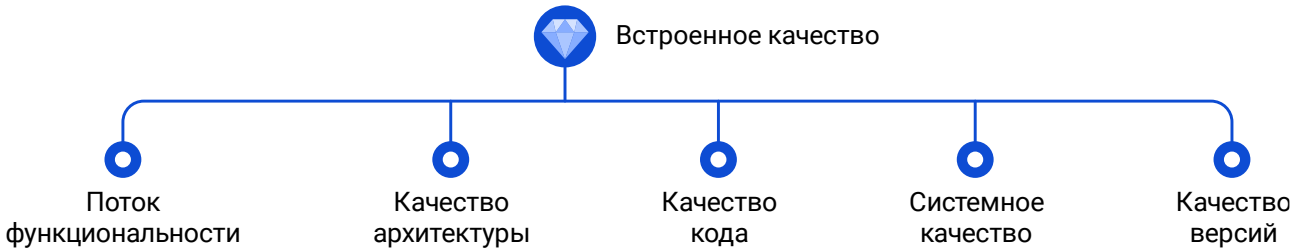

Рисунок 34. Основные аспекты встроенного качества

Термин «встроенное качество» (ГОСТ 57 522 — 2017) предусматривает оценку исполнителем показателей процессов выполнения работ с целью снижения рисков несоответствий на приемлемо низком уровне. Деятельность по встроенному качеству необходимо проводить на всех этапах производства продукта (оказания услуги) и, в первую очередь, на этапах проектирования продукции и/или производственных процессов. Также должна проводиться специальная организация операций контроля и аудита производственных процессов для поддержания установленной скорости потока создания ценности. При этом, обеспечение встроенного качества существенно уменьшает необходимость в процедурах контроля качества, позволяет передать функцию контроля качества непосредственно исполнителям и реализовать принцип «3 НЕ» — не делай, не передавай, не принимай дефектную продукцию (результат услуги).

5.2.1. Обеспечение качества архитектуры

Архитектура Системы в конечном итоге определяет, насколько хорошо Система может поддерживать текущие и будущие потребности конечных пользователей (клиентов). Качество архитектуры облегчает реализацию как функциональных, так и нефункциональных требований, а также упрощает тестирование Системы [50, 75].

Поскольку требования к Системе со временем могут изменяться, проектные и архитектурные решения, заложенные на начальном этапе создания Системы, также могут изменяться. Традиционные процессы, которые заставляют не изменять принятые проектные решения, могут привести к снижению качества разработки и конечной версии Системы в условиях изменяющихся требований. Определение наилучшего проектного решения требует знаний, полученных в ходе экспериментов, моделирования, прототипирования и других исследований. Это также требует подхода к проектированию на основе формирования и исследования альтернативных вариантов архитектуры, чтобы прийти к наилучшему решению.

Архитектура также влияет на тестируемость системы. Модульные компоненты, которые взаимодействуют через четко определенные программные интерфейсы, создают «швы», которые позволяют тестировщикам и разработчикам заменять дорогие или медленные компоненты тестовыми «заглушками».

Отправной точкой разработки крупной и сложной системы должно быть создание ее архитектуры (поддающейся разделению на подсистемы), которая бы позволила творческим коллективам работать (параллельно и независимо) над подсистемами, предоставляющими критические наборы функциональных возможностей. Объем каждой подсистемы должен быть таким, чтобы ее разработку смог осуществить один коллектив или несколько тесно связанных коллективов.

По мере того, как все больше становится очевидно, что толерантность к изменениям является главным ценным качеством для большинства систем программного обеспечения, поддерживающие инкрементную разработку архитектуры (такие, как ориентированные на сервис и основанные на компонентах), быстро заменяют монолитные архитектуры. Однако даже при использовании таких архитектур все равно существуют некоторые ограничения, особенно связанные с нефункциональными требованиями (безопасность, расширяемость и т.п.), которые необходимо учесть до начала процесса разработки. Целью хорошей архитектуры является возможность свести к минимуму такие необратимые решения и предоставить структуру, поддерживающую итерационную разработку (раздел 3).

Сама архитектура может и обычно должна создаваться (или наращиваться) постепенно (т.е. с использованием итерационного подхода)[14]. В результате

[14] Mark Denne, Jane Cleland-Huang, Software by Numbers, Prentice Hall, 2003

архитектура программного обеспечения, по сути, состоит из некоторых элементов, которые должны добавляться, когда они будут востребованы соответствующими наборами функциональных возможностей, находящимися в данный момент в разработке. Для успешных программных продуктов, срок жизни которых продолжается много лет, крупные архитектурные улучшения появляются по мере того, как для них находятся новые сферы применения, или возникает потребность в функциональных возможностях, которые невозможно было предусмотреть в оригинальной архитектуре.

5.2.2. Достижение качества кода

Все возможности Системы в конечном итоге реализуются программным кодом (или компонентами) Системы. При этом скорость и простота добавления новых возможностей зависят от того, насколько быстро и надежно разработчики смогут его модифицировать. Рекомендуется использовать практики модульного тестирования, разработки на основе встроенных тестов, парной работы и стандарты коллективной ответственности и кодирования.

Одной из важных существующих практик обеспечения качества программного кода предполагается «встраивание качества» в программный код, а не тестирование этого кода после его создания. Сегодня существуют средства, позволяющие устранять дефекты из программного кода в ходе его создания. Группа разработчиков, используя практики разработки на основе тестирования, создает блочные тесты и приемочные тесты до создания соответствующего кода. Разработчики интегрируют код и тесты в систему так часто, как это возможно (каждый час или около того), и инициируют запуск тестовой нагрузки, чтобы убедиться, что со времени предыдущей проверки дефекты в коде не появились. Если тест не выполняется, они не создают новый код, пока проблема не будет устранена. В конце дня запускается более сложная и полная тестовая нагрузка. В конце недели система проверяется с помощью еще более сложной тестовой нагрузки. В организациях, где используется такой подход, дефекты возникают редко, и их причины устраняются очень быстро.

Целью процесса тестирования (а также Команд разработки, разрабатывающих тесты и проводящих тестирование) должно быть предотвращение дефектов, а не поиск их. Разработка должна быть ориентирована на удаление дефектов из программного кода в процессе его создания (если угодно, это можно назвать встраиванием качества в код), а не на проверку качества кода позже.

Практики обеспечения встроенного качества включают модульное тестирование, разработку на основе тестов, парную работу и стандарты коллективной собственности на код.

Модульное тестирование и разработка на основе тестов. Практика модульного тестирования [60, 77] разбивает код на части и гарантирует, что каждая часть имеет автоматизированные тесты для его выполнения. Эти тесты запускаются автоматически после каждого изменения программного кода и позволяют разработчикам вносить быстрые изменения, будучи уверенными, что модификация не сломает другую часть Системы. Тесты также служат документацией и являются исполняемыми примерами взаимодействия с интерфейсом компонента, чтобы показать, как этот компонент должен использоваться. Разработка на основе тестирования направляет создание модульных тестов, указывая тест для изменения перед его созданием. Это заставляет разработчиков мыслить более широко о проблеме, включая крайние случаи и граничные условия перед реализацией. Лучшее понимание приводит к более быстрой разработке с меньшим количеством ошибок и меньшим количеством доработок.

При парной работе [31, 50, 65] два разработчика работают с одинаковыми изменениями на одной и той же рабочей станции. Один — пишет код, а другой — обеспечивает обзор и обратную связь в режиме реального времени. При этом разработчики часто меняют роли. Такое сопряжение создает и поддерживает качество, поскольку код будет содержать общие знания, перспективы и лучшие практики от каждого участника. Это также повышает и расширяет набор навыков для всей команды, поскольку товарищи по команде учатся друг у друга.

Стандарты коллективной собственности и кодирования. Коллективная собственность уменьшает зависимости между командами [50] и гарантирует, что любой отдельный разработчик или команда будет работать эффективно. Любой член команды может добавлять функциональность, исправлять ошибки, улучшать дизайн и проводить рефакторинг программного кода. Поскольку код не принадлежит одной команде или отдельному человеку, поддержка стандартов кодирования способствует согласованности, чтобы каждый сотрудник мог понять и поддерживать качество каждого программного компонента.

5.2.3. Системное качество

В то время как качество кода и архитектуры гарантируют, что системные артефакты могут быть легко поняты и изменены, качество системы подтверждает, что системы работают так, как ожидалось, и что все согласны с тем, какие изменения внести.

Системное качество обеспечивается:

- использованием производственных конвейеров *(Раздел 4.2)*, обеспечивающих непрерывную интеграцию и тестирование Системы;
- использованием практик разработки, основанной на поведении;

– модельно-ориентированной системной инженерией.

Практики разработки, основанной на поведении — это гибкий процесс разработки программного обеспечения, который поощряет сотрудничество между разработчиками, тестировщиками и представителями клиентов в рамках разработки программного обеспечения. Использование практики разработки, основанной на поведении, помогает разработчикам создавать правильное поведение с первого раза и уменьшает количество переделок и ошибок.

Разработка, основанная на поведении, требует согласование действий всех заинтересованных сторон в создании (развитии) Системы:

а. владелец продукта и члены команды договариваются о приоритетах для реализуемых функций;

б. заинтересованные стороны, ориентированные на клиента, понимают их потребности, а также относительную важность реализуемых командами требований к Системе;

в. заинтересованные стороны, ориентированные на развитие, понимают альтернативные решения и их технологическую осуществимость;

г. заинтересованные стороны, ориентированные на тестирование, рассматривают исключения, граничные случаи и граничные условия для реализуемого командами разработки поведения.

Модельно-ориентированная системная инженерия (МОСИ) — это практика разработки набора связанных системных моделей, которые помогают определять, проектировать и документировать разрабатываемую Систему. Эти модели обеспечивают эффективный способ изучения, обновления и передачи системных аспектов заинтересованным сторонам, при этом значительно уменьшая или устраняя зависимость от традиционных документов.

Также МОСИ обеспечивает высокоуровневое, полное представление обо всех требуемых функциях Системы и о том, как команды реализуют их.

Хотя модели не являются идеальным представлением системы, они обеспечивают знания и обратную связь быстрее и более экономически эффективно, чем только реализация. И они позволяют моделировать сложные взаимодействия в создаваемой Системе с соответствующей точностью для обучения. На практике инженеры используют модели для получения знаний и служат руководством для внедрения системы.

5.2.4. Обеспечение потока функциональности и качества релиза

Создание потока функциональности имеет основополагающее значение для всех команд разработки, поскольку оно описывает, как избежать ошибок, переделок и других причин, которые замедляют разработку.

Поток в процессе создания (развития) Системы обеспечивается использованием с помощью:

1. практики разработки на основе тестирования;
2. практики раннего тестирования;
3. использование технологии конвейера непрерывной разработки для обеспечения качества Релизов *(Разделы 4.2, 4.3)*.

Практика раннего тестирования предполагает вместо выполнения большого количества тестов в конце разработки выполнение многих тестов на ранних стадиях разработки кода. Раннее тестирование применяется как к функциональным требованиям, так и к нефункциональным требованиям, к производительности, надежности и другим требованиям к качеству Системы, указанных в ТЗ [16, 75].

Тесты должны выполняться быстро, и команды должны стремиться автоматизировать их. Поскольку более крупные, основанные на пользовательском интерфейсе, сквозные тесты выполняются намного медленнее, чем небольшие автоматизированные тесты, необходимо формировать сбалансированный набор тестов с большим количеством небольших быстрых тестов и меньшим количеством больших медленных тестов.

Поскольку тесты могут расти с течением времени (по мере роста объема кода), они начинают вносить задержки в работу команд. Поскольку настройка и выполнение полных наборов тестов может занять значительное время, команды могут создавать сокращенные наборы тестов и тестовых данных («дымовой тест»). Кроме этого, повышение скорости тестирования может быть обеспечено использованием более производительного оборудования в тестовых контурах. Для этого команды должны взаимодействовать с системной командой, чтобы сбалансировать скорость и качество тестирования *(Рисунок 35)*.

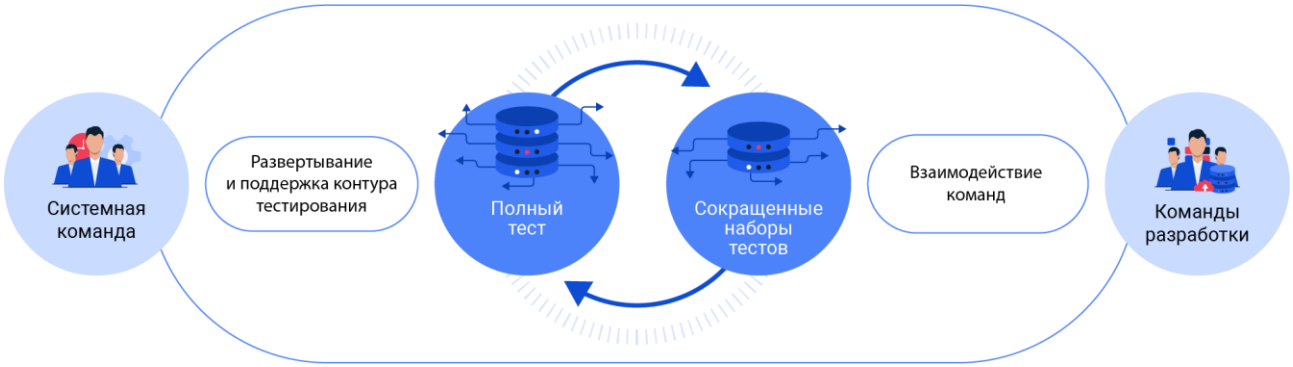

Рисунок 35. Оптимизация набора тестов

Разработка на основе тестирования (РНТ) — это философия и практика, которая включает в себя построение и выполнение тестов перед реализацией кода или компонента Системы *(Рисунок 36)* [17, 31, 75, 79]. Проверяя их на соответствие серии согласованных тестов, РНТ — практика гибкого тестирования — улучшает результаты системы, гарантируя, что реализация системы соответствует ее требованиям. РНТ, наряду с поведенческой разработкой, является частью подхода «сначала тест» для достижения встроенного качества. Написание тестов сначала создает более сбалансированный портфель тестов с множеством быстрых автоматизированных тестов разработки и меньшим количеством медленных, ручных, сквозных тестов.

Рисунок 36. Разработка на основе тестирования

РНТ определяется следующими правилами:

1. Сначала напишите тест, убедившись, что разработчик понимает требуемое поведение. Это может быть новый тест или модификация существующего теста.
2. Запустите тест и наблюдайте, как он завершается ошибкой. Поскольку кода еще нет, это может показаться бессмысленным, но оно выполняет две полезные цели: проверяет работу теста, включая любые тестовые программы, и демонстрирует, как система будет вести себя, если код неправильный.
3. Напишите минимальный объем кода, необходимый для прохождения теста. В случае сбоя переработайте код или тест до тех пор, пока он не начнет проходить регулярно.
4. Продолжайте реализацию нового кода до тех пор, пока не пройдут все тесты. Этот шаг дает разработчику уверенность в том, что его изменения соответствуют текущим требованиям и не создали ошибку в другой части системы.
5. Рефакторинг по мере необходимости для обеспечения соответствия проекта изменяющимся требованиям (например, возникающий дизайн). Разработчики постоянно обновляют свои проекты, чтобы гарантировать, что меняющиеся требования и растущая кодовая база не приведут к низкому качеству кода.

5.3. Акцент на обучении и создании знания

Разработка программного обеспечения — это процесс создания знания. Хотя архитектурные решения могут быть разработаны до начала создания кода, подтверждение (или не подтверждение) правильности этой архитектуры имеет место всегда после начала разработки. На практике реальная архитектура программного обеспечения всегда материализуется в процессе разработки, даже если подробный документ, описывающий эту архитектуру, был составлен предварительно. При предварительном определении архитектуры невозможно учесть ни всей сложности, с которой приходится сталкиваться в процессе ее реализации, ни обратной связи с процессом разработки. Кроме этого, заданная предварительно и подробно архитектура не чувствительна к обратной связи с заказчиками. Процесс разработки, направленный на создание знания, предполагает, что архитектура программного обеспечения возникнет во время создания программного кода, и исключает затраты на его предварительное определение.

Компании, долгое время демонстрирующие великолепные результаты в создании программного обеспечения, имеют одну общую особенность: они генерируют новое знание в процессе регулярного экспериментирования и сохраняют это знание в виде, удобном для доступа к нему в более крупных организациях. Такие компании аккумулируют не только явное знание, они находят способы неявное знание представить в более определенном виде и сделать его частью базы знаний организации. В этих компаниях существует понимание, что, хотя приобретение опыта при создании программного продукта важно, сохранение этого знания в виде, удобном для использования в процессе создания новых продуктов, еще важнее.

Исследователями определены четыре подхода, ведущие к созданию успешного программного обеспечения [89]:

1. Ранний релиз с минимумом функциональных возможностей с тем, чтобы потребители могли его оценить и выразить свое мнение, пожелания и претензии.
2. Ежедневный выпуск сборок, их тестирование и учет (в дальнейшей разработке) результатов тестирования.
3. Наличие коллектива и (или) лидера с достаточным опытом и развитой интуицией, позволяющих принимать правильные решения.
4. Использование модульной архитектуры, поддерживающей возможность добавления новых функций.

Важно организовать процесс разработки, который поощряет систематическое и целенаправленное приобретение опыта, но также важно систематически совершенствовать этот процесс. Иногда в попытках создать «стандартный процесс» реальные процессы разработки программного обеспечения ограничиваются рамками документации, что затрудняет коллективам разработчиков совершенствование этих процессов. В организации, где внедрены принципы бережливости, знают, что необходимо постоянно совершенствовать используемые процессы, поскольку в комплексной среде проблемы неизбежны.

Каждая аномалия должна инициировать поиск источника, вызвавшего проблему, а также поиск средств ее решения, а затем соответствующие изменения должны быть внесены в процесс разработки, чтобы предотвратить возникновение этой проблемы вновь. Усилия по совершенствованию процесса разработки должны стать стремлением и ответственностью коллектива разработчиков, и в каждом коллективе должно выделяться время специально для работы над совершенствованием этого процесса на регулярной основе.

5.4. Предельно отсроченное принятие решений

Принцип предполагает, что решение следует принимать не на основе предположений и прогнозов, а после появления существенных фактов.

В процессе создания системы следует избегать принятия решений, которые будет трудно в последствии изменить. Необходимо, чтобы в определенных точках процесса (там, где, скорее всего, потребуется вносить изменения) сохранялась возможность выбора того или иного варианта. В условиях неопределенности, особенно когда она сопровождается сложностью, наиболее успешный подход заключается в том, чтобы экспериментировать с различными вариантами, выполняя **параллельную проработку альтернатив** и откладывая принятие решения до последнего момента.

5.5. Предельно быстрая доставка продукта клиенту

Принцип предполагает ликвидацию непроизводительных затрат, что обеспечивает в свою очередь сокращение издержек на разработку. При этом скорость невозможна без высокого качества, что в свою очередь требует выстраивания производственного процесса таким образом, чтобы качество было встроено в создаваемый продукт. Кроме этого, чтобы обеспечить быстрое выполнение заказов, необходимо хорошо понимать потребности клиентов, для этого разработку рекомендуется выполнять короткими циклами с получением обратной связи от клиентов по результатам каждой итерации.

5.6. Мотивация команды

Принцип предполагает управление коллективами разработчиков, основанный на четырех предпосылках:

а. разработка продукции под руководством лидера;

б. технические специалисты высшей квалификации;

в. планирование и контроль, основанные на ответственности.

Уважение к людям означает, что коллективу дается общий план и определяются реальные задачи, а дальше коллективу предоставляется свобода действий в их выполнении. Работающие по этому принципу Команды разработки являются самоорганизующимися кросс-функциональными командами.

5.7. Оптимизировать целое

Принцип предполагает ориентацию процесса разработки на поставку завершенного продукта, ориентированного на удовлетворение потребностей клиента.

Процесс разработки программного обеспечения известен своей тенденцией к оптимизации отдельных частей системы или стадий процесса.

Бережливое производство позволяет оптимизировать поток создания ценности с момента принятия заказа и до поставки готового продукта и удовлетворения запросов клиента. Если организация сосредотачивается на оптимизации отдельных этапов создания продукта, а не на потоке создания ценности в целом, можно почти гарантировать, что процесс предоставления ценности будет неэффективным.

Для внедрения указанных принципов в процессы разработки необходимо участие и целостное видение процессов руководством организации, выполнение мероприятий по стандартизации, а также принятие и разделение всеми специалистами Команд разработки принципов бережливости.

6. ДИЗАЙН-МЫШЛЕНИЕ

6.1. Общие положения

Дизайн-мышление — это методология решения инженерных и деловых задач, основывающаяся на творческом подходе [29, 45]. Главной особенностью дизайн-мышления, в отличие от аналитического мышления, является не критический анализ, а творческий процесс, в котором порой самые неожиданные идеи ведут к лучшему решению проблемы.

Первые упоминания дизайн-мышления можно встретить в конце 1950-1960 годов, но они сосредоточены в области архитектуры и машиностроения. В 1969 году ученый и лауреат нобелевской премии по экономике Герберт Саймон предложил множество идей [52], включая быстрое прототипирование продуктов, тестирование с помощью наблюдения и иные концепты, которые составляют ядро дизайн-мышления, а также легли в основу формирования этапов стандартного процесса дизайн-мышления *(Рисунок 37)*:

1. определение проблемы;
2. исследование;
3. формирование идей;
4. прототипирование;
5. выбор лучшего решения проблемы (лучшей идеи);
6. внедрение работающего продукта;
7. оценка результатов и разработка предложений по изменению (развитию) продукта.

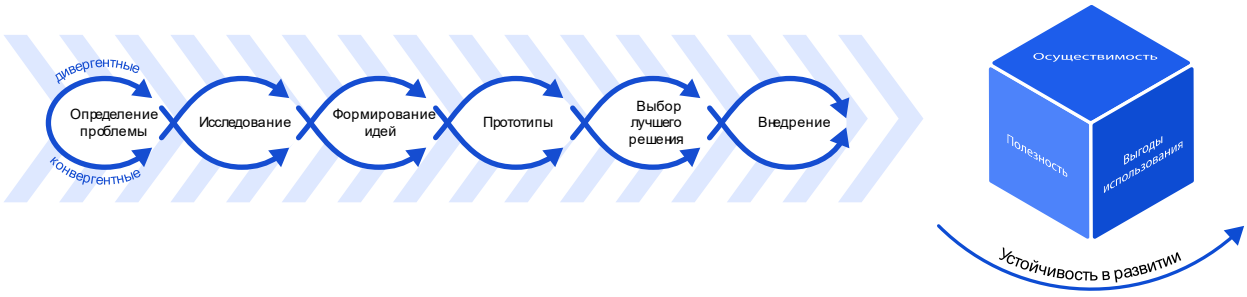

Рисунок 37. Концепция дизайн-мышления

Дизайн-мышление — ориентированный на клиента процесс разработки, который создает прибыльные и устойчивые на протяжении всего жизненного цикла продукты.

Подход к проектированию с использованием дизайн-мышления выходит за рамки традиционного акцента на особенностях и функциях предлагаемого продукта. Вместо этого он подчеркивает понимание проблемы, которую необходимо решить, контекста, в котором это решение будет использоваться, а также эволюции решения.

Традиционные каскадные подходы к разработке продукта являются последовательными: сначала определяются требования; затем решения проектируются, создаются и поставляются на рынок. Основное внимание, как правило, уделяется наиболее очевидным проблемам. Часто успех определяется внедрением решения, отвечающего требованиям, а не потребностям пользователя, в результате чего продукты и услуги с непригодными для использования или игнорируемыми функциями не соответствуют запросам пользователей.

Дизайн-мышление представляет собой ориентированный на клиента подход к разработке продуктов, в котором применяются дивергентные[15] и конвергентные[16] методы для понимания проблемы, создания продукта и предоставления его пользователям.

Дизайн-мышление сфокусировано на следующих основных свойствах продукта:

1. **Полезность —** Нужен ли клиентам и пользователям продукт?
2. **Осуществимость** — Можем ли мы создать продукт?
3. **Жизнеспособность —** Создает ли то, как мы создаем и предлагаем продукт пользователям, больше ценности, чем затрат на его создания? Например, на коммерческом предприятии мы прибыльны?
4. **Устойчивость в развитии** — Можем ли мы развивать наш продукт так, чтобы он соответствовал изменяющимся ожиданиям пользователей?

В процессе прохождения этих этапов формулируются проблемы, задаются правильные вопросы, придумываются идеи и выбираются лучшие решения. При этом данные этапы не являются линейными — разные этапы можно проходить одновременно и возвращаться к определенным этапам при необходимости.

6.2. Определение проблемы

Определение проблемы — это самый первый и самый важный этап дизайн-мышления, поскольку ошибочное определение проблемы может привести к тому, что в результате будет получено решение не той проблемы, которую нужно было решить.

[15] Дивергентный метод- поиск множества альтернативных решений проблемы.
[16] Конвергентный метод - точное использование инструкций по решению задачи.

Также для определения проблемы нужно сформирование понимание потребностей клиентов, а также сформулировать критерий достижения результата (например, ответить на вопрос, что является успешным результатом создания продукта).

6.3. Исследование

Второй этап в дизайн-мышлении — исследование — начинается с обзора истории проблемы: какие решения создавались ранее и какие результаты при этом были получены. Это помогает избежать «изобретения велосипеда» и ошибок, которые были совершены прежде

На этом этапе важно взаимодействие с конечными пользователями с целью выявления понимания проблем конечными пользователями (клиентами) и, возможно, идей о путях ее решения. Осуществляется сбор и исследование пользовательского опыта, для чего:

а. проводится интервьюирование или анкетирование пользователей;

б. используются практики и методы непрерывного изучения спроса пользователей, включая практики эмпатии, прогулки «Гемба», построения клиентских путей и другие;

в. осуществляется выявление проблем пользователей через мониторинг и изучение пользовательской аудитории.

Иногда наиболее эффективным способом получить информацию о проблеме будет наблюдение, поскольку в реальности конечные пользователи могут ведут себя иначе, чем они рассказывают при интервьюировании.

6.4. Формирование идей

На этом этапе необходимо собрать полученную на предыдущих стадиях информацию, чтобы формально зафиксировать потребности ваших пользователей, после чего, как правило, проводится процесс мозгового штурма. Главная задача мозгового штурма — придумать как можно больше самых разных альтернативных **идей, решающих проблему** (сформировать гипотезы). При этом недопустимо останавливаться на одной идее, даже если она представляется релевантной.

Практика мозгового штурма предполагает участие нескольких человек, работающих совместно, ориентировочно в течение одного дня. При этом высказываемые во время мозгового штурма идеи нельзя критиковать.

6.5. Прототипирование

После формирования перечня альтернативных идей решения проблемы необходимо провести их анализ с целью улучшения или объединения близких по смыслу идей.

Рекомендуется получение обратной связи от конечных пользователей (клиентов) для того, чтобы внести соответствующие изменения и уточнения в альтернативные варианты, после чего создать несколько работающих прототипов продукта, обеспечивающих решение проблемы клиента.

6.6. Выбор лучшего решения проблемы

Для выбора наилучшей идеи, дающей решение проблемы, возможно использование обратной связи от конечных пользователей или применение формальных методов на базе экспертных оценок, например, описанных в *Разделе 7.1. данного Приложения.*

6.7. Внедрение работающего продукта

На этом этапе осуществляется создание и внедрение работающего продукта, реализующего лучшею идею.

6.8. Оценка результатов

Создание и внедрение продукта — это не последний этап в процессе дизайн-мышления. Далее требуется организация процедур мониторинга с целью оценки качества решения первоначально определенной проблемы в соответствии с выбранным критерием и/или степени удовлетворенности клиента. На основе мониторинга принимаются решения о доработке или изменении продукта.

7. ОПРЕДЕЛЕНИЕ ПРИОРИТЕТОВ ЗАДАЧ

7.1. Метод анализа иерархий

Метод анализа иерархий [56, 57] предполагает определение цели выполнимых работ, критериев оценки достижения цели и набора возможных альтернатив с последующим построением оценок вкладов альтернатив в достижение цели:

1. На первом этапе методом экспертных оценок определяют веса критериев относительно цели.
2. На втором этапе методом экспертных оценок определяют веса альтернатив относительно каждого критерия.
3. На третьем этапе вычисляют приоритеты реализации функций относительно цели.

В общем случае последовательность определения приоритетов следующая:

1. Определить проблему, например, оценить приоритеты решаемых задач.
2. Построить иерархию, начиная с вершины (цели — с точки зрения управления), через промежуточные уровни (критерии, от которых зависят последующие уровни) к самому нижнему уровню (который обычно является перечнем альтернатив / решаемых задач).
3. Построить множество матриц парных сравнений[17] для каждого из нижних уровней — по одной матрице для каждого элемента верхнего уровня иерархии. Этот элемент называют направляемым по отношению к элементу, находящемуся на нижнем уровне, так как элемент нижнего уровня влияет на расположенный выше элемент. Элементы любого уровня сравниваются друг с другом относительно их воздействия на направляемый элемент. Таким образом, получаем квадратную матрицу суждений. Попарные сравнения проводятся в терминах доминирования одного из элементов над другим. Эти суждения затем выражаются в целых числах от 1 (равная важность) до 9 (очень сильное превосходство). Если элемент А доминирует над элементом Б, то клетка, соответствующая строке А и столбцу Б, заполняется целым числом, а клетка, соответствующая строке Б и столбцу А, заполняется обратным к нему числом (дробью). Если элемент Б доминирует над элементом А, то происходит обратное: целое число ставится в позицию Б, А, а обратная величина автоматически в позицию А, Б. Если считается, что А и Б одинаковы, в обе позиции ставится единица.

[17] На этапе 3 для получения каждой матрицы требуется n(n — 1)/2 суждений (при каждом парном сравнении автоматически приписываются обратные величины), где n – количество альтернатив

4. После проведения всех парных сравнений и ввода данных по собственному значению нужно определить их согласованность. Все измерения, включая те, в которых используются приборы, подвержены погрешностям измерений, а также погрешностям из-за неточностей в самом измерительном приборе. Эти погрешности могут привести к несогласованным выводам. Например, при взвешивании предметов измерения могут показать, что А тяжелее, чем Б, Б тяжелее, чем В, однако В тяжелее, чем А. В частности, это может случиться, когда веса предметов А, Б и В близки, а прибор недостаточно точен, чтобы их различить. Отсутствие согласованности может быть серьезным ограничивающим фактором для исследования некоторых проблем. Для улучшения согласованности можно рекомендовать поиск дополнительной информации и пересмотр данных, использованных при построении шкалы.

   Однако совершенной согласованности при измерениях даже с точными инструментами трудно достичь на практике. Нужен способ оценки степени согласованности при решении конкретной задачи. Вместе с матрицей парных сравнений мы имеем меру оценки степени отклонения от согласованности. Когда такие отклонения превышают установленные пределы, тому, кто проводит суждения, следует перепроверить их в матрице.

   Индекс согласованности в каждой матрице и для всей иерархии может быть приближенно получен следующим образом. Сначала суммируется каждый столбец суждений, затем сумма первого столбца умножается на величину первой компоненты нормализованного вектора приоритетов, сумма второго столбца — на вторую компоненту и т. д. Затем полученные числа суммируются. Таким образом можно получить величину, обозначаемую L_{max}. Для индекса согласованности имеем:

$$\textbf{ИС} = (L_{max} - n)/(n - 1),$$

   где n — число сравниваемых элементов.

   Далее нужно сравнить полученную величину ИС с той, которая получилась бы при случайном выборе количественных суждений из шкалы 1/9, 1/8, 1/7,.. , 1, 2..9. Ниже даны средние согласованности для случайных матриц разного порядка.

Размер матрицы	1	2	3	4	5	6	7	8	9	10
Случайная согласованность	0	0	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,49

   Если разделить полученное значение ИС на число, соответствующее случайной согласованности матрицы того же порядка, получим отношение согласованности (ОС). Величина ОС должна быть порядка 10% или менее, чтобы быть приемлемой.

В некоторых случаях можно допустить 20%, но не более. Если ОС выходит из этих пределов, то участникам нужно исследовать задачу и проверить свои суждения.

5. Провести этапы 3, 4 и 5 для всех уровней и групп в иерархии.
6. Вычислить обобщённые (глобальные) приоритеты альтернатив как сумму произведений локальных приоритетов (по отношению к каждому отдельному критерию) и величину приоритета этого отдельного критерия по отношению к цели исследования.

$$\mathbf{P_i} = \sum_j p_{ij} k_j$$

P_i — глобальный приоритет i-й альтернативы,

p_{ij} — локальный приоритет i-й альтернативы по j-му критерию,

k_j — приоритет j-го критерия.

7. Определить согласованность всей иерархии, перемножая каждый индекс согласованности на приоритет соответствующего критерия и суммируя полученные числа. Результат затем делится на выражение такого же типа, но со случайным индексом согласованности, соответствующим размерам каждой взвешенной приоритетами матрицы (приемлемым является ОС около 10% или менее). В случае если ОС > 10%, возможно пересмотреть способ, следуя которому задаются вопросы при проведении парных сравнений. Если это не поможет улучшить согласованность, то, вероятно, задачу следует более точно структурировать, т. е. сгруппировать аналогичные элементы под другими значащими критериями.

При проведении оценок следует иметь в виду все сравниваемые элементы, чтобы сравнения были релевантными. Нетрудно убедиться в том, что для проведения обоснованных численных сравнений не следует сравнивать более чем 7 ± 2 элементов. В таком случае маленькая погрешность в каждой относительной величине меняет ее не очень значительно. Если это так, то каким образом работать с более широким классом объектов? Ответ таков: посредством иерархической декомпозиции. Элементы группируются (в качестве первой оценки) в сравниваемые классы приблизительно из семи элементов в каждом. Элемент с наивысшим весом в классе также включается в следующий класс элементов с большими весами и как своеобразный стержень между двумя классами придает однородность шкале. Процедура повторяется от одного класса к смежному классу, пока все элементы не будут взвешены соответствующим образом.

В некоторых задачах с большим числом альтернатив нам не всегда нужно проводить парные сравнения между ними. Вместо этого вводим субкритерии (например, высокий, средний, низкий) и устанавливаем важность этих субкритериев по отношению к критериям. Затем берем каждую альтернативу, проверяем, который из субкритериев описывает ее наилучшим образом, и принимаем приоритет этого субкритерия. Далее складываем все приоритеты для этой альтернативы, и, наконец, нормализуем величины альтернатив, чтобы получить их общий приоритет.

Метод анализа иерархий является универсальным (фактически на нем базируются другие, более «легкие» методы определения приоритетов), но для большого числа критериев и альтернатив требуется проведение ресурсоемких матричных вычислений, что затрудняет его использование без автоматизации.

7.2. Метод «Более ценная и короткая работа сначала»

Метод «Более ценная и короткая работа сначала» (WSJF) — это метод приоритезации задач, предложенный Дональдом Рейнертсоном в книге «The Principles of Product Development Flow» [66] и используемый, в частности, в фреймворке SAFe.

Смысл метода в том, чтобы наиболее ценные для пользователей и быстровыполнимые задачи брать с более высоким приоритетом.

Формула расчёта показателя WSJF выглядит так:

WSJF = Стоимость задержки / Продолжительность

WSJF = Стоимость задержки / Сложность задачи

Стоимость задержки определяется следующим образом:

Стоимость задержки = Ценность для пользователей + Временная критичность + (Снижение рисков или Новые возможности).

Оценка показателей, составляющих Стоимость задержки, осуществляется методом экспертных оценок, при этом эксперты, оценивая показатели отвечают на вопросы:

1. Ценность для пользователей: насколько сильно просят об этом пользователи? Какой потенциально негативный эффект будет, если это выполнить позже, а не раньше?

2. Временная критичность: задерживает ли функция реализацию других функций? Нужно ли это выпустить к определенной дате? Есть ли риск того, что опоздание с реализацией функции обесценит ранее проделанную работу?
3. Снижение рисков: снижает ли реализация функции какие-то риски? Будет ли это позитивно влиять на качество в других областях? Будет ли эффект сиюминутным или долгосрочным?
4. Новые возможности: откроет ли реализация функции новые возможности для продукта? Поможет ли расширить пользовательскую аудиторию?

Оценки проставляются путём сравнения задач относительно друг друга, от 1 до 21 из последовательности Фибоначчи (1, 3, 5, 8, 13, 21…). Полученные оценки суммируются и делятся на размер работы. Размер работы тоже оценивается числами Фибоначчи методом экспертных оценок.

	Ценность для пользователей	Временная критичность	Снижение риска/ Новые возможности	Стоимость задержки	Сложность задачи	WSJF	Приоритет
Задача **1**	1	3	13	17	2	8,50	1
Задача **2**	13	5	13	31	5	6,20	2
Задача **3**	5	1	8	14	3	4,67	3
Задача **4**	13	5	3	21	5	4,20	4
Задача **5**	13	8	13	34	13	2,62	5

Правила проставления оценок:

1. Оценивать задачи по одному столбцу за раз, указать единицу (минимальный размер) для наименьшего элемента, а затем оцените остальные задачи относительно наименьшей.
2. Следствие: должна быть минимум одна единица в каждой колонке.
3. Самое большое число по WSJF означает наиболее высокий приоритет.

Суммируя показатели в трех первых столбцах и деля на размер задачи, мы получаем значения показателя, сортируя которые от большего к меньшему, мы получаем приоритет задач.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Федеральный закон от 31 июля 2020 г. № 248-ФЗ «О государственном контроле (надзоре) и муниципальном контроле в Российской Федерации»;

[2] Федеральный закон от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации»;

[3] Постановление Правительства Российской Федерации от 6 июля 2015 г. № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных систем и дальнейшего хранения содержащейся в их базах данных информации»;

[4] Постановление Правительства Российской Федерации от 10 октября 2020 г. № 1646 «О мерах по обеспечению эффективности мероприятий по использованию информационно-коммуникационных технологий в деятельности федеральных органов исполнительной власти и органов управления государственными внебюджетными фондами»;

[5] Порядок организации и проведения работ по аттестации объектов информатизации на соответствие требованиям о защите информации ограниченного доступа, не составляющей государственную тайну», утвержденный приказом ФСТЭК России от 29 апреля 2021 г. № 77;

[6] Стандарт мониторинга оказания сервисов, утвержденный Протоколом заседания президиума Правительственной комиссии по цифровому развитию, использованию информационных технологий для улучшения качества жизни и условий ведения предпринимательской деятельности от 29 декабря 2021 г. № 50;

[7] Методические рекомендации по обеспечению разработки безопасного программного обеспечения на единой цифровой платформе «ГосТех»: утвержденные приказом Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации, 2022;

[8] Методические рекомендации по формированию концепции создания государственной информационной системы, утвержденные приказом Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации, 2022;

[9] Навигатор цифровой трансформации: Agile-подход в государственном управлении: электронное издание / под ред. Е.Г. Потаповой. — М.: РАНХиГС, 2019;

[10] Agile: практическое руководство, Project Management Institute, Inc., пер. с англ., Олимп–Бизнес; Москва; 2018;

[11] Белл Л., Брантон-Сполл М., Смит Р., Бэрд Д., Безопасность разработки в Agile-проектах. Обеспечение безопасности в конвейере непрерывной поставки, Пер. с англ. А. Слинкин. — М.: ДМК «Пресс», 2018;

[12] Блауберг И.В., Садовский В. Н., Юдин Э. Г., Системный подход в современной науке. Проблемы методологии системных исследований. — М.: Мысль, 1970;

[13] Вольфсон Б., Гибкое управление проектами и продуктами, -СПб.: Питер, 2014;

[14] Вехен Джульен, Безопасный DevOps. Эффективная эксплуатация систем. — СПб.: Питер, 2020;

[15] Вумек Джеймс П., Джонс Даниел Т. Бережливое производство. Как избавиться от потерь и добиться процветания вашей компании. Пер с англ., — М.: «Альпина Паблишер», 2011;

[16] Грегори Дж., Криспин Л., Agile-тестирование. Обучающий курс для всей команды; пер. с англ. Е. Кротовой, науч. ред. С. Виноградов. — М.: Манн, Иванов и Фербер, 2019;

[17] Грэхем Ли, Разработка через тестирование для iOS. Пер. с англ. Киселев А. Н. — М.: ДМК «Пресс», 2013;

[18] Деннинг Стивен, Эпоха Agile. Как умные компании меняются и достигают результатов пер. с англ. Ю. Гиматовой, науч. ред. А. Макарова. — М.: Манн, Иванов и Фербер, 2019;

[19] Джен Ким, Патрик Дебуа, Джон Уиллис, Джез Хамбл, Руководство по DevOps. Как добиться гибкости, надежности и безопасности мирового уровня в технологических компаниях, Пер. с англ. И. Лейко и И. Васильева, науч. ред. Н. Корытко. — М.: Манн, Иванов и Фербер, 2018;

[20] Дженнифер Дэвис, Кэтрин Дэниелс, Философия DevOps. Искусство управления IT, -СПб.: «Питер», 2017;

[21] Джефф Сазерленд. Scrum. Революционный метод управления проектами, Пер с англ., Манн, Иванов и Фербер (МИФ), 2014;

[22] Джим Калбах, Путь клиента: создаем ценность продуктов и услуг через карты путей, блюпринты и другие инструменты визуализации; пер. с англ. П. Миронова, науч. ред. М. Сташенко. — М.: Манн, Иванов и Фербер, 2022;

[23] Джозеф О'Коннор, Иан Макдермотт Искусство системного мышления. Необходимые знания о системах и творческом подходе к решению проблем. Пер с англ., Альпина Паблишер, 2013;

[24] Джон Уэлен, Дизайн пользовательского опыта. Как создать продукт, который ждут; пер. с англ. Э. Кондуковой ; науч. ред. М. Сташенко. — М.: Манн, Иванов и Фербер, 2021;

[25] Джордж Л. Майкл, «Бережливое производство + шесть сигм» в сфере услуг: Как скорость бережливого производства и качество шести сигм помогают совершенствованию бизнеса; Пер. с англ. — М.: Альпина Бизнес Букс, 2005;

[26] Дэвид Андерсон, Канбан. Альтернативный путь в Agile, Пер. с англ., — М.: Манн, Иванов и Фербер, 2017;

[27] Дэвид Дж. Андерсон и Энди Кармайкл, Канбан: краткое руководство. Пер. с англ., LeanKanban University, 2016;

[28] Деннинг С., Эпоха Agile. Как умные компании меняются и достигают результатов. Манн, Иванов, Фаббер (МИФ). 2019;

[29] Кемпкенс Оливер, Дизайн-мышление. Все инструменты в одной книге. Пер с англ., «Издательство «Эксмо», 2019;

[30] Кеннет С. Рубин, Основы Scrum. Практическое руководство по гибкой разработке ПО, Пер с англ. Диалектика. -М.: 2019;

[31] Кент Бек. Экстремальное программирование: разработка через тестирование. Пер с англ., — М.: И.Д. «Вильямс», 2017;

[32] Ким, Джин, Джез Хамбл, Патрик Дебуа и Джон Уиллис, Руководство по DevOps: как создать гибкость, надежность и безопасность мирового класса в технологических организациях. IT Революция Пресс, 2016;

[33] Клод Обри, Все о Scrum. Изучение, разработка, интеграция. Пер. с англ., Степанянц Е.Г, «Эксмо», 2021;

[34] Лайкер, Джеффри, Дао Toyota, 14 принципов менеджмента ведущей компании мира. 14 Management Principles from the World's Greatest Manufacturer. — М.: «Альпина Паблишер», 2011;

[35] Ленц М., Python: Непрерывная интеграция и доставка. Пер. с анг. А. Е. Мамонова, Д. А. Беликова. — М.: ДМК Пресс, 2020;

[36] Леффингуэлл Дин, Уидриг Дон, Принципы работы с требованиями к программному обеспечению. Унифицированный подход, Пер. с англ., — М.: И.Д. «Вильямс», 2002;

[37] Липаев В.В., Документирование и управление конфигурацией программных средств, -М.: Синтег, 1998;

[38] Липаев В.В., Обеспечение качества программных средств. Методы и стандарты, -М.: Синтег, 2001;

[39] Липаев В.В., Системное проектирование сложных программных средств для информационных систем, -М.: Синтег, 2002;

[40] Майк Барроуз, Канбан Метод. Улучшение системы управления. Пер с англ., — М.: Альпина Паблишер, 2020;

[41] Майк Кон «Agile: оценка и планирование проектов. Пер с англ., — М.: «Альпина Паблишер», 2018;

[42] Майк Кон Scrum: гибкая разработка ПО. Пер с англ. — М.: «Вильямс», 2011;

[43] Майк Кон. Пользовательские истории: гибкая разработка программного обеспечения. Пер с англ. Диалектика. -М.: 2019;

[44] Марк Лоффлер, Ретроспектива в Agile. Проверенные методы и инновационные подходы. Пер с англ. — М.: «Манн, Иванов и Фербер», 2020;

[45] Мартин Томич, Кара Ригли, Мейделин Бортвик, Насим Ахмадпур, Джессика Фроули, Придумай. Сделай. Сломай. Повтори. Пер с англ., Манн, Иванов и Фербер (МИФ), 2019;

[46] Месарович М., Мако Д., Такахара И., Теория многоуровневых иерархических многоуровневых систем, -М.: Мир, 1973;

[47] Макконнелл Стив, Еще более эффективный Agile, Пер. с англ., — СПб.: Питер, 2021;

[48] Перегудов Ф.И., Тарасенко Ф.П., Ведение в Системный анализ, — М.: Высшая школа, 1989;

[49] Петрушенко Л.А., Принцип обратной связи. Некоторые философские и методологические проблемы управления — М.: Мысль, 1967;

[50] Поппендик Мэри, Поппендик Том, Бережливое производство программного обеспечения: от идеи до прибыли, Пер. с англ. -М.: И.Д. «Вильямс», 2010;

[51] Сазерленд Джефф, Scrum. Революционный метод управления проектами, Пер. с англ.- М.: Манн, Иванов и Фербер; Москва; 2016;

[52] Саймон Г., Науки об искусственном (The Sciences of the Artificial), Пер. с англ., Мир, 1972;

[53] Саймон Хейворд, Agile-трансформация: готовый план перехода к гибкой бизнес-модели организации, Пер. с англ. Авдеева А.А., Издательство «Эксмо», 2021;

[54] Скрынник О. В., DevOps для ИТ-менеджеров: концентрированное структурированное изложение передовых идей. — М.: ДМК «Пресс», 2019;

[55] Стиллмен Эндрю, Грин Дженифер, Head First Agile. Гибкое управление проектами. Пер. с англ.,— СПб.: Питер, 2019;

[56] Саати Т., Принятие решений. Метод анализа иерархий: Пер. с англ.-М.: «Радио и связь», 1993;

[57] Саати Т., Кернс К., Аналитические планирование. Организация систем, -М.: Радио и связь, 1991;

[58] Тюкин И. Ю., Терехов В. А., Адаптация в нелинейных динамических системах, СПб.: ЛКИ, 2008;

[59] Херинг М., DevOps для современного предприятия, Пер. с анг. М. А. Райтмана. — М.: ДМК «Пресс», 2020;

[60] Хориков В., Принципы юнит-тестирования, — СПб.: «Питер», 2021;

[61] Цифровая трансформация отраслей: стартовые условия и приоритеты, Доклад к XXII Апр. международного научной конференции по проблемам развития экономики и общества, Москва, 13–30 апр. 2021 г., Национальный исследовательский университет «Высшая школа экономики». — М.: И.Д. Высшей школы экономики, 2021;

[62] Черняк Л., Адаптируемость и адаптивность, Открытые системы. СУБД, № 9, 2009;

[63] Ajay Reddy, Scrumban [R]Evolution, Getting the Most Out of Agile, Scrum, and Lean Kanban, Addison-Wesley Professional, 2015;

[64] Kent Beck, Extreme Programming Explained, First Edition September 29, 1999;

[65] Degrandis Dominica, Making Work Visible: Exposing time theft to optimize work & flow, IT Revolution Press, 2017;

[66] Donald G. Reinertsen, The Principles of Product Development Flow: Second Generation Lean Product Development Hardcover, Celeritas Pub, 2009;

[67] Esther Derby, Diana Larsen, Ken Schwaber. Agile Retrospectives: Making Good Teams Great. The Pragmatic Bookshelf, 2009;

[68] Henrik Kniberg, Scrum and XP from the Trenches. How We do Scrum — 2nd Edition. InfoQ, 2015;

[69] José Manuel Ortega, Implementing DevSecOps with Docker and Kubernetes, BPB Publications, 2022;

[70] Larman Craig, Agile and Iterative Development: A Manager's Guide 1st Edition, Addison-Wesley Professional, 2003;

[71] Larman Craig, Bas Vodde, Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum, Addison-Wesley Professional, 2008;

[72] Larman Craig, Bas Vodde, Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum, Addison-Wesley Professional, 2010;

[73] Larman Craig, Bas Vodde, Large-Scale Scrum: More with LeSS, Addison-Wesley Professional, 2016;

[74] Leffingwell Dean, Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise, Addison-Wesley Professional, 2011;

[75] Maurício Aniche, Effective Software Testing, Manning Publications Co., 2021;

[76] Mikael Krief, Learning DevOps. Second Edition., Packt Publishing, 2022;

[77] Osherove Roy, The Art of Unit Testing, Manning Publications, 2022;

[78] Saleem Siddiqui, Learning Test-Driven Development, O'Reilly, 2022;

[79] Skelton Mathew, Manuel Pais, Team Topologies, IT Revolution Press, 2019;

[80] Sudipta Malakar, Agile in Practice. Practical Use-cases on Project Management. Methods including Agile, Kanban and Scrum, BPB Publications, 2021;

[81] Schwaber Ken, Beedle Mike, Agile software Development with Scrum, Prentice Hall, 2002;

[82] Taiichi Ohno. Toyota Production System: Beyond Large-Scale Production, 1988;

[83] James P. Womack, Daniel T. Jones, Daniel Roos, The Machine That Changed the World: The Story of Lean Production, Free Press, 1990;

[84] Кев Зеттлер, Инструменты DevSecOps, обеспечивающие безопасность рабочих процессов DevOps. [Электронный ресурс], URL: https://www.atlassian.com/ru/devops/devops-tools/devsecops-tools ;

[85] SAFe 5.0 Glossary. Scaled Agile Framework Terms and Definitions. Scaled Agile, Inc. [Электронный ресурс], www.scaledagileframework.com ;

[86] Рогачев Сергей, Гибкая разработка пользовательской документации, [Электронный ресурс], https://rsn81.wordpress.com/2013/02/20/user-guide-agile-development ;

[87] The 2021 State of DevOps Report. [Электронный ресурс], https://puppet.com/resources/report/2021-state-of-devops-report ;

[88] Гибкая разработка в масштабе, [Электронный ресурс], https://www.atlassian.com/agile/agile-at-scale ;

[89] Alan MacCormack, Creating a Fast and Flexible Process: Research Suggests Keys to Success [Электронный ресурс] www.roundtable.com/MRTIndex/FFPD/ART-maccormack.html .

ПРИЛОЖЕНИЕ №2

к Методическим рекомендациям
по организации производственного процесса разработки
государственных информационных систем с учетом
применения итерационного подхода к разработке

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, ТЕРМИНОВ И ОПРЕДЕЛЕНИЙ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АС	—	автоматизированная система
АТ	—	архитектурные (технические) требования
ВПЦТ	—	ведомственная программа цифровой трансформации
ВЦ	—	важность цели
ГИС	—	государственная информационная система
ГКР	—	группа команд разработки
ГрТр	—	группа требований
КНР	—	конвейер непрерывной разработки
КР	—	команда разработки
МДВ	—	минимально достаточная версия системы
МОСИ	—	модельно-ориентированная системная инженерия
НФТ	—	нефункциональные требования
ПА	—	проверка и адаптация
ПАК	—	программно-аппаратный комплекс
ПО	—	программное обеспечение
ПП	—	постановление правительства
СМИБ	—	система менеджмента информационной безопасности
РНТ	—	разработка на основе тестирования
ТЗ	—	техническое задание
ТЭО	—	технико-экономическое обоснование
ФТ	—	функциональные требования
CD	—	Continuous Delivery (непрерывное развёртывание)
CE	—	Continuous Exploration (непрерывное исследование)
CI	—	Continuous Integration (непрерывная интеграция)
PDCA	—	цикл Деменга-Шухарта «Планируй (P) -Делай (D) — Проверяй (C)- Действуй (A)»
WSJF	—	Weighted Shortest Job First (наиболее важная и короткая работа сначала)
XP	—	eXtreme Programming (экстремальное программирование)

ПЕРЕЧЕНЬ ТЕРМИНОВ И ОПРЕДЕЛЕНИЙ

Термин	Определение	Источник
Архитектура Системы	Совокупность основных понятий или свойств системы в окружающей среде, воплощенной в ее элементах, отношениях и конкретных принципах ее проектирования и развития. Архитектура системы состоит из шести архитектурных представлений: а) архитектура деятельности; б) архитектура программных средств (программная архитектура); в) архитектура данных; г) интеграционная архитектура; д) инфраструктура информационной безопасности; е) технологическая архитектура.	ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011 «Национальный стандарт Российской Федерации. Системная и программная инженерия. Описание архитектуры». Типовое Соглашение о моделировании архитектуры государственной информационной системы
Архитектурная поддержка	Специальное программное обеспечение и техническая инфраструктура, которые в каждый момент времени позволяют реализовывать высокоприоритетную функциональность продукта без задержек и перепроектирования.	Типовое Соглашение о моделировании архитектуры государственной информационной системы
Бережливое производство	Концепция организации бизнеса, ориентированная на создание привлекательной ценности для потребителя путем формирования непрерывного потока создания ценности с охватом всех процессов организации и их постоянного совершенствования через вовлечение персонала и устранение всех видов потерь.	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь

Термин	**Определение**	**Источник**
Ведомственная программа цифровой трансформации (ВПЦТ)	Документ, содержащий мероприятия, направленные на поэтапную реализацию цифровой трансформации государственного органа, цели и соответствующие им показатели (включая их значения) результативности и эффективности системы госуправления, которые планируется достигнуть посредством цифровой трансформации государственного органа в очередном плановом периоде, включая сведения об источниках и объемах необходимого для этого финансового обеспечения	Постановление Правительства Российской Федерации от 10 октября 2020 г. № 1646 «О мерах по обеспечению эффективности мероприятий по использованию информационно-коммуникационных технологий в деятельности федеральных органов исполнительной власти и органов управления государственными внебюджетными фондами»
Визуализация (visualization)	Расположение всех инструментов, деталей, производственных стадий и информации о результативности работы производственной системы таким образом, чтобы они были четко видимы и чтобы каждый участник производственного процесса моментально мог оценить состояние системы.	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь
Встроенное качество	Методология приведения возможностей процессов и систем измерения в соответствие с требованиями потребителя к качеству продукции, в том числе предоставление доказательств выполнения данных требований. Термин "встроенное качество" предусматривает оценку исполнителем показателей возможностей процессов с целью снижения рисков несоответствий на приемлемо	ГОСТ 57 522 — 2017 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Руководство по интегрированной системе менеджмента качества и бережливого производства

Термин	Определение	Источник
	низком уровне, а также специальную организацию операций контроля и аудита производственных процессов для поддержания установленной скорости потока создания ценности.	
Государственная информационная система	Система, создаваемая в целях реализации полномочий государственных органов и обеспечения обмена информацией между этими органами, а также в иных установленных федеральными законами целях. Государственные информационные системы создаются, модернизируются и эксплуатируются с учетом требований, предусмотренных законодательством Российской Федерации о контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд либо законодательством Российской Федерации о государственно-частном партнерстве, о муниципально-частном партнерстве, законодательством о концессионных соглашениях, а в случаях, если эксплуатация государственных информационных систем осуществляется без привлечения средств бюджетов бюджетной системы Российской Федерации, в соответствии с иными федеральными законами.	Федеральный закон от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации»
Государственная услуга, предоставляемая федеральным органом исполнительной власти,	Деятельность по реализации функций соответственно федерального органа исполнительной власти, государственного внебюджетного фонда, исполнительного органа государственной власти субъекта Российской Федерации, а также органа	Федеральный закон от 27 июля 2010 г. № 210-ФЗ «Об организации предоставления государственных и муниципальных услуг»

Термин	Определение	Источник
органом государственного внебюджетного фонда, исполнительным органом государственной власти субъекта Российской Федерации, а также органом местного самоуправления при осуществлении отдельных государственных полномочий, переданных федеральными законами и законами субъектов Российской Федерации (Государственная услуга)	местного самоуправления при осуществлении отдельных государственных полномочий, переданных федеральными законами и законами субъектов Российской Федерации, которая осуществляется по запросам заявителей в пределах установленных нормативными правовыми актами Российской Федерации и нормативными правовыми актами субъектов Российской Федерации полномочий органов, предоставляющих государственные услуги.	
Государственный контракт	Договор, заключенный от имени Российской Федерации, субъекта Российской Федерации государственным заказчиком для обеспечения соответственно государственных нужд, муниципальных нужд.	п. 8 ст. 3 Федерального закона от 5 апреля 2013 г. № 44-ФЗ «О контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд»
Государственные органы	Органы государственной власти Российской Федерации, Органы государственной власти субъектов Российской Федерации и иный государственные органы, образуемые в соответствии с законодательством Российской Федерации, законодательством субъектов Российской Федерации.	Федеральный закон «Об обеспечении доступа к информации о деятельности государственных органов и органов местного самоуправления» от 09.02.2009 № 8-ФЗ

Термин	**Определение**	**Источник**
Государственный контроль (надзор), муниципальный контроль в Российской Федерации	Государственный контроль (надзор), муниципальный контроль осуществляются на основе управления рисками причинения вреда (ущерба), определяющего выбор профилактических мероприятий и контрольных (надзорных) мероприятий, их содержание (в том числе объем проверяемых обязательных требований), интенсивность и результаты.	Федеральный закон от 31 июля 2020 г. № 248-ФЗ «О государственном контроле (надзоре) и муниципальном контроле в Российской Федерации»
Группа команд разработки	Несколько Команд разработки, совместно работающих над созданием и развитием Системы итерациями фиксированной длительности (рекомендуется — 12 недель) — Инкрементами.	Термин определён настоящим документом
Группа требований	Перечень требований к системе, выполнение которых обеспечивает реализацию **одной или нескольких** возможностей системы.	Постановление Правительства Российской Федерации от 6 июля № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных систем и дальнейшего хранения содержащейся в их базах данных информации»
Демонстрация системы	Комплексное представление заинтересованным сторонам новых функций Системы, реализованных в рамках Инкремента.	Термин определён настоящим документом

Термин	**Определение**	**Источник**
Жизненный цикл продукта	1. Совокупность взаимосвязанных процессов, объединяемых в этапы, изменения состояния информационной системы от принятия решения о ее создании, формирования исходных требований к ней и до окончания ее эксплуатации (вывода из эксплуатации).	Федеральный закон от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации»
	2. Цикл, охватывающий четыре основных этапа продукта — «запуск», «рост», «зрелость» и «упадок» — и связанный с активным маркетингом продукта на рынке. (Иногда в основу жизненного цикла продукта закладывают процесс, состоящий из пяти этапов — запуска, роста, зрелости, насыщения и упадка).	ГОСТ Р 58537- 2019. УПРАВЛЕНИЕ ПРОДУКЦИЕЙ. Основные положения
Заинтересованная сторона	Индивидуум, команда, организация или их группы, имеющие интерес к Системе.	Типовое Соглашение о моделировании архитектуры государственной информационной системы
Инкремент	Выполнение видов работ по реализации группы требований, в том числе повторное выполнение в целях устранения выявленных недостатков.	Постановление Правительства Российской Федерации от 6 июля № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных систем и дальнейшего хранения содержащейся в их базах данных информации»

Термин	Определение	Источник
Инструмент	Средство осуществления действий, направленных на решение определенных задач или достижение определенной цели.	ГОСТ 56407 — 2015 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Основные методы и инструменты
Информационная система	Совокупность содержащейся в базах данных информации и обеспечивающих ее обработку информационных технологий и технических средств.	Федеральный закон от 27 июля 2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации»
Инфраструктура взаимодействия	Единый комплекс информационно-технологических и телекоммуникационных элементов, включающий информационные элементы в составе следующих информационных систем и входящих в них подсистем: а) федеральная государственная информационная система "Единый портал государственных и муниципальных услуг (функций)"; б) федеральная государственная информационная система "Федеральный реестр государственных и муниципальных услуг (функций)"; в) информационная система головного удостоверяющего центра, функции которого осуществляет уполномоченный федеральный орган исполнительной власти; г) федеральная государственная информационная система "Единая система идентификации и аутентификации в инфраструктуре, обеспечивающей информационно-технологическое взаимодействие информационных систем, используемых для предоставления государственных и муниципальных услуг в электронной форме";	Постановление Правительства Российской Федерации от 8 июня 2011 г. № 451 «Об инфраструктуре, обеспечивающей информационно-технологическое взаимодействие информационных систем, используемых для предоставления государственных и муниципальных услуг и исполнения государственных и муниципальных функций в электронной форме»

Термин	**Определение**	**Источник**
	д) единая система межведомственного электронного взаимодействия; е) федеральная государственная информационная система "Единая система нормативной справочной информации"; ж) федеральная государственная информационная система, обеспечивающая процесс досудебного (внесудебного) обжалования решений и действий (бездействия), совершенных при предоставлении государственных и муниципальных услуг; з) федеральная государственная информационная система "Единая информационная платформа национальной системы управления данными"; и) федеральная государственная информационная система "Единая информационная система управления кадровым составом государственной гражданской службы Российской Федерации" и интегрированная с ней цифровая кадровая платформа органов публичной власти (платформа "Государственные кадры"); к) информационная система обеспечения внутриведомственного и межведомственного документооборота и контроля исполнения поручений, в том числе с использованием облачных сервисов.	

Термин	Определение	Источник
Итерационный подход к разработке	Выполнение последовательности итераций, каждая из которых включает в себя реализацию группы требований к Системе, предопределенный набор целей, критерии приемки каждого требования, плановые сроки реализации итераций, критерии готовности к вводу в эксплуатацию.	Постановление Правительства Российской Федерации от 6 июля № 676 «О требованиях к порядку создания, развития, ввода в эксплуатацию, эксплуатации и вывода из эксплуатации государственных информационных
		систем и дальнейшего хранения содержащейся в их базах данных информации»
Канбан	Средство информирования, с помощью которого дается разрешение или указание на производство или передачу изделий в производстве, организованном по принципу вытягивания. Примечание — В переводе с японского языка Канбан означает "бирка" или "значок". Наиболее известным и распространенным примером таких средств коммуникации служат карточки Канбан. Во многих случаях они представляют собой листки бумаги, иногда помещенные в прозрачные пластиковые конверты, на которых указана следующая информация: наименование детали, номер детали, внешний поставщик или внутренний процесс-поставщик, число изделий в упаковке, местоположение склада и процесса-потребителя. На карточке может быть помещен штрих-код для считывания или автоматического выставления счета.	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь

Термин	Определение	Источник
Клиент	Физическое или юридическое лицо, индивидуальный предприниматель, иные лица, получающие государственные услуги и (или) результаты осуществления государственных функций, а также лица, деятельность которых является объектом государственного контроля (надзора).	Материалы 32-го Всемирного Конгресс IPMA, сессия «Клиентоцентричное управление государственными проектами»
Клиентоцентричность	Определение и максимальное удовлетворение явных и скрытых потребностей пользователей с учетом разумных издержек и соблюдения интересов заинтересованных сторон.	Материалы 32-го Всемирного Конгресс IPMA, сессия «Клиентоцентричное управление государственными проектами»
Команда разработки	Кросс-функциональная команда специалистов размером от 7 до 12 человек, которая наделена полномочиями и способна определить, создать, протестировать и внедрить определенную функциональность Системы, работающая в рамках Инкремента короткими фиксированными циклами.	Термин определён настоящим документом
Конвейер непрерывной доставки	Автоматизированные рабочие процессы, обеспечивающие реализацию новой функциональности Системы от проектирования до развертывания в составе релиза.	Термин определён настоящим документом
Контур разработки	Часть платформы разработки, обеспечивающая разработку и/или адаптацию программного обеспечения Системы командами разработки.	Термин определён настоящим документом
Контур тестирования	Часть платформы разработки, обеспечивающая проведение тестирование разработанного программного обеспечения Системы.	Термин определён настоящим документом

Термин	**Определение**	**Источник**
Контур эксплуатации	Часть платформы разработки, обеспечивающая эксплуатацию Системы.	Термин определён настоящим документом

Термин	**Определение**	**Источник**
Метод	Систематизированная совокупность шагов, действий, которые необходимо предпринять, чтобы решить определенную задачу или достичь определенной цели.	ГОСТ 56407 — 2015 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Основные методы и инструменты
Минимально достаточная версия Системы (МДВ)	Версия Системы, обладающая минимальными, но достаточными функциями для удовлетворения одной или нескольких потребностей клиентов.	Термин определён настоящим документом
Непрерывная интеграция	Процесс обеспечения согласованного функционирования разработанных функций Системы, их проверки Системы в контуре тестирования, по результатам которого обеспечивается готовность релиза к развертыванию и выпуску.	Термин определён настоящим документом
Непрерывное исследование	Процесс, который поддерживает инновации в интересах реализации новой функциональности Системы путем постоянного изучения потребностей клиентов и определения набора функций Системы, которые отвечают этим потребностям.	Термин определён настоящим документом
Непрерывное развертывание	Процесс, который обеспечивает автоматизированный перенос проверенных функций в контуре тестирования в контур эксплуатации, где они могут быть предоставлены клиентам.	Термин определён настоящим документом

Термин	**Определение**	**Источник**
Очередь (автоматизированной системы)	Часть автоматизированной системы, для которой в техническом задании на создание в целом установлены отдельные сроки ввода и набор реализуемых функций.	ГОСТ Р59853 — 2021 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения
Планирование Инкремента	Регулярно повторяющееся мероприятие по планированию работ в рамках предстоящего Инкремента, которое является основополагающим для Группы команд разработки и объединяет все Команды разработки для согласования общих целей выполнения работ.	Термин определён настоящим документом
Платформа разработки	Программно-аппаратная среда, предоставляющая функционально полный набор информационно-технологических сервисов, позволяющих эффективно создавать, развивать и эксплуатировать прикладное программное обеспечение государственных информационных систем и их компонентов.	Постановление Правительства Российской Федерации от 12 октября 2020 г. № 1674 «О проведении эксперимента по созданию, переводу и развитию государственных информационных систем и их компонентов на единой цифровой платформе Российской Федерации «ГосТех»
Поток создания ценности	Все действия, как создающие, так и не создающие ценность, которые позволяют продукции пройти все процессы — от разработки концепции до запуска в производство и от принятия заказа до доставки потребителю. Используется как интегральное понятие, включающее в себя материальные потоки (сырье, материалы,	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь

Термин	Определение	Источник
	комплектующие, детали и сборочные единицы, готовую продукцию), информационные и финансовые потоки, направленные на создание и доставку готовой продукции потребителю в установленное время, в установленном месте, с установленной стоимостью, с последующим ее обслуживанием в процессе эксплуатации и утилизации.	
Поток	Совокупность элементарных действий, которые управляются как целое, характеризуемое скоростью перемещения основной характеристики объекта.	ГОСТ Р 57524-2017 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Поток создания ценности
Пред- и пост-планирование инкремента	Мероприятия по планированию до и после планирования инкремента, используются для подготовки к планированию инкремента для Группы команд разработки и поставщиков, а также последующих действий после планирования.	Термин определён настоящим документом
Программно-аппаратный комплекс (ПАК)	Интегрированная программно-аппаратная среда, имеющая компонентную структуру и предназначенная для развертывания и функционирования в соответствии с его назначением.	Отчетные материалы по государственному заданию от 19 октября 2020 г. № 071-00002-20-05
Продукт	Изделие и/или услуга, которые в настоящее время и в будущем будут предлагаться производителем и обладают определенной ценностью для существующих или потенциальных клиентов.	ГОСТ Р 58537- 2019. УПРАВЛЕНИЕ ПРОДУКЦИЕЙ. Основные положения
Пул команд разработки	Несколько совместно работающих Групп команд разработки, каждая из которых обеспечивает	Термин определён настоящим документом

Термин	Определение	Источник
	разработку отдельных функциональных подсистем Системы.	
Рабочая группа по созданию и развитию Системы	Группа, включающая должностных лиц Ведомства и привлекаемых Ведомством исполнителей, обеспечивающая создание, развитие и эксплуатацию Системы на всех этапах ее жизненного цикла.	Термин определён настоящим документом
Результат оказания государственной услуги	Сведения в электронной форме, формируемые по результатам оказания государственных услуг, внесенные в государственные и муниципальные информационные системы.	Постановление Правительства Российской Федерации от 26 марта 2016 г. № 236 «О требованиях к предоставлению в электронной форме государственных и муниципальных услуг»
Релиз	Набор новых и/или измененных элементов конфигурации, которые были совместно протестированы и внедрены в рабочую среду.	ГОСТ Р ИСО/МЭК 19770-1-2014 Информационные технологии. Менеджмент программных активов. Часть 1. Процессы и оценка соответствия по уровням

Термин	**Определение**	**Источник**
Риск	Следствие влияния неопределенности на достижение поставленных целей. Риск часто характеризуют путем описания возможного события и его последствий или их сочетания. Под следствием влияния неопределенности необходимо понимать отклонение от ожидаемого результата или события (позитивное и/или негативное).	ГОСТ Р 57306-2016 Инжиниринг. Терминология и основные понятия в области инжиниринга
Синергетический эффект	Повышение результативности и эффективности деятельности в результате соединения, интеграции или слияния отдельных частей в единую систему.	ГОСТ 57 522 — 2017 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Руководство по интегрированной системе менеджмента качества и бережливого производства
Системная команда	Команда для решения специальных задач по созданию инфраструктуры разработки, включающей необходимые программно-аппаратные средства для интеграции разработанной функциональности в составе Системы, проведения всех видов тестирования, демонстрации Систем и решений, развертывания новых версий Системы, а также предоставления доступа к инфраструктуре разработки для Рабочей группы.	Термин определён настоящим документом

Термин	**Определение**	**Источник**
Системный подход	Методология изучения (рассмотрения) системы как целостного комплекса взаимосвязанных и взаимодействующих элементов для достижения поставленных целей.	Садовский В. Н. Системный подход и общая теория систем: статус, основные проблемы и перспективы развития. — М.: Наука, 1980.
Стадия создания (автоматизированной системы, АС)	Одна из частей процесса создания АС, установленная нормативными документами и заканчивающаяся выпуском документации на АС, содержащей описание полной, в рамках заданных требований, модели АС на заданном для данной стадии уровне или изготовлением несерийных компонентов АС, или приемкой АС в промышленную эксплуатацию.	ГОСТ Р59853 — 2021 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы
Управление потоком создания ценности	Деятельность по планированию, реализации, проверке и улучшению ПСЦ как системы процессов, направленной на удовлетворение требований потребителей и других заинтересованных сторон.	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь
Управление продуктами	Управление продуктами отвечает за определение и поддержку создания желательных, выполнимых, жизнеспособных и устойчивых продуктов, которые удовлетворяют потребности клиентов в течение жизненного цикла продукта на рынке.	ГОСТ Р 58537- 2019. УПРАВЛЕНИЕ ПРОДУКЦИЕЙ. Основные положения
Функциональность	Способность продукта или услуги обеспечивать функции, которые соответствуют установленным и предполагаемым потребностям при их использовании в определенных условиях.	ГОСТ Р 57306-2016 Инжиниринг. Терминология и основные понятия в области инжиниринга

Термин	**Определение**	**Источник**
Функция (автоматизированной системы)	Совокупность действий, направленная на достижение определенной цели.	ГОСТ Р59853 — 2021 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения.
Функция федеральных органов исполнительной власти	Нормативно установленный вид властной деятельности указанного органа власти, постоянно осуществляемый им в масштабах Российской Федерации. Функций федеральных органов исполнительной власти: а) функции по принятию нормативных правовых актов; б) функции по контролю и надзору; г) функции по управлению государственным имуществом; д) функции по оказанию государственных услуг.	Указ Президента Российской Федерации от 9 марта 2004 г. № 314 «О системе и структуре федеральных органов исполнительной власти»
Ценность (value)	Полезность, присущая продукции с точки зрения потребителя и находящая отражение в цене продаж и рыночном спросе.	ГОСТ Р 56020-2020 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО. Основные положения и словарь
Цикл «Планируй — Делай — Проверяй — Действуй» (PDCA: Plan — Do — Check — Act)	Цикл, позволяющий организации обеспечивать ее процессы необходимыми ресурсами, осуществлять их менеджмент, определять и реализовывать возможности для улучшения. Цикл PDCA может быть применен ко всем процессам и к системе менеджмента в целом. Описание цикла PDCA: - планируй — разработка целей системы и ее процессов, определение ресурсов, необходимых для	ГОСТ Р 57524-2017 БЕРЕЖЛИВОЕ ПРОИЗВОДСТВО Поток создания ценности

Термин	Определение	Источник
	достижения результатов в соответствии с требованиями потребителей и политикой организации, определение и рассмотрение рисков и возможностей; - делай — выполнение того, что запланировано; - проверяй — мониторинг и (там, где это применимо) измерение процессов, продукции в сравнении с политикой, целями, требованиями и запланированными действиями и сообщение о результатах; - действуй — принятие мер по улучшению результатов деятельности в той степени, насколько это необходимо.	
Цифровой продукт	Отдельная программа для ЭВМ (приложение) для выполнения некоего конечного процесса.	Приказ Минкомсвязи России от 1 августа 2019 г. № 428 «Об утверждении Разъяснений (методических рекомендаций) по разработке региональных проектов в рамках федеральных проектов национальной программы «Цифровая экономика Российской Федерации»

Термин	Определение	Источник
Цифровой сервис	Комплексное решение на базе цифровых продуктов, направленное на значимое качественное улучшение или ускорение процессов жизнедеятельности, организационных или бизнес-процессов, в том числе производственных процессов.	Приказ Минкомсвязи России от 1 августа 2019 г. № 428 «Об утверждении Разъяснений (методических рекомендаций) по разработке региональных проектов в рамках федеральных проектов национальной программы «Цифровая экономика Российской Федерации»
Эмерджéнтность	Эмерджéнтность в теории систем — свойство систем, обуславливающее появление новых свойств и качеств, не присущих элементам, входящим в состав системы.	ГОСТ Р 43.0.30-2022 Информационное обеспечение техники и операторской деятельности. Системность.
Этап создания (автоматизированной системы, АС)	Часть стадии создания АС, выделенная по соображениям единства характера работ и (или) завершающего результата, и (или) специализации исполнителей.	ГОСТ Р59853 — 2021 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы
DevOps	Набор практик и средств автоматизации разработки, нацеленных на активное взаимодействие разработчиков программного обеспечения со специалистами служб эксплуатации и поддержки, с тем чтобы обеспечить взаимную интеграцию их рабочих процессов. Основу составляет идея о тесной взаимозависимости процессов разработки и эксплуатации программного обеспечения (ПО).	Описание производственного процесса по созданию сервисов на цифровой платформе «ГосТех»

Термин	**Определение**	**Источник**
DevSecOps	Интегрированный набор практик и средств автоматизации разработки, обеспечивающий управление безопасностью в рамках DevOps и автоматизирующий основные процессы, связанные с обеспечением безопасности, в том числе за счет использования автоматических тестов функций безопасности.	Инструменты DevSecOps, обеспечивающие безопасность рабочих процессов DevOps. Кев Зеттлер. Atlassian.

ПРИЛОЖЕНИЕ № 3

к Методическим рекомендациям
по организации производственного процесса разработки
государственных информационных систем с учетом
применения итерационного подхода к разработке

ОБЯЗАННОСТИ РАБОЧЕЙ ГРУППЫ СОЗДАНИЯ И РАЗВИТИЯ ГОСУДАРСТВЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Оглавление

1. ОБЯЗАННОСТИ ЧЛЕНОВ РАБОЧЕЙ ГРУППЫ ПРИ ВЫПОЛНЕНИИ РАБОТ НА БАЗОВОМ УРОВНЕ

1.1. Владелец системы

Владелец Системы —должностное лицо Ведомства, которое несет основную, в том числе финансовую, ответственность за результаты создания (развития) Системы, за надзор и соответствие Системы нормативным правовым актам, за ввод в эксплуатацию и эксплуатацию Системы в течение всего ее жизненного цикла.

Владелец Системы назначается внутренним распорядительным документом Ведомства (приказом).

Владельцу Системы непосредственно подчинена Группа управления Системой.

Владелец Системы осуществляет управление созданием (развитием) Системы, в том числе:

а. принимает участие в формировании Группы ответственных за Систему - должностных лиц Ведомства, отвечающих за процесс сопровождения создания и развития Системы;

б. организовывает разработку Концепции создания (развития) Системы и ТЭО, представляет документы на утверждение вышестоящим должностным лицам Ведомства, ответственным за создание Системы;

в. организовывает разработку ТЗ на создание (развитие) Системы, утверждает ТЗ или представляет его на утверждение вышестоящим должностным лицам Ведомства, ответственным за создание Системы;

г. представляет уполномоченным должностным лицам Ведомства на согласование и утверждение проекты Концепции, ТЭО и правовых актов, являющихся основанием для создания Системы, ТЗ, а также обеспечивает согласование ТЗ с уполномоченным федеральным органом исполнительной власти, осуществляющим функции по выработке и реализации государственной политики и нормативно-правовому регулированию в сфере информационных технологий (в соответствии с Постановлением № 676);

д. организовывает проведение мероприятий по привлечению исполнителей либо внутри Ведомства, либо внешних - на контрактной или иной основе в соответствии с законодательством Российской Федерации;

е. организовывает подготовку и непосредственно участвует в планировании Инкрементов, дает оценку текущему состоянию работ, определяет ценность

командных целей и утверждает окончательный план Команд разработки на Инкремент;

ж. осуществляет контроль разработки и согласования рабочей документации, предоставляет на утверждение рабочую документацию и (при необходимости) доработанную документацию на Систему;

з. принимает участие в демонстрации Системы для оценки прогресса и предоставления Командам разработки обратной связи, в том числе принимает решение о привлечении клиентов к участию в демонстрации Системы;

и. принимает решение о выпуске и составе релизов Системы;

к. принимает решение о вводе в эксплуатацию релиза, согласовывает (утверждает/представляет на утверждение) соответствующие документы;

л. контролирует проведение предварительных испытаний, опытной эксплуатации, приемочных испытаний, согласовывает (утверждает\представляет на утверждение) соответствующие документы;

м. организует проведение аудита Системы на соответствие нормативным требованиям с привлечением уполномоченных Ведомств, при необходимости организует взаимодействие с другими Ведомствами (например, ФСТЭК, ФСБ);

н. принимает участие в мероприятиях по корректировке выполняемых работ, оценивает фактически достигнутый результат, организовывает корректировку целей разработки на следующий Инкремент с учетом выявленных проблем;

о. принимает решения по возникающим проблемам создания (развития) Системы.

1.2. Группа ответственных за систему

Группа ответственных за Систему — должностные лица и специалисты Ведомства, отвечающие за реализацию требований к Системе, соответствующих потребностям конечных пользователей Системы (потребителей государственных услуг), зафиксированных в ТЗ на Систему.

Группа ответственных за Систему контролирует работу Исполнителя, а также иных специалистов и команд, привлечение которых необходимо для выполнения работ по созданию (развитию) Системы.

Состав Группы ответственных за Систему утверждается внутренним распорядительным документом Ведомства (приказом) и, в общем случае, включает руководителя группы, его заместителя, должностных лиц различных подразделений

Ведомства, в том числе отвечающих за предоставление требуемой функциональности Системы клиентам, информационной безопасности и организации эксплуатации.

Группа ответственных за Систему (Ответственные за подсистему — в части касающейся):

а. принимает участие в разработке проектов документов планирования создания и развития Системы, а также проводит обоснование необходимости и целесообразности ее создания (развития);

б. разрабатывает основные положения Концепции создания (развития) Системы, согласовывает ее с должностными лицами Ведомства и, при необходимости, другими Ведомствами;

в. разрабатывает Техническое задание на создание (развитие) Системы, включая Группы требований, согласовывает его с должностными лицами Ведомства и другими Ведомствами;

г. разрабатывает документы, необходимые для привлечения исполнителей на контрактной или иной основе (в соответствии с законодательством Российской Федерации);

д. организовывает совместно с Менеджером процессов Группы команд подготовку и проведение планирования Инкремента, координируют совместно с Ответственными за реализацию и Менеджером процессов Группы команд взаимодействие Команд разработки в ходе планирования;

е. участвует в мероприятиях по синхронизации Команд разработки;

ж. организовывает совместно с Менеджером процессов Группы команд и Ответственными за реализацию подведение итогов Инкремента, оценивает под руководством Владельца Системы фактически достигнутый результат, готовит и представляет Владельцу Системы предложения по корректировке целей разработки Системы на следующий Инкремент с учетом выявленных проблем;

з. обеспечивает подготовку и проведение мероприятия выпуска релизов Системы, включая демонстрацию Системы, проведение предварительных испытаний, опытной эксплуатации, разрабатывает и представляют на утверждение Владельцу Системы соответствующие документы;

и. участвует в приемке релизов и Системы в целом в эксплуатацию, разрабатывает и представляет на утверждение Владельцу Системы соответствующие документы;

к. обеспечивает юридическое сопровождение выполняемых работ;

л. организовывает взаимодействие с поставщиками в интересах обеспечения команд разработки необходимым оборудованием и программным обеспечением;

м. участвует при необходимости в совместной работе с другими Ведомствами (Минцифры, ФСТЭК, ФСБ).

1.3. Ответственный за реализацию

Ответственный за реализацию — член Команды разработки, ответственный за определение приоритетов и управление реализацией функциональности Системы посредством итераций при сохранении концептуальной и технической целостности компонентов Системы и достижении целей Инкремента.

Ответственный за реализацию имеет следующие обязанности.

При подготовке к планированию Инкремента Ответственный за реализацию:

а. участвует в формировании и уточнении задач Команды разработки на предстоящий Инкремент, а также в определении критериев достижения целей команды в ходе Инкремента;

б. разрабатывает предложения по целям и задачам выполнения работ.

Во время мероприятия планирования итерации Ответственный за реализацию:

а. участвует в определении планируемых к реализации функций;

б. формирует детали реализации функций, таким образом, чтобы Команда разработки к концу планирования согласовала окончательный план команды;

в. координирует и согласовывает зависимости между работами Команд с другими Ответственными за реализацию;

г. если Команда разработки в период планирования Инкремента берет на себя риски, осуществляет планирование и работу по парированию рисков или уменьшению их влияния.

В рамках выполнения Инкремента Ответственный за реализацию:

а. осуществляет контроль реализации функциональности Системы;

б. обеспечивает скоординированную работу Команды для реализации запланированной функциональности точно в срок — за время Инкремента, с требуемым качеством, уделяя особое внимание в первую очередь пригодности для использования реализованной функциональности;

в. участвует (совместно с Системным архитектором / инженером) в принятии решений по созданию критически важной технологической инфраструктуры, которая будет обеспечивать создание новой функциональности;

г. взаимодействует с Командой разработки, чтобы обеспечить соответствие реализованной функциональности критериям приемлемости в ходе приемочных испытаний и опытной эксплуатации;

д. участвует во всех стандартных мероприятиях команды (планирование итераций, ежедневный стендап, демонстрация Системы, ретроспектива).

Ответственный за реализацию участвует в ретроспективе Инкремента, где Команды разработки собираются для улучшения своих процессов, и активно участвует в семинарах Группы команд разработки в рамках мероприятия «Инспекция и адаптация».

1.4. Менеджер процессов группы команд

Менеджер процессов Группы команд является лидером Группы команд разработки. Он отвечает за эффективность работы Группы команд разработки через соблюдение установленных процессов.

Менеджер процессов Группы команд поддерживает взаимодействие (коммуникации) со всеми заинтересованными сторонами Рабочей группы, решает возникающие проблемы и способствуют постоянному улучшению качества работы Группой команд за счет постоянного совершенствования процессов.

В интересах совершенствования процессов выпуска версий Менеджер процессов Группы команд должен иметь знания и практические навыки в области внедрения бережливых и гибких практик создания Системы, а также возможных вариантов масштабирования разработки Системы, в том числе с учетом рекомендаций настоящего документа.

Менеджер процессов Группы команд разработки (пула команд разработки) обязан:

а. улучшать процессы создания Системы с помощью различных инструментов гибкого и бережливого управления, принятых в организации, ведущей разработку продукта (решения);

б. обеспечивать стандартизацию и документирование процессов выпуска версий Системы;

в. участвовать в подготовке и планировании Инкремента;

г. содействовать периодической синхронизации действий и результатов всех команд, входящих в состав Группы команд;

д. организовывать обучение Команд разработки методологии гибкого и бережливого создания продуктов, в том числе в соответствии с рекомендациями настоящего документа;

е. оказывать помощь Командам разработки в обеспечении взаимодействия;

ж. выявлять проблемы взаимодействия Команд разработки и эскалировать их при необходимости на уровень Группы ответственных за Систему и Владельца Системы;

з. предоставлять Группе ответственных за Систему информацию о необходимых ресурсах для реагирования на риски и устранения возникших проблем взаимодействия Команд разработки;

и. участвовать в демонстрации Системы с целью минимизации рисков несогласованной работы Команд разработки;

к. обеспечивать улучшение процессов работы Группы команд разработки с помощью мероприятий по проверке их соответствия установленным требованиям к процессам и внедрение (адаптацию) улучшений процессов;

л. оценивать уровень соответствия команд рекомендованным процессам подготовки и выпуска версий и содействовать улучшению процессов.

Менеджер процессов Группы команд должен обладать лидерскими организационными навыками для выполнения своей роли:

а. проводить мониторинг социальных аспектов Команд разработки;

б. слушать и поддерживать команды в выявлении проблем и принятии решений;

в. быть открытым и ценить открытость в других;

г. создавать среду взаимного влияния;

д. понимать и сопереживать другим;

е. поощрять и поддерживать личностное развитие каждого сотрудника и развитие команд;

ж. формировать у членов команд активную позицию и инициативу в обсуждении рабочих вопросов и (децентрализованном) принятии решений;

з. применять проактивное и системное мышление;

и. поддерживать обязательства команд.

1.5. Менеджер процессов команды разработки

Менеджер процессов команды разработки — это лидер Команды разработки, отвечающий за эффективность ее работы через выполнение установленных требований к процессу разработки. Эту роль, как правило, берет на себя член Команды разработки. Менеджер процессов команды тратит большую часть своего рабочего времени на поддержание согласованных процессов, помогая другим членам Команды разработки взаимодействовать и сотрудничать. Менеджер команды также помогает команде координировать свои действия с другими командами в Группе команд.

Менеджер процессов команды обязан:

а. поддерживать принципы организации процессов итеративной разработки Системы;

б. способствовать прогрессу Команды разработки в достижении целей команды через повышение качества и производительности выполнения работ, а также предсказуемости и согласованности;

в. оказывать помощь Команде разработки в планировании и проведении итерации команды и достижении ежедневных целей в контексте создания Системы в целом;

г. участвовать совместно с Ответственным за реализацию в организации и проведении демонстрации результатов каждой итерации при участии всех заинтересованных лиц;

д. организовывать проведение ретроспективы при участии всех членов Команды разработки;

е. организовывать и проводить мероприятия, в том числе (где это применимо) ежедневное совещание на ногах, планирование, обзор итераций и ретроспективу итераций;

ж. внедрять практику качества, в соответствии с настоящим документом, предоставлять рекомендации по улучшению качества результатов;

з. помогать развивать культуру технической дисциплины, которая является отличительной чертой эффективных команд гибкой разработки;

и. помогать команде совершенствоваться и брать на себя ответственность за свои действия;

к. помогать команде разрешать межличностные конфликты и проблемы, а также выявлять возможности для роста;

л. помогать координировать межкомандное сотрудничество в Группе команд, постоянно улучшать коммуникации и отношения с другими командами;

м. помогать команде строить эффективные отношения с системной командой, архитекторами/инженерами и общими службами;

н. взаимодействовать с Менеджерами процессов других команд, чтобы помочь Команде разработки внести свой вклад в создание и развитие Системы;

о. участвовать в оценке результатов итераций и разрабатывать рекомендации по улучшению работы на последующих итерациях;

п. проводить мониторинг социальных аспектов команды и поддержание командного духа.

1.6. Архитектор системы

Архитектор Системы — технический специалист (или команда специалистов), который разрабатывает и уточняет проектные (технические) решения, определяет архитектуру Системы и осуществляет архитектурный надзор в процессе создания и развития Системы.

Архитектор Системы:

а. разрабатывает решения по архитектуре Системы и документацию на Систему, в том числе техническую документацию в объеме, необходимом для описания полной совокупности проектных решений (в том числе по защите информации) и достаточном для дальнейшего выполнения работ по созданию Системы, а также описание проектных (технических) решений, обеспечивающих реализацию одного или нескольких процессов предоставления государственных услуг, государственных функций, включая контрольно-надзорную деятельность, подлежащих автоматизации и/или цифровой трансформации и реализуемых посредством Системы;

б. уточняет проектные (архитектурные) решения, обеспечивающие реализацию и интеграцию функциональности, реализуемой различными Командами разработки в предстоящей итерации;

в. определяет необходимость создания вспомогательных технических и программных средств, необходимых для создания новых функций и возможностей Системы;

г. определяет основные технологии, планируемые к использованию при разработке Системы, и нефункциональные требования;

д. разрабатывает архитектурные решения для реализации функциональности в предстоящем Инкременте, а также соответствующие изменения в методах разработки и в использовании практик DevSecOps, таких как непрерывная интеграция, непрерывное развертывание и автоматизация тестирования;

е. организовывает разработку и осуществляет архитектурный надзор за разработкой рабочей документации на Систему;

ж. проводит изучение альтернативных вариантов архитектурных решений и определяет рациональные проектные решения в соответствии с обоснованными критериями;

з. планирует и обеспечивает архитектурную поддержку реализации новых возможностей Системы;

и. разрабатывает и согласовывает технологические подходы к разработке функциональности Системы Командами разработки, входящими в состав в Группы команд;

к. внедряет и обеспечивает использование конвейера непрерывной разработки функциональности Системы;

л. принимает участие в подготовке и проведении планирования Инкремента, в проведении демонстрации Системы, а также в подготовке релизов Системы и внедрении Системы в эксплуатацию;

м. обеспечивает архитектурный надзор за качеством разработки, а также использования необходимого инструментария в Командах разработки.

2. ОБЯЗАННОСТИ ЧЛЕНОВ РАБОЧЕЙ ГРУППЫ ПРИ ВЫПОЛНЕНИИ РАБОТ НА РАСШИРЕННОМ УРОВНЕ

2.1. Владелец системы

Владелец Системы — должностное лицо Ведомства, которое несет основную, в том числе финансовую, ответственность за результаты создания (развития) Системы, за надзор и соответствие Системы нормативным правовым актам, за ввод в эксплуатацию и эксплуатацию Системы в течение всего ее жизненного цикла.

Владелец Системы назначается внутренним распорядительным документом Ведомства (приказом).

Владельцу Системы непосредственно подчинена Группа управления Системой.

Владелец Системы осуществляет управление созданием (развитием) Системы, в том числе:

а. принимает решение об организации работ по созданию и развитию Системы на расширенном уровне;

б. принимает участие в формировании Группы ответственных за Систему - должностных лиц Ведомства, отвечающих за процесс сопровождения создания и развития Системы;

в. организовывает разработку Концепции создания (развития) Системы и ТЭО, представляет документы на утверждение вышестоящим должностным лицам Ведомства, ответственным за создание Системы;

г. организовывает разработку ТЗ на создание (развитие) Системы, утверждает ТЗ или представляет его на утверждение вышестоящим должностным лицам Ведомства, ответственного за создание Системы;

д. представляет уполномоченным должностным лицам Ведомства на согласование и утверждение проекты Концепции, ТЭО и правовых актов, являющихся основанием для создания Системы, ТЗ, а также обеспечивает согласование ТЗ с уполномоченным федеральным органом исполнительной власти, осуществляющим функции по выработке и реализации государственной политики и нормативно-правовому регулированию в сфере информационных технологий (в соответствии с Постановлением № 676);

е. организовывает проведение мероприятий по привлечению исполнителей либо внутри Ведомства, либо внешних - на контрактной или иной основе в соответствии с законодательством Российской Федерации;

ж. организовывает подготовку и непосредственно участвует в планировании Инкрементов, дает оценку текущему состоянию работ, определяет ценность командных целей и утверждает окончательный план Команд разработки на Инкремент;

з. Осуществляет контроль разработки и согласования рабочей документации, предоставляет на утверждение рабочую документацию и (при необходимости) доработанную документацию на Систему;

и. принимает участие в демонстрации Системы для оценки прогресса и предоставления Командам разработки обратной связи, в том числе принимает решение о привлечении клиентов к участию в демонстрации Системы;

к. принимает решение о выпуске и составе релизов Системы;

л. принимает решение о вводе в эксплуатацию релиза, согласовывает (утверждает/представляет на утверждение) соответствующие документы;

м. контролирует проведение предварительных испытаний, опытной эксплуатации, приемочных испытаний, согласовывает (утверждает/представляет на утверждение) соответствующие документы;

н. организует проведение аудита Системы на соответствие нормативным требованиям с привлечением уполномоченных Ведомств, при необходимости организует взаимодействие с другими Ведомствами (например, ФСТЭК, ФСБ);

о. принимает участие в мероприятиях по корректировке выполняемых работ, оценивает фактически достигнутый результат, организовывает корректировку целей разработки на следующий Инкремент с учетом выявленных проблем;

п. принимает решения по возникающим проблемам создания (развития) Системы;

2.2. Группа ответственных за Систему

Группа ответственных за Систему — должностные лица и специалисты Ведомства, отвечающие за реализацию требований к Системе, соответствующих потребностям конечных пользователей Системы (потребителей государственных услуг), зафиксированных в ТЗ на Систему.

При выполнении работ на расширенном уровне Группа ответственных за Систему, включает Ответственных за каждую из подсистем, назначаемых из должностных лиц Ведомства и отвечающих за предоставление клиентам требуемой функциональности соответствующей подсистемы.

Группа ответственных за Систему контролирует работу Исполнителя, а также иных специалистов и команд, привлечение которых необходимо для выполнения работ по созданию (развитию) Системы.

Состав Группы ответственных за Систему утверждается внутренним распорядительным документом Ведомства (приказом) и, в общем случае, включает руководителя группы, его заместителя, должностных лиц различных подразделений Ведомства, в том числе отвечающих за предоставление требуемой функциональности Системы клиентам, информационной безопасности и организации эксплуатации.

Группа ответственных за Систему:

а. принимает участие в разработке проектов документов планирования создания и развития Системы, а также проводит обоснование необходимости и целесообразности ее создания (развития);

б. разрабатывает основные положения Концепции создания (развития) Системы, согласовывает ее с должностными лицами Ведомства и, при необходимости, другими Ведомствами;

в. разрабатывает Техническое задание на создание (развитие) Системы, включая Группы требований, согласовывает его с должностными лицами Ведомства и другими Ведомствами;

г. разрабатывает документы, необходимые для привлечения исполнителей на контрактной или иной основе (в соответствии с законодательством Российской Федерации);

д. организовывает совместно с Менеджером процессов Группы команд подготовку и проведение планирования Инкремента, координирует совместно с Ответственными за реализацию и Менеджером процессов Группы команд взаимодействие Команд разработки в ходе планирования;

е. участвует в мероприятиях по синхронизации работ Команд разработки;

ж. организовывает совместно с Менеджером процессов Группы команд и Ответственными за реализацию подведение итогов Инкремента, оценивает под руководством Владельца Системы фактически достигнутый результат, готовит и представляет Владельцу Системы предложения по корректировке целей разработки Системы на следующий Инкремент с учетом выявленных проблем;

з. обеспечивает подготовку и проведение мероприятия выпуска релизов Системы, включая демонстрацию системы, проведение предварительных испытаний, опытной эксплуатации, разрабатывает и представляет на утверждение Владельцу Системы соответствующие документы;

и. участвует в приемке релизов и Системы в целом в эксплуатацию, разрабатывает и представляет на утверждение Владельцу Системы соответствующие документы;

к. обеспечивает юридическое сопровождение выполняемых работ;

л. организовывает взаимодействие с поставщиками в интересах обеспечения команд разработки необходимым оборудованием и программным обеспечением;

м. участвует при необходимости в совместной работе с другими Ведомствами (Минцифры России, ФСТЭК, ФСБ).

2.3. Ответственный за подсистему

Ответственный за Систему — должностное лицо или специалист Ведомства, отвечающий за реализацию требований к подсистеме в рамках создания (развития) Системы, соответствующих потребностям конечных пользователей подсистемы, зафиксированных в ТЗ на Систему.

Ответственный за Систему контролирует работу Исполнителя, а также иных специалистов и команд, привлечение которых необходимо для выполнения работ по созданию (развитию) подсистемы.

Ответственный за подсистему назначается внутренним распорядительным документом Ведомства (приказом) и входит в состав Группы ответственных за Систему.

Ответственный за подсистему:

а. принимает участие в разработке проектов документов планирования, создания и развития подсистемы, а также проводит обоснование необходимости и целесообразности ее создания (развития);

б. участвует в разработке основных положений Концепции создания (развития) Системы (в части касающейся подсистемы);

в. участвует в разработке Технического задания на создание (развитие) Системы, включая Группы требований (в части касающейся подсистемы);

г. участвует в планировании Инкремента;

д. участвует в мероприятиях по синхронизации Команд разработки;

е. участвует в подведении итогов Инкремента, оценивает под руководством Владельца Системы фактически достигнутый результат в части реализации функциональности подсистемы, готовит и представляет Владельцу Системы

предложения по корректировке целей разработки подсистемы на следующий Инкремент с учетом выявленных проблем;

ж. участвует в подготовке и проведении мероприятий выпуска релизов Системы, в которых реализуется (дорабатывается) функциональность подсистемы;

з. участвует в приемке релизов и Системы в целом в эксплуатацию, разрабатывает и представляют на утверждение Владельцу Системы соответствующие документы (в части касающейся подсистемы);

и. организовывает взаимодействие с поставщиками в интересах обеспечения команд разработки необходимым оборудованием и программным обеспечением (в части касающейся подсистемы);

к. участвует при необходимости в совместной работе с другими Ведомствами (Минцифры, ФСТЭК, ФСБ).

2.4. Главный архитектор Системы

Главный архитектор Системы — технический специалист, который разрабатывает и уточняет проектные (технические) решения, определяет архитектуру Системы и осуществляет архитектурный надзор в процессе создания и развития Системы.

При выполнении работ на расширенном уровне Главный архитектор Системы руководит командой архитекторов, включающей в том числе архитекторов подсистем.

Главный архитектор Системы:

а. разрабатывает решения по архитектуре Системы и документацию на Систему, в том числе техническую документацию в объеме, необходимом для описания полной совокупности проектных решений (в том числе по защите информации) и достаточном для дальнейшего выполнения работ по созданию Системы, а также описание проектных (технических) решений, обеспечивающих реализацию одного или нескольких процессов предоставления государственных услуг, государственных функций, включая контрольно-надзорную деятельность, подлежащих автоматизации и/или цифровой трансформации и реализуемых посредством Системы;

б. уточняет проектные (архитектурные) решения, обеспечивающие реализацию и интеграцию функциональности, реализуемой различными Командами разработки в предстоящей итерации;

в. определяет необходимость создания вспомогательных технических и программных средств, необходимых для создания новых функций и возможностей Системы;

г. определяет основные технологии, планируемые к использованию при разработке Системы, и нефункциональные требования;

д. разрабатывает архитектурные решения для реализации функциональности в предстоящем Инкременте, а также соответствующие изменения в методах разработки и в использовании практик DevSecOps, таких как непрерывная интеграция, непрерывное развертывание и автоматизация тестирования;

е. организовывает разработку и осуществляет архитектурный надзор за разработкой рабочей документации на Систему;

ж. организовывает и проводит синхронизацию архитектурных решений;

з. проводит изучение альтернативных вариантов архитектурных решений и определяет рациональные проектные решения в соответствии с обоснованными критериями;

и. планирует и обеспечивает архитектурную поддержку реализации новых возможностей Системы;

к. разрабатывает и согласовывает технологические подходы к разработке функциональности Системы Командами разработки, входящими в состав в Группы команд;

л. внедряет и обеспечивает использование конвейера непрерывной разработки функциональности Системы;

м. принимает участие в подготовке и проведении планирования Инкремента, в проведении демонстрации Системы, а также в подготовке релизов Системы и внедрении Системы в эксплуатацию;

н. обеспечивает архитектурный надзор за качеством разработки, а также использования необходимого инструментария в Командах разработки.

2.5. Архитектор подсистемы

Архитектор подсистемы — технический специалист, который разрабатывает и уточняет проектные (технические) решения, определяет архитектуру подсистемы.

Архитектор подсистемы может назначаться при выполнении работ на расширенном уровне и подчиняется Главному архитектору Системы.

Архитектор подсистемы:

а. разрабатывает решения по архитектуре Системы и документацию на Систему (в части касающейся подсистемы), в том числе техническую документацию в объеме, необходимом для описания полной совокупности проектных решений (в том числе по защите информации) и достаточном для дальнейшего выполнения работ по созданию Системы, а также описание проектных (технических) решений, обеспечивающих реализацию одного или нескольких процессов предоставления государственных услуг, государственных функций, включая контрольно-надзорную деятельность, подлежащих автоматизации и/или цифровой трансформации и реализуемых посредством подсистемы;

б. уточняет проектные (архитектурные) решения, обеспечивающие реализацию и интеграцию функциональности, реализуемой различными Командами разработки в предстоящей итерации (в части касающейся подсистемы);

в. определяет необходимость создания вспомогательных технических и программных средств, необходимых для создания новых функций и возможностей подсистемы;

г. определяет основные технологии, планируемые к использованию при разработке подсистемы, а также технические (архитектурные) требования;

д. разрабатывает архитектурные решения для реализации функциональности подсистемы в предстоящем Инкременте;

е. организовывает разработку и осуществляет архитектурный надзор за разработкой рабочей документации на Систему;

ж. проводит изучение альтернативных вариантов архитектурных решений по подсистеме и определяет рациональные проектные решения в соответствии с обоснованными критериями;

з. планирует и обеспечивает архитектурную поддержку реализации новых возможностей подсистемы;

и. участвует в мероприятиях по синхронизации архитектурных решений;

к. разрабатывает и согласовывает технологические подходы к разработке функциональности подсистемы Командами разработки, входящих в состав в Группы команд;

л. внедряет и обеспечивает использование конвейера непрерывной разработки функциональности Системы (в части касающейся подсистемы);

м. принимает участие в подготовке и проведении планирования Инкремента, в проведении демонстрации Системы, а также в подготовке релизов Системы и внедрении Системы в эксплуатацию (в части касающейся подсистемы);

н. обеспечивает архитектурный надзор за качеством разработки, а также использования необходимого инструментария в Командах разработки.

2.6. Менеджер процессов пула команд

Менеджер процессов пула команд является лидером Пула команд разработки. Он отвечает за эффективность работы Пула команд разработки через соблюдение установленных процессов выполнения работ на расширенном уровне.

Менеджер процессов Пула команд поддерживает взаимодействие (коммуникации) со всеми заинтересованными сторонами Рабочей группы, решает возникающие проблемы и способствуют постоянному улучшению качества работы Пулом команд за счет постоянного совершенствования процессов.

В интересах совершенствования процессов выпуска версий Менеджер процессов Пула команд должен иметь знания и практические навыки в области внедрения бережливых и гибких практик создания Системы, а также возможных вариантов масштабирования разработки Системы, в том числе с учетом рекомендаций настоящего документа.

При выполнении работ на расширенном уровне Менеджер процессов Пула команд разработки координирует действия Менеджеров процессов Групп команд.

Менеджер процессов Пула команд разработки обязан:

а. улучшать процессы создания Системы с помощью различных инструментов гибкого и бережливого управления, принятых в организации, ведущей разработку продукта (решения);

б. координировать действия Менеджеров процессов групп команд разработки;

в. обеспечивать стандартизацию и документирование процессов выпуска версий Системы;

г. участвовать в подготовке и организовывать планирование Инкремента, в том числе проведение мероприятий пред- и пост- планирования Инкремента;

д. содействовать периодической синхронизации действий и результатов всех Команд и Групп команд разработки, входящих в состав Пула команд;

е. организовывать обучение Команд разработки методологии гибкого и бережливого создания продуктов, в том числе в соответствии с рекомендациями настоящего документа;

ж. оказывать помощь Командам разработки в обеспечении взаимодействия;

з. выявлять проблемы взаимодействия Команд разработки и эскалировать их при необходимости на уровень Группы ответственных за Систему и Владельца Системы;

и. предоставлять Группе ответственных за Систему информацию о необходимых ресурсах для реагирования на риски и устранения возникших проблем взаимодействия Команд и Групп команд разработки;

к. участвовать в демонстрации Системы с целью минимизации рисков несогласованной работы Команд разработки;

л. участвовать в проведении мероприятий синхронизации Групп команд разработки;

м. обеспечивать улучшение процессов работы Пула команд разработки с помощью мероприятий по проверке их соответствия установленным требованиям к процессам и внедрение (адаптацию) улучшений процессов;

н. оценивать уровень соответствия команд рекомендованным процессам подготовки и выпуска версий и содействовать улучшению процессов.

Менеджер процессов Пула команд должен обладать лидерскими организационными навыками для выполнения своей роли:

а. проводить мониторинг социальных аспектов Команд и Групп команд разработки;

б. слушать и поддерживать команды в выявлении проблем и принятии решений;

в. быть открытым и ценить открытость в других;

г. создавать среду взаимного влияния;

д. понимать и сопереживать другим;

е. поощрять и поддерживать личностное развитие каждого сотрудника и развитие команд;

ж. формировать у членов команд активную позицию и инициативу в обсуждении рабочих вопросов и (децентрализованном) принятии решений;

з. применять проактивное и системное мышление;

и. поддерживать обязательства Команд разработки (Групп команд разработки).

2.7. Менеджер процессов группы команд

Менеджер процессов Группы команд является лидером Группы команд разработки. Он отвечает за эффективность работы Группы команд разработки через соблюдение установленных процессов.

Менеджер процессов Группы команд поддерживает взаимодействие (коммуникации) со всеми заинтересованными сторонами Рабочей группы, решает возникающие проблемы и способствуют постоянному улучшению качества работы Группой команд за счет постоянного совершенствования процессов.

В интересах совершенствования процессов выпуска версий Менеджер процессов Группы команд должен иметь знания и практические навыки в области внедрения бережливых и гибких практик создания Системы, а также возможных вариантов масштабирования разработки Системы, в том числе с учетом рекомендаций настоящего документа.

При выполнении работ на расширенном уровне Менеджер процессов группы команд взаимодействует с Менеджером процессов Пула команд, а также другими Менеджерами процессов группы команд.

Менеджер процессов Группы команд разработки обязан:

а. улучшать процессы создания Системы с помощью различных инструментов гибкого и бережливого управления, принятых в организации, ведущей разработку продукта (решения);

б. обеспечивать стандартизацию и документирование процессов выпуска версий Системы;

в. участвовать в подготовке и планировании Инкремента, а также проведении мероприятий пред- и пост- планирования Инкремента;

г. содействовать периодической синхронизации действий и результатов всех команд, входящих в состав Группы команд;

д. организовывать обучение Команд разработки методологии гибкого и бережливого создания продуктов, в том числе в соответствии с рекомендациями настоящего документа;

е. оказывать помощь Командам разработки в обеспечении взаимодействия;

ж. выявлять проблемы взаимодействия Команд разработки и эскалировать их при необходимости на уровень Группы ответственных за Систему и Владельца Системы;

з. предоставлять Группе ответственных за Систему информацию о необходимых ресурсах для реагирования на риски и устранения возникших проблем взаимодействия Команд разработки;

и. участвовать в демонстрации Системы с целью минимизации рисков несогласованной работы Команд разработки;

к. участвовать в проведении мероприятий синхронизации Групп команд разработки;

л. обеспечивать улучшение процессов работы Группы команд разработки с помощью мероприятий по проверке их соответствия установленным требованиям к процессам и внедрение (адаптацию) улучшений процессов;

м. оценивать уровень соответствия команд рекомендованным процессам подготовки и выпуска версий и содействовать улучшению процессов.

Менеджер процессов Группы команд должен обладать лидерскими организационными навыками для выполнения своей роли:

а. проводить мониторинг социальных аспектов Команд разработки;

б. слушать и поддерживать команды в выявлении проблем и принятии решений;

в. быть открытым и ценить открытость в других;

г. создавать среду взаимного влияния;

д. понимать и сопереживать другим;

е. поощрять и поддерживать личностное развитие каждого сотрудника и развитие команд;

ж. формировать у членов команд активную позицию и инициативу в обсуждении рабочих вопросов и (децентрализованном) принятии решений;

з. применять проактивное и системное мышление;

и. поддерживать обязательства команд.

2.8. Ответственный за реализацию

Обязанности Ответственного за реализацию на расширенном уровне аналогичны обязанностям на базовом уровне.

2.9. Менеджер процессов команды разработки

Обязанности Менеджера процессов команды разработки на расширенном уровне аналогичны обязанностям на базовом уровне.