Тема №2. Современная концепция управления IT-проектами, мировые практики управления проектами, в том числе и в области искусственного интеллекта.

PMI и PMBOK

PMI (Project Management Institute, Институт управления проектами) - всемирная некоммерческая профессиональная организация по управлению проектами.

- разработка стандартов,
- проведение исследований,
- образовательная деятельность,
- аккредитация в области управления проектами.

PMBOK – Свод знаний по управлению проектами, разработанный PMI.

- руководства, рекомендованные практики, стандартные принципы и общая терминология для управления проектами,
- 1-е издание – 1996г, 6-е издание – 2017г, 7-е издание намечено на 2023г,
- подразумевает процессно-ориентированный подход к управлению проектами,
- вся деятельность по управлению проектами подразделяется на 49 процессов, сгруппированных по группам процессов и областям знаний PMBOK.

Проект – это временное предприятие, предназначенное для создания уникальных продуктов, услуг или результатов.

IT проект – это проект в сфере создания, внедрения или применения информационных технологий и программного обеспечения.

Свойства проекта.

1. Временная ограниченность. Каждый проект имеет четкое начало и окончание.
2. Уникальность результатов. Не существует проектов с одинаковым результатом. Каждый проект создает некий результат, имеющий уникальные черты или характеристики.
3. Последовательная разработка. Любой проект можно представить как последовательность выполнения некоторых фаз.
4. Выполняются людьми. Реализация проекта выполняется командой исполнителей.
5. Ограниченная доступность ресурсов. В качестве ресурсов выступают люди, финансы, материалы. Каждый ресурс ограничен. Исполнители – составом команды проекта. Финансы и материалы – бюджетом проекта.
6. Все проекты планируются, исполняются и управляются. Для реализации проекта всегда составляется план. Команда придерживается плана при исполнении работ проекта. В случае отклонения от плана он корректируется.

Причины инициации проектов.

1. Требования рынка. Рынок требует новый продукт, который создается в результате проекта. Например, новая программная система.
2. Нужды организации. Для оптимизации организационных процессов внутри предприятия его руководство инициирует новый проект. Например, внедрение ERP-системы.
3. Требования заказчика. Заказчику требуется некое уникальное программное обеспечение с заданным набором свойств.
4. Технологический прогресс. На рынке аппаратных устройств появилась новая аппаратная платформа, которая требует разработки нового программного обеспечения. Например, новая игровая приставка требует новых игр.
5. Требования законодательства. Появление новых законов может инициировать проекты модификации программного обеспечения с целью исполнения их требований. Например, для обеспечения требуемого уровня зашиты персональных данных.

Результаты проекта.

1. Продукт, который может выступать как компонент другого изделия или некоторое принципиально новое изделие. Продукт ориентируется на использование сторонними потребителями.
2. Новая услуга или способность предоставлять новую услугу (например, веб-сайт для потенциальных покупателей продукции предприятия в сети интернет).
3. Модернизация или модификация существующей номенклатуры продуктов или услуг.
4. Некий уникальный результат. Например, отчет о маркетинговом исследовании рынка. Ценность и уникальность заключаются в содержании отчета и собранных результатах исследований.

Основные характеристики проектов

1. Цель проекта. Что должно быть получено после его реализации.
2. Время на реализацию проекта. Через какой промежуток времени должен быть получен требуемый результат.
3. Команда проекта. Кто будет заниматься его реализацией.
4. Новизна. Что уникального реализует проект, чем он отличается от всех предшествующих проектов.
5. Ограничения проекта. В качестве ограничений выступают время, ресурсы и качество.

Области знаний и процессы управления проектом.

Управление проектами – это приложение знаний, навыков, инструментов и методов к операциям проекта для удовлетворения предъявляемых к проекту требований.

Согласно PMBok при управлении проектом может быть задействовано 49 процессов управления проектом, логически сгруппированных по областям знаний.

Процессы управления проектом объединены в пять групп:

- Инициация – авторизация и начало проекта.
- Планирование – составление плана работ по проекту.
- Исполнение – выполнение запланированных работ.
- Мониторинг и контроль – отслеживание всех этапов проекта и своевременное внесение изменений
- Завершение – закрытие проекта.

В PMBOK описаны 10 областей знаний, которые практически всегда используются в проектах.

Область знаний	Назначение области знаний
Управление интеграцией	Объединение всех процессов управления проектом
Управление содержанием	Обеспечения того, чтобы проект содержал только те работы, которые требуются для успешного выполнения проекта
Управление расписанием	Управление сроками выполнения отдельных задач, фаз и всего проекта в целом
Управление стоимостью	Разработка бюджета, привлечение финансирования, контроль стоимости, исполнение проекта в рамках одобренного бюджета
Управление качеством	Применение политики организации в области качества относительно собственно проекта, а также требований к качеству продукта с целью удовлетворения ожиданий заинтересованных сторон
Управление ресурсами	Приобретение и управление ресурсами, необходимыми для успешного выполнения проекта
Управление коммуникациями	Обеспечения своевременного сбора, распространения, хранения, извлечения, обработки и в конечном счете архивирования/утилизации информации проекта
Управление рисками	Идентификация, анализ, планирование реагирования, собственно реагирование, а также мониторинг рисков в проекте
Управление закупками	Покупки или приобретения извне необходимых продуктов, услуг или результатов

Управление заинтересованными сторонами	Идентификация источников влияния, которые могут воздействовать на проект или подвергаться воздействию проекта, для проведения анализа ожиданий заинтересованных сторон и их воздействия на проект, а также для разработки стратегий взаимодействия с ними

Окружение проекта

Окружение проекта - набор внешних и внутренних факторов, а также ключевых заинтересованных лиц, влияющих на достижение результатов проекта.

Внутреннее окружение проекта - условия реализации проекта.

Внешнее окружение проекта представлено предприятием и его окружением.

Систематическое наблюдение за окружением проекта - одна из важнейших функций руководителя проекта и его команды. Для руководителя проекта важно не только знать окружение проекта, но и обеспечивать связь проекта с ключевыми заинтересованными лицами и факторами для достижения максимального успеха проекта.

Организационная структура проекта - определяет взаимоотношения между основными участниками проекта. Распределение прав, ответственности и обязанностей.

Команда проекта - "мозговой центр", мотор и исполнительный орган проекта. От психологического климата в команде и уровня доверия между участниками и их слаженности зависит успех проекта.

Стиль руководства - характеризует творческую активность лиц, принимающих решения относительно проекта и управления командой.

Методы и средства коммуникаций - определяют полноту, достоверность и оперативность обмена информацией.

Участники проекта (заинтересованные стороны) - преследуют различные интересы в проекте, формируют свои требования в соответствии со своими целями и мотивацией, оказывают влияние на проект своими интересами. К заинтересованным сторонам проекта могут относиться: инициатор проекта, заказчик (владелец), инвесторы, команда проекта и функциональные группы, руководитель проекта, генеральный контрактор, субконтрактор, проектировщики, генеральный подрядчик, субподрядчики, поставщики, лицензоры, органы власти, владелец земельного участка, владелец здания, производитель готовой продукции, потребители, общественные группы населения, конкуренты, другие заинтересованные стороны.

Социальные условия проекта - уровень заработной платы, условия труда и техники безопасности и др.

К внешнему окружению проекта относят требования руководства предприятия и влияние предприятия на проект.

Требования руководства предприятия:

- требования к результатам проекта,
- требования к реализации проекта,
- методика и порядок корректировки цели и требований к проекту со стороны специалистов предприятия и др.

Влияние предприятия на проект:

- сфера финансов - определяет бюджетные рамки проекта, способы и источники финансирования проекта;
- сфера сбыта - формирует важные требования к проекту, связанные с рынком сбыта, наличием и действиями конкурентов;
- сферы производства, материального обеспечения, инфраструктуры и прочего.

Жизненный цикл проекта

Жизненный цикл проекта – последовательность этапов реализации проекта от его начала до завершения.

Пример жизненного цикла проекта разработки программного обеспечения:

- начало проекта;
- анализ предметной области;
- построение математической модели;
- разработка алгоритма;
- разработка программы;
- тестирование и исправление ошибок;
- внедрение;
- завершение проекта.

Фаза проекта – совокупность логически связанных операций проекта, завершающихся достижением одного или ряда поставляемых результатов. Фазы, как правило, являются последовательными. Фаза обозначает некоторый завершенный этап проекта и должна заканчиваться получением некоторого законченного результата, который физически передается следующей фазе. Результатом фазы может быть изделие, компонент, программный модуль, библиотека подпрограмм, информация в виде отчета и т.п. Примеры фаз: разработка концепции, анализ целесообразности, требования заказчика, разработка решения, проектирование, испытания, ввод в эксплуатацию.

Этапы развития методов управления проектами

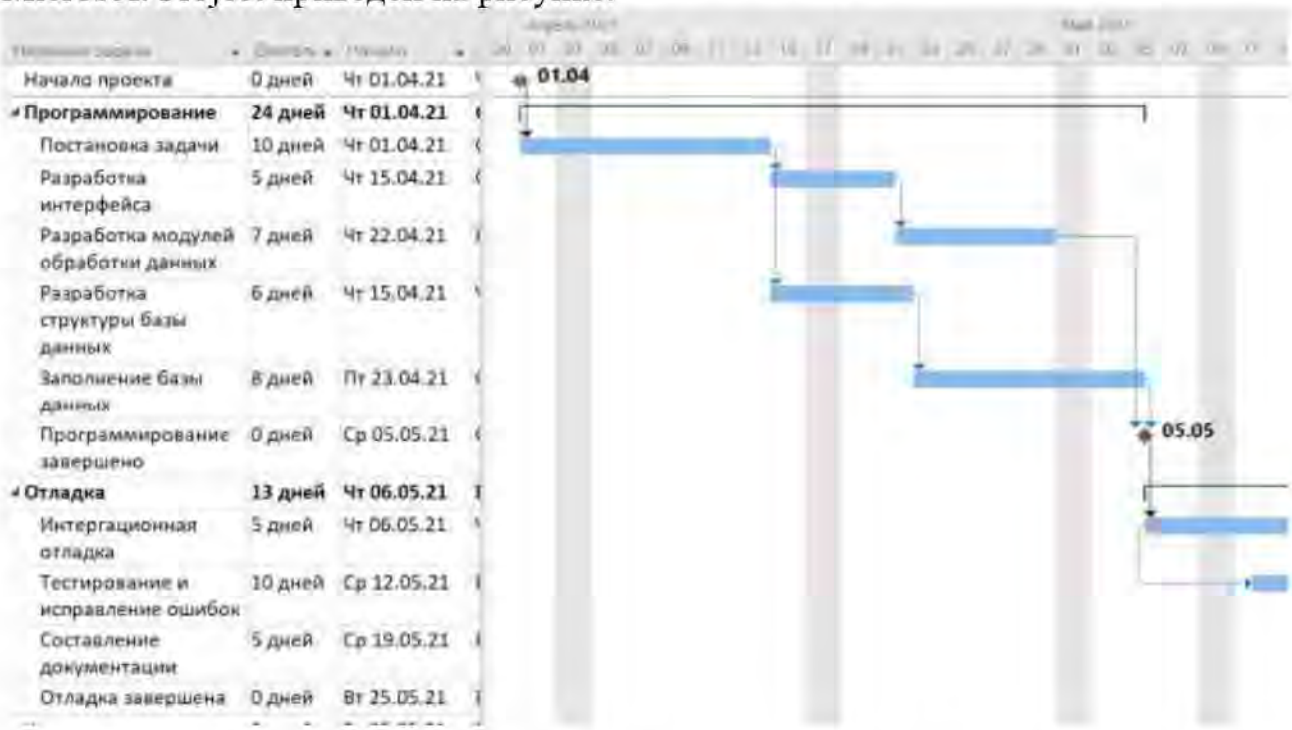
1910-е годы – появление диаграммы Ганта. Первая диаграмма была разработана Генри Ганттом в 1910г в качестве календарного графика выполнения работ по проекту.

Основные характеристики и особенности диаграммы Гантта:

- по горизонтальной оси откладывается время выполнения задач проекта;
- по вертикальной оси откладываются задачи проекта, которые нужно выполнить от его начала до окончания;

- горизонтальная полоса обозначает плановую длительность выполнения задачи, дату ее начала и окончания;
- задачи могут выступать в качестве вех (работ с нулевой длительностью), которые обозначают начало или завершение определенных этапов проекта;
- диаграмма не содержит сведений о сущности проводимых работ и их важности для достижения целей проекта;
- диаграмма не содержит сведений о трудоемкости задач, количестве и квалификации требуемых для них ресурсов;
- благодаря своей простоте и наглядности диаграмма Гантта и в XXI веке является важным инструментом в системе управления проектами.

Пример современной диаграммы Ганта, сформированной в системк Microsoft Project приведен на рисунке.



1920-е годы – разработана теория научной организации труда НОТ (А. К. Гастев и др.).

- Основная задача – поиск методов повышения производительности труда на каждом отдельно взятом рабочем месте.
- Основное внимание уделяется человеческому фактору и организации труда работников.
- Человек – это основная производительная сила организации. От его эффективности зависит эффективность работы организации.
- Залог и ключевой фактор эффективной работы сотрудников – эффективное использование рабочего времени в течение рабочего дня. Этот постулат является прототипом современного тайм-менеджмента.

Теория научной организации труда послужила прототипом и основой концепции «бережливого производства» (Lean Production), которая была

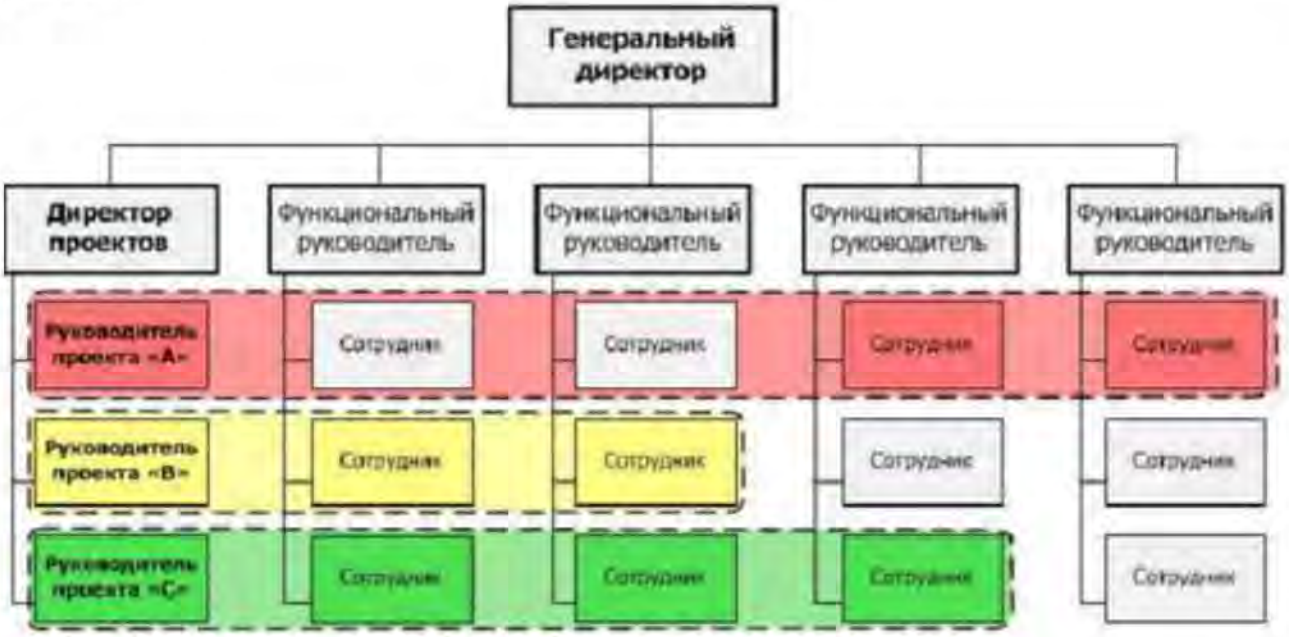
разработана и внедрена в производственный процесс концерна Toyota в 1950-х годах. Эта концепция предполагает «управление производственным предприятием, связанным постоянным стремлением к устранением всех видов потерь» и используется концерном Toyota до настоящего времени.

1930-е годы – появление матричной организации в управлении проектами. В США были разработаны методы координации и управления крупными проектами для нескольких предметных областей:

- авиационные проекты − в компании US Air Corporation,
- нефтегазовые проекты − в компании Exon Mobil.

В 1930г - в US Air Corporation впервые был создан проектный офис, отвечающий за реализацию проекта в целом. В 1937 г. Л. Гулик (США) опубликовал первую статью по матричной организации. Предложенная им матричная организация управления проектами содержала горизонтальные связи между подразделениями предприятия. В 1930-е гг. заложены современные основы управления проектами в военно-промышленном комплексе.

Пример модели матричной организации сложных проектов приведен на рисунке.



1940-е годы – управление проектами на основе исследования операций.

- Методы исследования операций позволяют находить оптимальные проектные решения за счет применения экономико-математических методов и методов оптимизации.
- В период Второй Мировой войны применение этих методов позволило значительно более эффективно управлять подразделениями вооруженных сил, их логистикой и взаимодействием.
- Наиболее известный проект - Manhattan−project в исполнении которого участвовало 600 000 человек, с бюджетом 2 млрд $.

Основными источниками успеха Manhattan–project считается высокое качество организации проекта, включая:

- четкую постановку цели проекта,
- полную поддержку руководством планов выполнения проектов,
- рациональную матричную структуру работ в рамках проекта,
- систему планирования обеспечения проектов ресурсами с использованием линейных графиков.

Пример задачи исследования операций. Имеется N предприятий, каждое из которых производит может производить одну и ту же продукцию в объеме Pi. Продукция хранится на M складах. Известна стоимость доставки продукции на каждый склад Sij и стоимость хранения единицы продукции на каждом складе Sj. Известен план потребления продукции K потребителями в объеме Rl и стоимость доставки со склада потребителю Rjl. Нужно определить сколько и какому производителю производить продукции, на какие склады ее отправлять, с каких складов потреблять потребителям и в каком количестве.

1950-е годы – разработка метода критического пути, полномасштабное внедрение наработок по матричной организации для управления проектами в вооруженных силах США:

- 1953 г. – в проектах для военно-воздушных сил,
- 1954 г. – в проектах по специальным видам вооружений,
- 1955 г. – в проектах для военно-морского флота.

В 1956-57 гг реализуется проект по модернизации заводов корпорации Du Pont de Nemours Co для выполнения нового оборонного заказа. В рамках управления этим проектом создается исследовательская группа для разработки и исследования технологий управления проектами с применением средств вычислительной техники. К работе группы привлекаются специалисты из UNIVAC и Remington Rand. К концу 1957 г. данная группа разрабатывает метод критического пути (CPM) и программно реализует его на ЭВМ UNIVAC.Применение метода CPM позволило на 1,5 года раньше успешно завершить проект модернизации заводов Du Pont de Nemours Co. Данный факт повлиял на широкое распространение CPM и успешное применение в других проектах в последующем.

1950-е годы – разработана система сетевого планирования PERT. Эта система была впервые использована в программе «Поларис» по оснащению подводных лодок США. В программу входили 250 фирм-исполнителей и свыше 9000 субподрядчиков.

Отличие метода PERT от CMP заключается в учете неопределенностей, возникающих при выполнении каждой проектной задачи. Оценка длительности задачи выполняется по трем точкам:

- оптимистическая длительность T(O), длительность задачи при отсутствии воздействия неблагоприятных факторов;

- пессимистическая длительность Т(П), длительность задачи при воздействии всех возможных неблагоприятных факторов;
- реалистическая длительность Т(Р), длительность задачи при воздействии наиболее вероятных неблагоприятных факторов.

Ожидаемое время выполнения задачи вычисляется по формуле:

$$T_{ож} = \frac{T(O) + 4T(P) + T(П)}{6}.$$

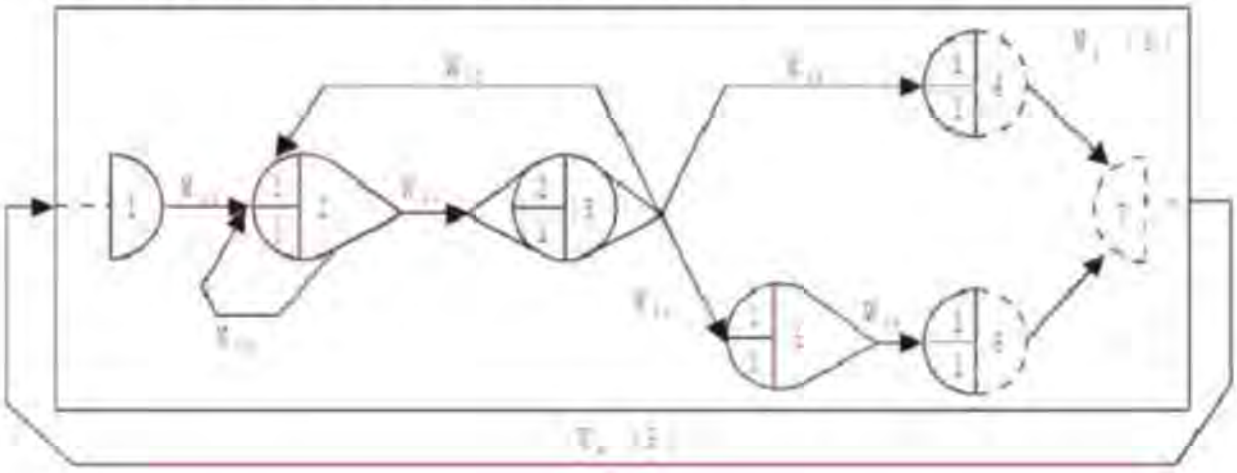
1960-е годы – разработан метод GERT. Основа применения метода GERT – использование альтернативных сетей, именуемых GERT-сетями.

GERT-сети позволяют более адекватно описывать сложные процессы производства в рамках проекта, когда затруднительно или невозможно однозначно определить, какие именно работы и в какой последовательности должны быть выполнены для достижения цели проекта.

Метод GERT применяется в случаях организации работ, когда последующие задачи проекта могут начинаться после завершения только некоторого числа из предшествующих задач, причем не все задачи, представленные на сетевой модели, должны быть выполнены для завершения проекта.

Расчет GERT-сетей, моделирующих реальные процессы в рамках проекта, требует специального программного обеспечения для вычисления эффективности предлагаемых сетевых моделей.

Пример GERT-сети изображен на рисунке.



1960-е годы – появление специализированных профессиональных организаций по управлению проектами, которые стали центрами сбора, анализа и распространения передовых методик. В Европе таким центром стала Международная Ассоциация управления проектами (IPMA), в США – Институт управления проектами (PMI).

Сопоставление параметров IPMA и PMI приведено в таблице.

	IPMA	PMI
Год создания	1965	1969

Основные методы деятельности	Сертификация организаций, сертификация специалистов, обучение, организация групп по интересам	Сертификация, стандартизация производства
Число участников	Более 65 тыс. человек	Более 341 тыс. человек
Количество стран	55	Более 160
Число направлений специализации и уровней сертификации	4 уровня сертификации	33 направления специализации
Количество копий основных материалов	Более 50 тыс. сертификатов	Более 2 млн. копий Свода Знаний по управлению проектами (PMBok)

1970-е годы – развитие подходов к управлению проектами, обусловленное широким внедрением средств вычислительной техники в различных отраслях. Масштабное использование метода СРМ при управлении проектами привело к его законодательной поддержке в США.

Существенно продвинулись исследования и практические методики в области управления командой проекта. В 1971г разработаны схемы компетенций и порядок взаимодействия менеджера проекта с командой проекта. В 1977г - предложены методы управления конфликтами между участниками проекта.

В это же время происходило становление профессиональных организаций в области управления проектами в различных регионах мира:

- в Австралии организован Австралийский институт управления проектами (AIPM),
- в Азии − Японская ассоциация развития инжиниринга (ENAA).

1980-е годы стали началом разработки фундаментального труда в области управления проектами – Свода знаний PMBok (Project Management Body of Knowledge). Работа над PMBok проводилась силами американской профессиональной организации PMI.

В это же время продолжают развиваться специализированные направления управления проектами по отраслям:

- в строительстве – с ориентацией на условия конкретного контракта и конкретного заказчика;
- в сфере информационных технологий – разработка подходов и методов управления конфигурацией, методов управления изменениями требований;
- в промышленном производстве – методов управления качеством реализуемого проекта;

- в социальных системах – методов управления, ориентированных на обеспечение слаженной работы команды проекта;
- в инвестиционных проектах – методов своевременной идентификации, оценки и управления рисками.

1990-е годы – первое издание PMBok. 1991г - в Германии издан учебник и практическое руководство по управлению проектами, подготовленные национальной Ассоциацией Управления Проектами Германии (GPM). 1996г – первое издание PMBok. PMBok стал фундаментальным трудом в области управления проектами, который постепенно дополнялся описанием новых методик и передовых практик. До настоящего времени было выпущено 6 изланий PMBok. Последнее 6-е издание появилось в 2017 г. А очередное 7-е издание ожидается к выходу в свет в 2023г.

В 1990г - создана Советская (позднее Российская) Ассоциация управления проектами СОВНЕТ. Начало разработке и вводу в действие программ сертификации менеджеров проектов по стандартами PMI. В СССР на рубеже 1980-90 гг разработан рад ГОСТ, стандартизировавших порядок управления проектами в различных сферах деятельности. Например, в сфере разработки программного обеспечения утвержден ГОСТ 34.601-90 «Автоматизированные системы. Стадии создания», который отражал сложившуюся к тому моменту каскадную модель жизненного цикла разработки программных систем. Серия ГОСТ 34 надолго послужила основой регламентов разработки и документирования программного обеспечения, многие из которых используются в России по настоящий день

2000-е годы – развитие гибких методологий Agile. В 2001 году, 17 разработчиков ПО, которые называли себя «организационными анархистами» встретились в городе Сноубёрд (Snowbird, Utah) чтобы поделиться идеями повышения эффективности управления проектами. Они сформулировали знаменитый манифест Agile, включающий 4 ценности и 12 принципов.

Манифест закрепил методы управления разработкой программного, отличные от каскадной модели, ориентированные на принятие и внедрение в проект изменяющихся требований к разрабатываемому программному продукту. Появление, развитие и внедрение таких методов происходило постепенно в течение 1990-х гг. Но именно в 2001 г. они получили формальную основу в виде манифеста Agile.

2010-е годы – развитие и внедрение гибких методологий Agile. Появление фреймворков и методов, основанных на принципах Agile:

- Scrum – гибкая методология, основанная на итерациях разработки, называемых Спринтом. Длительность спринта – 2-4 недели. Результат спринта – рабочая версия ПО.
- Lean – проект делится на итерации, каждая из которых проходит свои фазы инициации, планирования, исполнения подобно тем, что описаны в PMBok.

- Kanban – методология основана на доске, на которую помещаются карточки работ, которые перемещаются между этапами в зависимости от состояния и степени завершенности работы.
- Extreme Programming (XP) – разработка очень мелкими итерациями с постоянным вовлечением заказчика для оценки релизов.

Тема №4. Разработка проекта. Жизненный цикл проекта.

Жизненный цикл проекта

Жизненный цикл проекта – это последовательность этапов, называемых фазами, которые проект проходит от своего начала до окончания.

Результатом фазы является некоторый завершенный результат, который передается на последующие фазы проекта. В качестве результата может выступать некое изделие, компонент, программный модуль, библиотека подпрограмм, отчет, график и т.п. Именно факт передачи результата является тем критерием, по которому последовательность проектных работ разбивается на отдельные фазы. Результат каждой фазы должен пройти внутреннюю приемку руководством проекта, одобрен и только после этого он может быть использован на последующих фазах проекта.

Жизненный цикл проекта определяет:

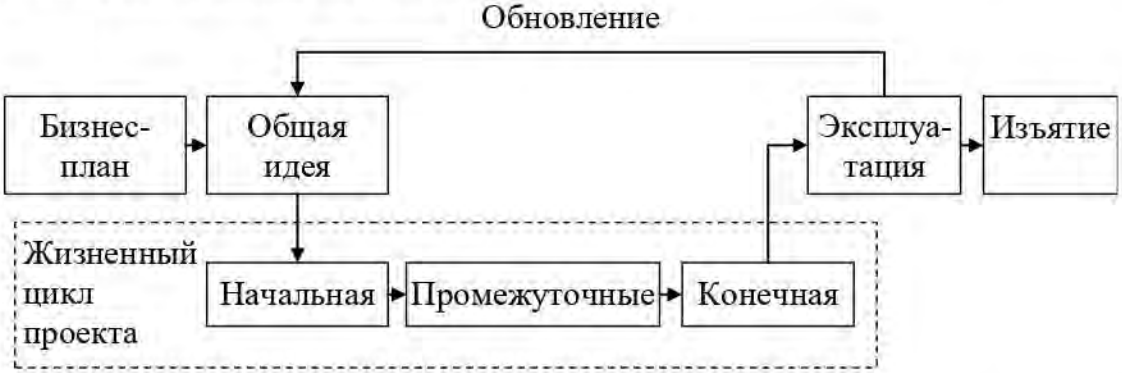
- перечень работ, которые должны быть выполнены в каждой фазе, и их последовательность;
- перечень результатов, которые должны быть получены в каждой фазе;
- перечень исполнителей каждой фазы;
- подходы и процедуры к контролю и подтверждению результатов, полученных в каждой фазе.

Свойства жизненного цикла проекта

- затраты на проектные работы и состав команды проекта минимальны в его начальной фазе;
- затраты на проектные работы и состав команды проекта максимальны в его промежуточных фазах и быстро снижаются на заключительных этапах реализации проекта;
- неуверенность в успешном завершении и риск неудачи проекта максимальны в начале и снижаются по ходу его реализации;
- уверенность в завершении проекта увеличивается по ходу выполнения проекта;
- возможности команды проекта и его стейкхолдеров оказать влияние на параметры конечного результата наиболее высоки на начальных фазах проекта и уменьшаются в процессе его реализации;
- внесение изменений в параметры проекта и его продукта имеют минимальную стоимость на начальных фазах и максимальную – на заключительных.

Жизненный цикл продукта и проекта

Жизненный цикл продукта гораздо шире жизненного цикла проекта. Их взаимосвязь изображена на рисунке.



В рамках жизненного цикла продукта изначально формируется бизнес-план, подтверждающий целесообразность создания продукта и инициации проекта по его разработке. Формируется общая идея продукта, которая ложится в основу инициируемого проекта. Начинается проект, результатом которого должен стать принципиально новый, востребованный, полезный и экономически оправданный продукт. Проект проходит все стадии своего жизненного цикла.

Жизненный цикл проекта завершается созданием намеченного продукта, который поступает в эксплуатацию. Через некоторое время эксплуатации появляется потребность модернизации продукта, которая ложится в основу общей идеи нового проекта. Запускается проект модернизации, снова проходящий все фазы своего жизненного цикла. Создается модернизированный продукт, который поступает в эксплуатацию. Продукт может проходить несколько итераций модернизации, до тех пор пока не наступает момент его морального старения и дальнейшая модернизация не имеет смысла. Продукт выводится из эксплуатации.

Каскадная модель жизненного цикла информационной системы

Каскадная модель состоит из последовательности следующих этапов (фаз): формирование требований, анализ требований, проектирование, кодирование, тестирование, внедрение, сопровождение и эксплуатация.

Особенность модели – неизменная последовательность фаз и отсутствие возвратов к предыдущим этапам разработки. Каждая следующая фаза должна быть завершена полностью, прежде чем начнется последующая фаза разработки. Фазы завершаются созданием некоторого промежуточного результата, который подробно документируется и передается дальше. Требования к разрабатываемому ПО (ИС) подробно документируются в виде технического задания, которое не изменяется до завершения проекта.

Преимущества каскадной модели:

- в конце каждой фазы создается максимально подробный комплект документации, который впоследствии становится частью общей документации по проекту;
- последовательность работ по проекту выглядит четко, ясно и понятно, что позволяет планировать сроки выполнения и затраты по этим работам.

Недостатки каскадной модели:

- реальная разработка никогда не проходит без изменения и корректировок в техническом задании;
- после корректировки технического задания возникает необходимость возврата назад, доработок на предыдущих этапах, что не предусмотрено каскадной моделью;
- возможные доработки приводят к значительному сдвигу заранее спланированных сроков разработки;
- полученное на выходе программное обеспечение может существенно отличаться от потребностей и ожиданий заказчиков и потенциальных потребителей.

Модель дает хорошие результаты в следующих случаях.

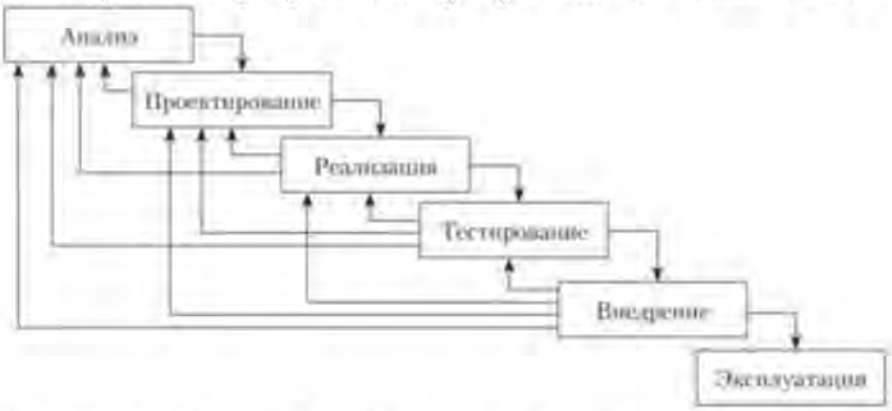
- В случаях, когда программный проект состоит из нескольких относительно независимых приложений, разработка каждого из может быть реализована по каскадной модели с достаточной степенью детализации.
- Для проектов разработки относительно простых информационных систем, функционирование которых может быть достаточно точно описано в самом начале в виде полного комплекта требований (учетно-аналитические, бухгалтерские системы).
- Для проектов, создающих новые версии уже существующих программных систем, когда заранее известны и описаны требования к новому функционалу (например, перенос программного продукта на новую аппаратную или программную платформу).
- При выполнении крупных проектов с несколькими большими командами разработчиков.

Дает плохие результаты в следующих случаях.

- Если разрабатывается новая система, для которой изначально, до начала работ, не сформулированы или неизвестны большинство требований.
- При разработке систем, для которых требования могут измениться еще до окончания процесса разработки.

Итеративная (инкрементная, эволюционная) модель ЖЦ ИС

Близкие по своей сути инкрементная, итеративная и эволюционная модели являются модификацией каскадной модели с предусмотренной возможностью возврата к предыдущим этапам, которые происходят при изменении первоначальных требований или уточнения деталей технического задания по ходу реализации продукта. Эти модели послужили основой при реализации многочисленных проектов разработки программного обеспечения.



Инкрементная модель предполагает реализацию программного обеспечения как последовательность законченных релизов на основе первоначально утвержденного и неизменного технического задания.

В итеративной модели этот подход скорректирован в сторону постепенного наращивания функциональности продукта. Жизненный цикл разбивается на последовательность итераций, в задачу которых входит создание некоторого завершенного функционального компонента. Разработка компонента осуществляется в рамках своеобразного подпроекта, а проект в целом представляет собой последовательность таких подпроектов. Результатом

итерации является завершенная версия системы, включающая функциональность как всех предыдущих подпроектов, так и функциональность текущего. После завершения последнего подпроекта на выходе получается полнофункциональный готовый продукт.

В эволюционной модели текущая итерация не завершается только сборкой готовой версии. Далее должно следовать ее развертывание в реальных условиях эксплуатации с целью получения откликов многочисленных пользователей и построения на их основе плана работ над следующей итерацией системы.

Достоинства модели.

- Пользователи имеют возможность увидеть, протестировать и оценить продукт уже на ранних стадиях разработки. Это позволяет своевременно внести коррективы, которые учитываются в последующих итерациях.
- Разработка итерациями позволяет равномерно распределить, а в некоторых случаях и сэкономить бюджет проекта. При этом нет необходимости выделять сразу весь объем на реализацию проекта.
- Первоначальная версия с ограниченной функциональностью становится доступна пользователю при гораздо меньших финансовых вложениях. Это позволяет снизить затраты на последующие итерации, если часть из них будет признана лишней или ненужной.
- Снижение неопределенности и рисков по окончании каждой итерации за счет своевременного внесения корректировок в требования и техническое задание.

Недостатки модели.

- Согласование результатов разработки с пользователями производится только в точках, планируемых после завершения каждого этапа работ, а общие требования к ИС зафиксированы в виде ТЗ на все время ее создания.
- Не предусмотрены итерации в рамках каждого инкремента.

- Определение полной функциональной системы должно осуществляться в начале ЖЦ.
- Затягивание проекта за счет переноса решений трудных проблем на будущее.

Область применения модели – для следующих проектов.

- Где большинство требований можно сформулировать заранее, но их появление ожидается через определенный период времени.
- Когда существует потребность быстро поставить на достаточно «узкий» рыночный сегмент продукт, имеющий базовый функционал.
- Для которых предусмотрен большой период времени разработки.
- Где при рассмотрении риска, финансирования, графика выполнения проекта, размера программы, ее сложности или необходимости в реализации на ранних фазах оказывается, что самым оптимальным вариантом является применение принципа пофазовой разработки.
- Выполнение которых проводится с применением новой технологии, что позволяет пользователю адаптироваться к системе путем выполнения более мелких инкрементных шагов, без резкого перехода к применению основного нового продукта.

V-образная модель

Создана с целью помочь работающей над проектом команде в планировании дальнейшего тестирования системы.

Позволяет гораздо лучше контролировать результат на предмет его соответствия ожиданиям, поскольку сфокусирована на тестировании. Дает возможность значительно повысить качество ПО за счет своей ориентации на тестирование. Во многом разрешила проблему соответствия созданного продукта выдвигаемым требованиям благодаря разработке процедур верификации и аттестации на ранних стадиях ЖЦ (пунктирные линии на рисунке указывают на зависимость этапов планирования/постановки задачи и тестирования/приемки).

Является всего лишь модификацией каскадной модели и обладает многими ее недостатками. Слабо приспособлена к возможным изменениям требований заказчика. Если процесс разработки занимает продолжительное время (иногда до нескольких лет), то полученный в результате продукт может оказаться фактически ненужным заказчику, поскольку его потребности существенно изменились.

Достоинства модели.

- Планирование верификации и тестирования разрабатываемого продукта на ранних стадиях его разработки.
- Предусматриваются аттестация и верификация всех внешних и внутренних полученных данных, а не только самого программного продукта.
- Определение требований выполняется перед разработкой проекта системы.

- Определяются продукты, которые должны быть получены в результате разработки, причем каждые полученные данные должны подвергаться тестированию.
- Модель проста в использовании.

Недостатки модели.

- Нет учета существующих между фазами итераций.
- Не предусмотрено внесение изменений на разных этапах ЖЦ.
- Тестирование требований происходит слишком поздно, вследствие чего невозможно внести изменения, не повлияв на график выполнения проекта.

Сфера применения.

- Проекты, в которых вся информация о требованиях доступна заранее.
- Для систем, требующих высокой надежности, например прикладные программы для наблюдения за пациентами в клиниках, а также встроенное ПО для устройств управления аварийными подушками безопасности в автомобилях.

Спиральная модель

Подразумевает разработку в виде последовательности версий, но в начале проекта определены не все требования. Требования уточняются в результате разработки версий.

На каждом витке спирали.

- Выполняется создание очередной версии продукта.
- Уточняются требования проекта.
- Определяется его качество.
- Планируются работы следующего витка.

Особое внимание уделяется начальным этапам разработки – анализу и проектированию, где реализуемость тех или иных технических решений

проверяется и обосновывается посредством создания прототипов (макетирования). Прикладное ПО (ИС) создается не сразу, как в случае каскадной схемы, а по частям, с использованием метода прототипирования. Под прототипом понимается программный компонент, реализующий отдельные функции и внешние интерфейсы, разрабатываемого ПО.

Создание ПО (ИС) осуществляется за несколько витков спирали. На каждом витке производится тщательная проверка риска превышения сроков и стоимости проекта.

Преимущества спиральной модели.

- Позволяет быстрее показать пользователям системы работоспособный продукт, активизируя процесс уточнения и дополнения требований.
- Допускает изменение требований при разработке ИС.
- Обеспечивает большую гибкость в управлении проектом.
- Позволяет получить более надежную и устойчивую систему. По мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации.
- Позволяет совершенствовать процесс разработки – анализ, проводимый в каждой итерации, позволяет проводить оценку того, что должно быть изменено и улучшено в организации разработки не следующем витке.
- Уменьшаются риски заказчика. Заказчик может с минимальными финансовыми потерями завершить развитие неперспективного проекта.

Недостатки:

- Увеличивается неопределенность у разработчика в перспективах развития проекта.
- Затруднены операции временного и ресурсного планирования проекта в целом. Нужно ввести временные ограничения на каждую из стадий ЖЦ.

Переход осуществляется в соответствии с планом, даже если не вся запланированная работа выполнена.

Сравнение моделей ЖЦ

Характеристика	**Каскадная**	**Инкрементная**	**Спиральная**
Новизна разработки и обеспеченность ресурсами	Типовой. Хорошо проработаны технология и методы решения задачи		Нетиповой (новаторский). Нетрадиционный для разработчика
	Ресурсов хватает для реализации в сжатые сроки	Ресурсов не хватает для реализации в сжатые сроки	
Масштаб проекта	Малые и средние	Средние и крупные	Любые
Сроки выполнения проекта	До года	До нескольких лет. Разработка одной версии может занимать срок от нескольких недель до года	
Заключение отдельных договоров на отдельные версии	Заключается один договор. Версия и есть итоговый результат проекта	На отдельную версию или несколько последовательных версий обычно заключается отдельный договор	
Определение основных требований в начале проекта	Да	Да	Нет
Изменение требований по мере развития проекта	Нет	Незначительное	Да
Разработка итерациями (версиями)	Нет	Да	Да
Распространение промежуточного ПО	Нет	Может быть	Да

Стандарт ГОСТ 34.601-90

Стандарт распространяется на автоматизированные системы и устанавливает стадии и этапы их создания и документального оформления. Содержит описание содержания работ на каждом этапе. Стадии и этапы работ, закрепленные в стандарте соответствуют каскадной модели жизненного цикла. Однако опыт его дальнейшего применения показал, что основные положения стандарта могут быть адаптированы и применены и к другим моделям жизненного цикла разработки ПО.

Предусмотрены следующие этапы создания информационной системы:

- формирование требований к АС;
- разработка концепции АС;
- техническое задание;
- эскизный проект;
- технический проект;
- рабочая документация;
- ввод в действие;
- сопровождение АС.

Тема №5 Часть 2. Устав проекта, требования и техническое задание.

Устав проекта
Устав проекта формируется Заказчиком проекта и содержит следующие разделы.

- Название проекта.
- Причины инициации проекта. Причины должны дат четкое объяснение зачем нужен данный проект.
- Цели проекта со стороны Заказчика. Описание ожидаемого результата с точки зрения Заказчика.
- Границы проекта. Границы должны явно указывать что не входит в рамки реализуемого проекта.
- Задачи проекта со стороны Заказчика. Описывают требования, которые должны быть реализованы для достижения поставленных целей. Например, автоматизация складского учета.
- Допущения и ограничения проекта со стороны Заказчика в отношении Исполнителя.
    - Допущения - это предположения относительно внешней среды проекта, в которой он будет реализован. Например, Заказчик допускает привлечение к исполнению определенной части работ сторонних организаций.
    - Ограничения - это совокупность факторов, ограничивающих возможности команды проекта. Например, предельный размер увеличения бюджета проекта равен 15%.
- Контрольные точки проекта. Определяют даты начала или завершения основных фаз проекта, даты контрольных демонстраций, получения промежуточных результатов.
- Бюджет проекта – суммарная стоимость всех его затрат, определяемая контрактом.
- Критерии успешности проекта. Четкие измеримые показатели, позволяющие сделать вывод о достижении проектом намеченных целей. Например, суммарное время обработки заказа – не более 5 мин.

- Назначение ответственных лиц по проекту. К ним относятся спонсор проекта, ответственный представитель заказчика, руководитель проекта и т.п.

Анализ требований

Анализ требований — это процесс сбора требований к программному обеспечению (ПО), их систематизации, документирования, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения.

Требования к проекту и продукту – важнейшая часть описания проекта и ТЗ на разработку ПО.

Источниками требований могут быть:

- Законодательство федерального или регионального уровня (законы, подзаконные акты, стандарты);
- нормативная база организации и принятая система документооборота (внутренние приказы и положения);
- текущая модель бизнес-процессов организации (порядок организации деятельности в настоящее время);
- переходные модели бизнес-процессов (как должна быть организована деятельность после реализации проекта);
- представления и ожидания стейкхолдеров проекта (что от него ждут потребители и потенциальные пользователи);
- регламенты, протоколы, журналы эксплуатации существующего программного обеспечения (какие проблемы имеются при его эксплуатации, какого функционала не хватает);
- конкурирующие программные продукты (что есть у конкурентов и насколько удачно конкурирующие продукты удовлетворяют потребностям пользователей).

Требования могут носить функциональный и нефункциональный характер.

Функциональный характер имеют требования к поведению системы. Они разделяются на такие категории.

- Бизнес-требования — описывают назначение ПО с точки зрения реализации тех или иных бизнес-процессов.
- Пользовательские требования — описывают набор задач, которые решаются пользователями при помощи программной системы. Наиболее распространенным способом описания таких требований являются пользовательские истории (user story). Они описывают проблему пользователя и последовательность действий его манипуляций с программой, которые обеспечивают ее решение. Также могут применяться описания способов применения (use case), суенариев взаимодействия (scenario) и описания в произвольной форме.
- Функциональные требования — это описание поведения системы в той или иной ситуации, последовательности обработки данных, форматы

входных и выходных данных и т.п. Эти требования принято описывать при помощи системной спецификации SRS (system requirement specification).

Нефункциональный характер имеют требования к характеру поведения системы. Они разделяются на такие категории.

- Бизнес-правила — набор правил и ограничений, которые присущи конкретной предметной области и конкретному предприятию. Например, непрерывный производственный цикл и трехсменный режим работы.
- Системные требования и ограничения — требования к аппаратному и программному окружению, в котором будет функционировать разрабатываемое программное обеспечение. Сюда входят программные интерфейсы, требования к аппаратному обеспечению и смежному стороннему ПО.
- Внешние системы и интерфейсы. Накладывают ограничения на взаимодействие со сторонним программным и аппаратным обеспечением.

Требования должны обладать свойствами, перечисленными в таблице.

Свойство	Содержание
Единичность	Требование описывает единственный аспект функциональности ПО
Завершённость	Требование описано от начала и до конца в единственном месте, которое содержит исчерпывающую информацию и все необходимые данные
Последовательность	Отсутствует противоречие с описанием других требований
Атомарность	Требование не может быть разбито на несколько других требований. В противном случае его следует разбить на несколько требований
Отслеживаемость	Требование каким-либо образом должно быть связано с потребностями предприятия, пользователей или стейкхолдеров разрабатываемого ПО
Актуальность	Требование не должно устаревать в процессе разработки программного обеспечения
Недвусмысленность	Описание требования не содержит неопределенностей, отсутствуют источники двусмысленной интерпретации. Прочтение требования должно давать четкое и понятное представление о его содержании
Обязательность	Требование описывает некий обязательный аспект программной системы, отсутствие которого делает ее незавершенной, недоработанной с точки зрения пользователя
Проверяемость	Требование должно быть таким, чтобы его выполнение можно было проверить. При проверке используются

	следующие методы: осмотр, демонстрация, тест или анализ.

К методам выявления требований относят:

- интервью, опросы, анкетирование;
- мозговой штурм, семинар;
- наблюдение за производственной деятельностью, «фотографирование» рабочего дня;
- анализ нормативной документации;
- анализ моделей деятельности;
- анализ конкурентных продуктов;
- анализ статистики использования предыдущих версий системы.

Все требования должны быть поддающимися проверке. Общепринятая методика проверки — тесты. Если проверка тестами невозможна, тогда должен использоваться другой метод проверки (анализ, демонстрация, осмотр или обзор дизайна).

Нефункциональные требования, которые являются неподдающимися проверке на программном уровне, все равно должны быть сохранены как документация намерений клиента. Такие требования к продукту могут быть преобразованы в требования к процессу. Например, нефункциональное требование, чтобы ПО не содержало «потайных ходов», может быть удовлетворено заменой на требование использовать парное программирование.

Требования обычно используются как средство коммуникации между различными заинтересованными лицами. Требования должны быть просты и понятны для обычных пользователей и разработчиков. Общий способ задокументировать требование — это написать утверждение о том, что должна сделать система.

В зарубежной и российской практике используется Спецификация требований программного обеспечения (англ. Software Requirements Specification, SRS). Спецификацию программного обеспечения часто называют техническим заданием. Это ошибка. Спецификация требований является частью технического задания в случае создания автоматизированных информационных систем.

SRS имеет следующую структуру.

- Введение
  - Цели
  - Соглашения о терминах
  - Предполагаемая аудитория и последовательность восприятия
  - Масштаб проекта
  - Ссылки на источники
- Общее описание
  - Видение продукта
  - Функциональность продукта
  - Классы и характеристики пользователей

    - Среда функционирования продукта (операционная среда)
    - Рамки, ограничения, правила и стандарты
    - Документация для пользователей
    - Допущения и зависимости
- Функциональность системы
    - Функциональный блок X (таких блоков может быть несколько)
    - Описание и приоритет
    - Причинно-следственные связи, алгоритмы (движение процессов, workflows)
- Функциональные требования
- Требования к внешним интерфейсам
    - Интерфейсы пользователя (UX)
    - Программные интерфейсы
    - Интерфейсы оборудования
    - Интерфейсы связи и коммуникации
- Нефункциональные требования
    - Требования к производительности
    - Требования к сохранности (данных)
    - Критерии качества программного обеспечения
    - Требования к безопасности системы
- Прочие требования
- Приложение А: Глоссарий
- Приложение Б: Модели процессов предметной области и другие диаграммы
- Приложение В: Список ключевых задач

Согласно с ГОСТ 34.602-89 «Техническое задание на создание автоматизированной системы» ТЗ состоит из следующих разделов.

1. Общие сведения
1.1. Наименование системы
1.2. Основания для проведения работ
1.3. Наименование организаций – Заказчика и Разработчика
1.4. Плановые сроки начала и окончания работы
1.5. Источники и порядок финансирования
1.6. Порядок оформления и предъявления заказчику результатов работ
2. Назначение и цели создания системы
2.1. Назначение системы
2.2. Цели создания системы
3. Характеристика объектов автоматизации
4. Требования к системе
4.1. Требования к системе в целом
4.2. Требования к функциям, выполняемым системой
4.3. Требования к видам обеспечения
5. Состав и содержание работ по созданию системы
6. Порядок контроля и приёмки системы

6.1. Виды и объем испытаний системы
6.2. Требования к приемке работ по стадиям
7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие
8. Требования к документированию
9. Источники разработки

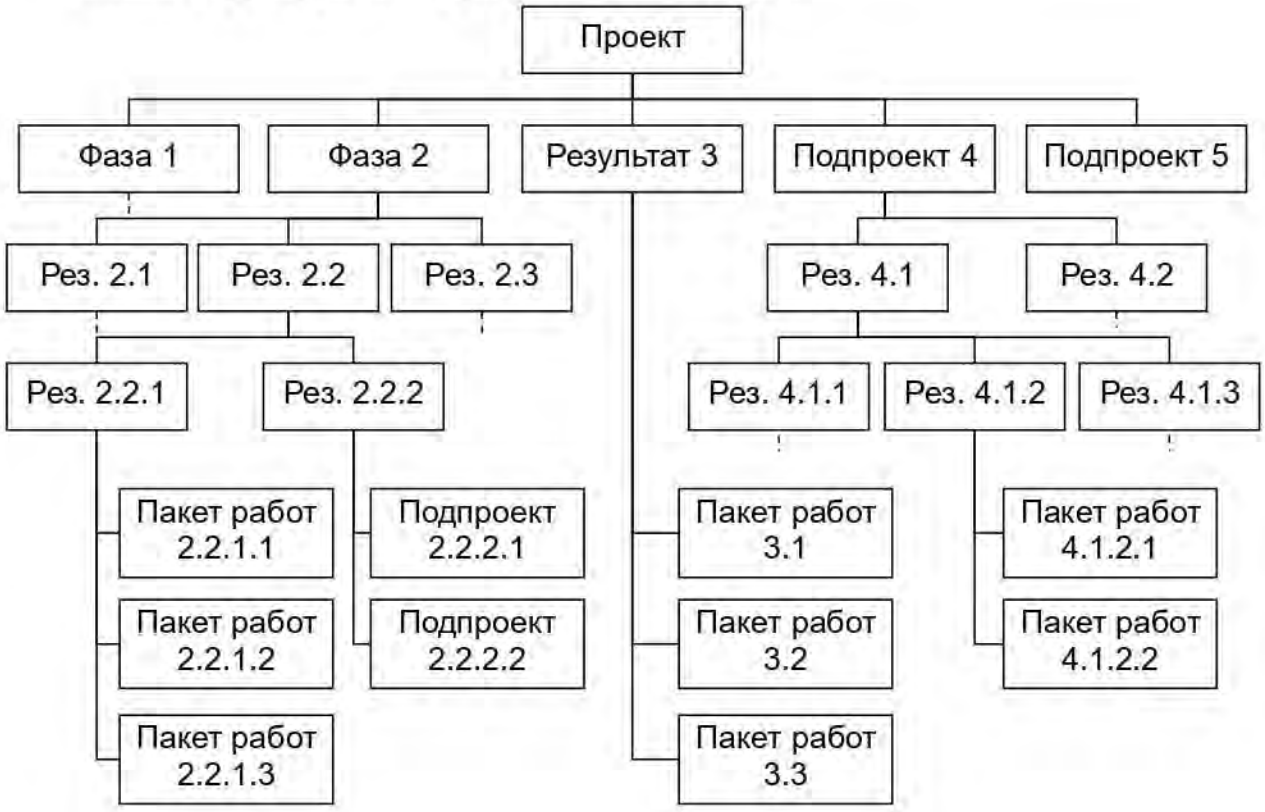
Тема №7. Планирование проекта.

Иерархическая структура работ (ИСР)

ИСР – это иерархическая декомпозиция работ, которые команда проекта должна выполнить для достижения целей проекта и создания оговоренных конечных результатов.

С ее помощью структурируется и определяется все содержание проекта. ИСР подразделяет работы проекта на более мелкие и более управляемые части, где на каждом более низком уровне дается более детальное определение проектных работ. Для запланированных работ, соответствующих элементам низшего уровня ИСР (их еще называют пакетами работ), можно определять график выполнения, сметную стоимость, осуществлять наблюдение и контроль. ИСР предыдущего проекта часто может служить шаблоном для нового проекта, поскольку некоторые проекты в той или иной степени могут быть схожи с предшествующими.

Общая структура ИСР приведена на рисунке.



Элементами верхних уровней ИСР могут выступать фазы, подпроекты, промежуточные и конечные результаты. Элементами нижнего уровня являются пакеты работ.

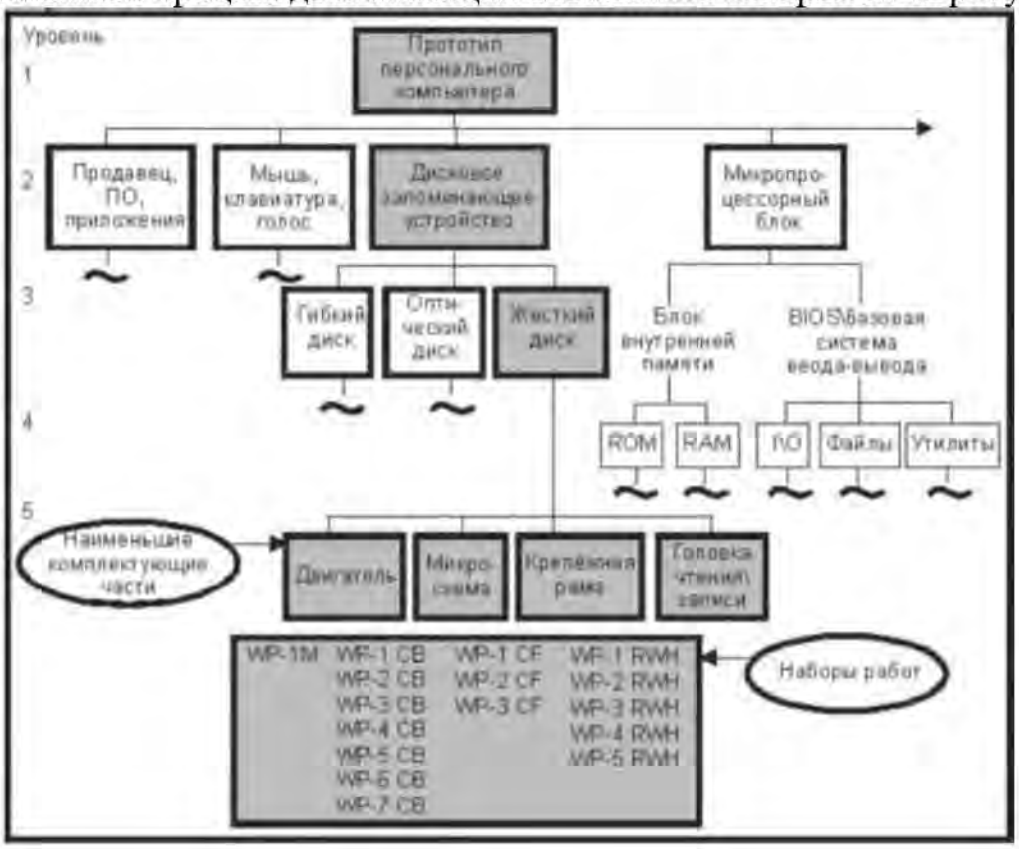
Построение ИСР производится методом декомпозиции. Вначале проект разбивается на последовательность фаз, имеющих конкретный промежуточный результат. При этом результат завершающей фазы совпадает с результатом всего проекта. Далее эти фазы подразделяются на более мелкие подфазы, заканчивающиеся созданием некоторого предварительного результата, на который опирается основной результат фазы. Подфазы, в свою очередь, разбиваются на последовательность элементарных результатов, выступающих в качестве самого низшего уровня ответственности менеджеров. Элементарный результат разделяется на последовательность пакетов работ, позволяющих проводить учет издержек, отслеживать ход работ и проводить учет ответственности исполнителей. Это – конечный этап декомпозиции ИСР. На основе пакетов работ исполнители получают рабочее задание, отличающееся от заданий других исполнителей.

Описанная схема декомпозиции изображена на рисунке.

В качестве примера рассмотрим уровни декомпозиции ИСР при разработке прототипа персонального компьютера (ПК). На первом уровне декомпозиции

ПК разбивается на набор устройств, входящих в его состав, и устанавливаемое программное обеспечение: комплект ПО, мышь, клавиатура, устройства обработки аудио и видео информации, процессорный блок, дисковые накопители информации и т.п. На втором уровне ИСР эти устройства разделяются на виды устройств, которые войдут в состав ПК. Например, из перечня возможных дисковых запоминающих устройств предполагается использовать накопитель на гибких дисках, накопитель на оптических дисках и накопитель на жестком диске. На следующем уровне каждое устройство разделяется на подвиды устройств. Для дисковых накопителей этот уровень отсутствует. Данный уровень присутствует у блока внутренней памяти, который разделяется по видам памяти RAM и ROM. На последнем уровне устройства разделяются на наименьшие комплектующие части. Для жесткого диска это – двигатель, микросхема, крепежная рамка и головка чтения/записи. Эти части образуют пакеты работ. Далее при планировании работ проекта эти пакеты разбиваются на элементарные операции, за которыми стоят конкретные исполнители.

Описанный процесс декомпозиции схематично изображен на рисунке.



Определение состава операций

К методам определения состава операций относятся: декомпозиция, шаблоны, метод набегающей волны.

Декомпозиция – разбиение проектных работ на более мелкие и более управляемые элементы, называемые плановыми операциями. Каждый пакет работ в ИСР разбивается на плановые операции, необходимые для получения

результатов этого пакета. Формулировка операции должна быть максимально конкретной и указывать на ее результат.

В качестве шаблона можно использовать список операций из предыдущего проекта, если имеются элементы совпадения или подобия текущего проекта с предыдущим.

Метод набегающей волны предполагает, что работа, которую надо будет выполнить в ближайшей перспективе, подробно планируется на низшем уровне ИСР, а далеко отстоящая работа планируется на сравнительно высоком уровне ИСР. Планирование работ на один-два ближайших периода уточняется по мере выполнения работ в текущем периоде. Поэтому на разных стадиях ЖЦ проекта плановые операции могут иметь разную степень конкретизации.

Результатами определения состава операций являются список операций, параметры операций и список контрольных событий.

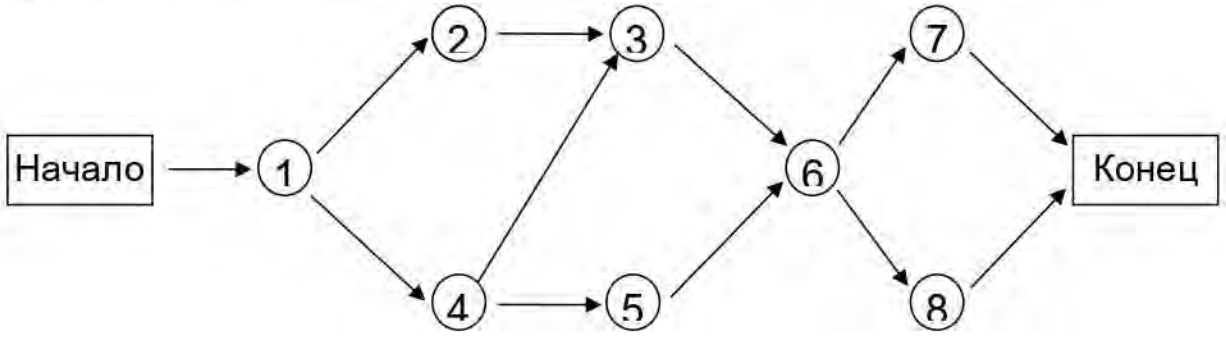
Список операций – это исчерпывающий перечень, включающий в себя все плановые операции проекта, описание содержания работ по каждой из них, подробное настолько, чтобы члены команды проекта понимали, какие работы необходимо провести. Содержание работ операции может выражаться в физических величинах. Например, количество полей в интерфейсном окне.

К параметрам операции относятся: идентификатор операции, ее описание, перечни предшествующих и последующих операций, логические взаимосвязи, опережения и задержки, требования к ресурсам, требуемые даты, ограничения и допущения. Могут включать ответственного за выполнение работы, местоположение выполнения работ и тип планирования.

Список контрольных событий задает все контрольные события расписания, указывая при этом, является ли событие обязательным (необходимым согласно контракту) или необязательным.

Определение взаимосвязей операций

Для определения взаимосвязей операций используется метод предшествования. Последовательность операций изображается в виде ориентированного графа, вершинами которого являются работы проекта (операции), а дугами – последовательность исполнения. Такой граф принято называть сетевым графиком работ. Пример сетевого графика изображен на рисунке.



Изначально в сетевых графиках использовались взаимосвязи типа «финиш - старт», когда начало следующей работы происходит не раньше, чем закончатся все непосредственно предшествующие. Впоследствии, после широкого внедрения автоматизированных систем управления проектами, стали применяться и другие виды зависимостей.

- Финиш-финиш. Завершение последующей операции зависит от завершения предшествующей операции.
- Старт-старт. Инициация последующей операции зависит от инициации предшествующей операции.
- Старт-финиш. Завершение последующей операции зависит от инициации предшествующей операции.

В некоторых случаях последующая операция должна быть начата не сразу после окончания предыдущей, а с некоторым временным смещением в ту или иную сторону. Для реализации такого эффекта в системах планирования используются задержки и опережения.

Опережение позволяет ускорить последующую операцию. Например, программист может приступить к проектированию архитектуры системы до того как системный аналитик завершит описание требований к интерфейсу. Это может быть достигнуто при помощи взаимосвязи "финиш-старт" с пятнадцатидневным опережением.

Задержка управляет приостановкой последующей операции. Например, после завершения модульного тестирования перед началом интеграционного тестирования нужно провести профилактику оборудования в течение 3 рабочих дней. Используется задержка во взаимосвязи "финиш-старт« длительностью 3 дня.

С точки зрения источника выявления связи между операциями различают обязательные зависимости, произвольные зависимости и внешние зависимости.

Обязательные зависимости – это зависимости, которые являются неотъемлемым свойством выполняемой работы. Они часто подразумевают физические ограничения. Например прототип должен быть создан до того, как он будет протестирован. Обязательные зависимости часто называют жесткой логикой.

Произвольные зависимости называют предпочитаемой логикой или мягкой логикой. Они обычно устанавливаются на основе передовых методов организации работ, где желательна особая последовательность операций, несмотря на то, что имеются и другие приемлемые последовательности. Например, гибкие методологии разработки ПО предполагают демонстрацию разработки программной системы заказчику еще до того как система будет создана полностью, чтобы как можно раньше получить обратную связь и внести изменения в требования и разрабатываемое ПО.

Внешние зависимости включают взаимоотношения операций проекта с непроектными операциями. Например, в проекте по разработке программного обеспечения сроки операции тестирования могут зависеть от поставки аппаратного обеспечения сторонней организацией.

Оценка длительностей операций

К методам оценки длительности операций проекта относят: экспертную оценку, оценку по аналогам, параметрическую оценку, оценку по трем точкам, анализ резервов.

При экспертной оценке используется накопленный опыт и историческая информация экспертов как из предыдущих, так и из текущего проекта. Этот метод может использоваться на начальных этапах планирования проекта. Качество оценок существенно зависит от наличия и опыта привлекаемых экспертов.

Оценка по аналогам предполагает использование фактической длительности аналогичной предыдущей плановой операции в качестве основы для оценки длительности будущей плановой операции. Например, длительность операции по заполнению базы данных о сотрудниках подразделения Б можно принять равным длительности уже выполненной операции по заполнению базы данных о сотрудниках подразделения А, если в этих подразделениях сопоставимая численность персонала.

Параметрическая оценка используется когда можно точно определить физический объем операции и производительность ресурса. Длительность вычисляется по формуле:

$$Длительность = \frac{физический\ объем}{производительность\ ресурса}.$$

Например, при разработке веб-сайта известно, что на создание одной страницы требуется один день. Если сайт состоит из 10 страниц, длительность операции по созданию такого сайта будет оценена в 10 дней.

Оценка по трем точкам предполагает использование трех видов оценок длительности: наиболее вероятную (НВ), оптимистичную (О) и пессимистичную (П). Наиболее вероятная – длительность операции с учетом предварительного выделения ресурсов, их производительности, реалистичной оценки их доступности, а также задержек. Оптимистичная – длительность при условии отсутствия воздействия каких-либо рисков на ход выполнения операции. Пессимистичная – длительность при условии воздействия всех возможных рисков на ход выполнения операции. Итоговое значение длительности вычисляется по формуле:

$$Длительность = \frac{О + 4 * НВ + П}{6}$$

При анализе резервов происходит добавление дополнительного времени, называемого временным резервом или буфером, в общее расписание проекта в качестве учета рисков нарушения графика. Резерв может добавляться для операций, фаз и проекта в целом. Резерв можно использовать полностью или частично, его можно впоследствии сократить или убрать вовсе.

На рисунке изображена кривая распределения вероятности времени завершения задачи. В точке минимального времени завершения задачи задача никогда не завершается, поскольку вероятность равна нулю. Эта точка задает минимально необходимое время выполнения задачи. В точке агрессивной

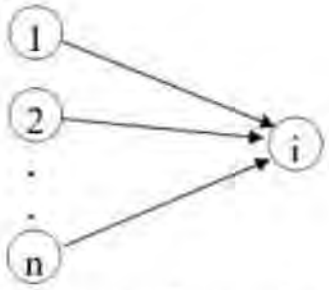
оценки задача может быть завершена с вероятностью 50%. Другими словами, в этой точке завершается только половина задач. В точке безопасной оценки задача завершается с вероятностью 85-95%. Большинство задач будет завершено к этому моменту. Крайний правый момент времени задает точку со стопроцентной вероятностью завершения.

При планировании длительности можно выбрать какой момент из трех перечисленных выбрать за время завершения задачи. Оставшееся время составляет резерв, который используется по мере необходимости.

Методы разработки расписания

При разработке расписания используются метод критического пути, сжатие расписания, анализ возможных сценариев, выравнивание ресурсов, метод критической цепи.

Метод критического пути использует предварительно построенный сетевой график работ. Первым делом вычисляется раннее время начала всех работ относительно момента начала проекта. Раннее время начала задает момент времени, раньше которого работа не может быть начата, поскольку не завершены все непосредственно предшествующие работы. Вычисление проходит прямым проходом работ от первой до последней согласно сетевому графику по следующей схеме:

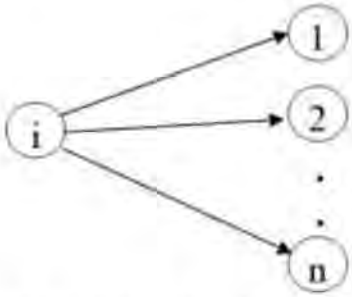


$$T_P(i) = \max \begin{pmatrix} T_P(1)+t_1 \\ T_P(2)+t_2 \\ \cdots \\ T_P(n)+t_n \end{pmatrix}$$

Далее выполняется обратный проход работ от последней к первой, в ходе которого вычисляется позднее время начала работ. Позднее время начала работы

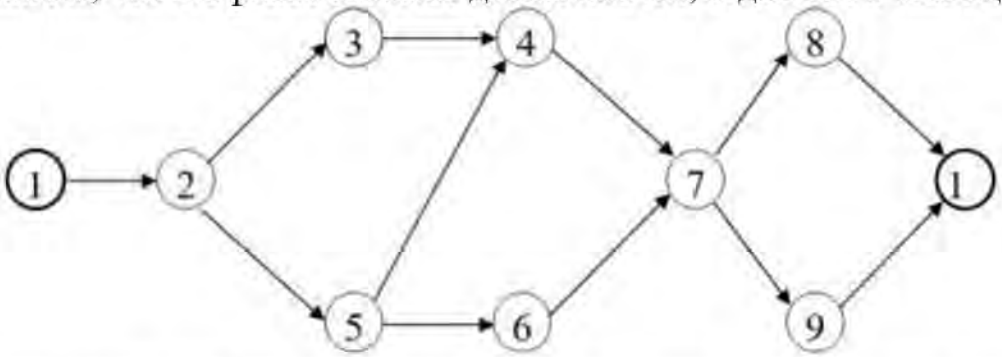
задает время, позже которого работа не может быть начата без увеличения длительности проекта в целом.

$$T_П(i) = \min \begin{pmatrix} T_П(1) - t_1 \\ T_П(2) - t_2 \\ \cdot \cdot \cdot \\ T_П(n) - t_n \end{pmatrix}$$



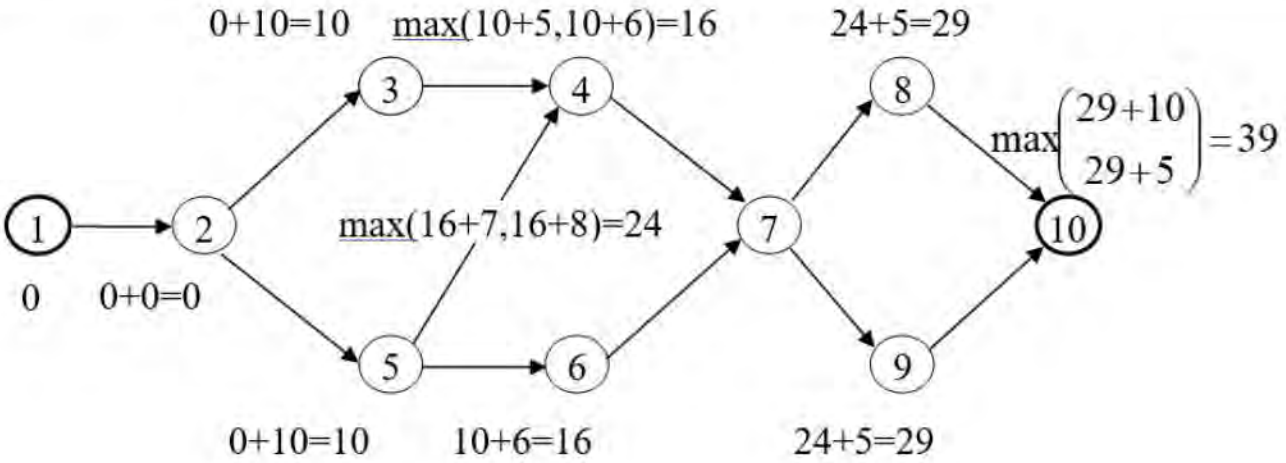
На следующем шаге вычисляются временные резервы работ как разность между поздним и ранним временем начала работ. Резерв показывает, насколько можно задержать начало работы без влияния на время выполнения всего проекта. У работ с нулевым резервом задержка начала недопустима. Такие работы называются критическими, а путь в сетевом графике от начальной до конечной работы проекта, проходящий только через критические работы, называется критическим путем. Длительность работ, лежащих на критическом пути, равна длительности выполнения всего проекта. В сетевом графике одновременно может существовать несколько критических путей, но их длительность при этом всегда будет совпадать.

Рассмотрим пример сетевого графика, изображенный на рисунке. Предположим, что его работы имеют длительности, заданные в таблице.

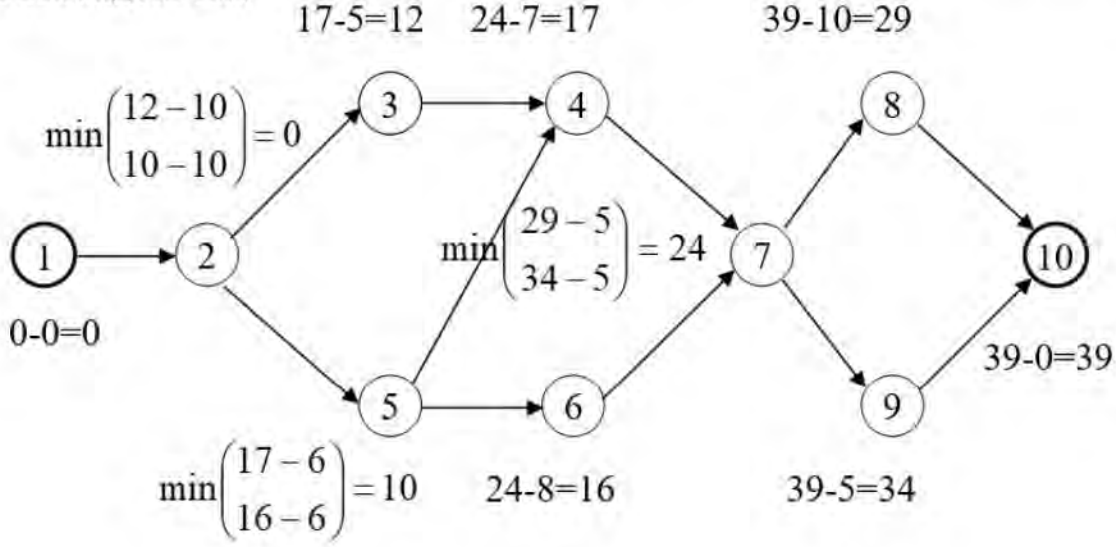


Номер работы	*Длительность*	*Номер работы*	*Длительность*
1	0	6	8
2	10	7	5
3	5	8	10
4	7	9	5
5	6	10	0

Последовательность и результаты вычисления раннего времени начала работ будут выглядеть так:



А последовательность и результаты вычисления позднего времени начала работ будут выглядеть так:



Результаты расчетов и резервы времени работ приведены в таблице, из которой следует, что критический путь проходит через работы 1, 2, 5, 6, 7, 8, 10. Работы 3, 4, 9 имеют ненулевой резерв времени, что позволяет менеджеру проектов варьировать их ресурсами с целью наиболее эффективного использования.

Работа	1	2	3	4	5	6	7	8	9	10
Раннее время начала	0	0	10	16	10	16	24	29	29	39
Позднее время начала	0	0	12	17	10	16	24	29	34	39
Резерв времени	0	0	2	1	0	0	0	0	5	0

Сжатие расписания подразумевает приемы укоротить расписание проекта без изменения его содержания, с сохранением ограничений на сроки. Обычно сжатие подразумевает дополнительные расходы с целью сократить длительность отдельных работ. Примером сжатия является увеличение числа сотрудников,

занятых на длительной операции с целью уменьшения ее длительности. При этом нужно понимать, что далеко не для всех задач, особенно из сферы информационных технологий, можно сократить длительность за счет дополнительных затрат и привлечения дополнительных исполнителей.

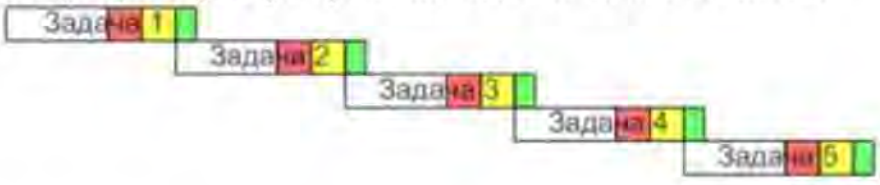
Разновидностью сжатия является быстрый проход. Здесь азы или операции, обычно выполняемые последовательно, проводятся параллельно. Например, программирование отдельных модулей до окончания формулирования требований к программному комплексу. Быстрый проход может привести к доработкам и возрастанию рисков снижения качества конечного продукта.

При анализе возможных сценариев формулируются сценарии как ответы на вопросы типа "Что произойдет, если ситуация будет развиваться по сценарию 'X'?». Для каждого сценария выполняется анализ расписания, при возникновении некоторого события (риска). Например, задержка поставки основного компонента или увеличение длительности отдельных инженерных операций. Результаты анализа могут использоваться для оценки выполнимости расписания в условиях риска и для составления планов реагирования на риски.

Выравнивание ресурсов означает выявление ситуаций неравномерного использования трудовых ресурсов и изменение расписания с целью их устранения. На практике неравномерность означает, что по ходу реализации проекта для некоторого или нескольких сотрудников работы запланированы так, что часть времени сотрудник вынужден бездействовать, поскольку у него нет работы, а в пиковые периоды его загрузка превышает допустимую дневную норму загрузки. Такие ситуации недопустимы и разрешаются коррекцией плана работ с целью устранить пиковые периоды с превышением лимита дневной загрузки сотрудников.

После выравнивания ресурсов может измениться критический путь проекта, что может привести к новому перепланированию и новой потребности в выравнивании.

Метод критической цепи является усовершенствованием классического подхода к управлению сроками. При классическом подходе каждая задача имеет резерв по времени, который на практике практически никогда не может быть перемещен на другие задачи вследствие действия синдрома студента и закона Паркинсона. На рисунке резерв времени задач выделен заливкой.



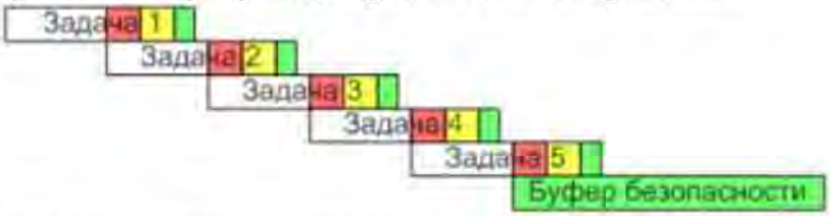
Синдром студента – начало любой операции всегда откладывается на как-можно более поздний срок. Подобно студентам, откладывающим подготовку к экзамену на последний момент, исполнители проекта всегда откладывают на последний момент начало работ над задачей проекта. Следствие синдрома

студента – окончание работы может быть задержано даже в том случае, если мы используем временной резерв, учитывающий все возможные обстоятельства!

Закон Паркинсона – работа всегда будет длиться столько времени, сколько мы на нее не отвели. Как следствие временные буферы задач не могут быть перенесены от одной задачи к другой, поскольку от них ничего не остается!

В методе критической цепи приняты следующие правила.

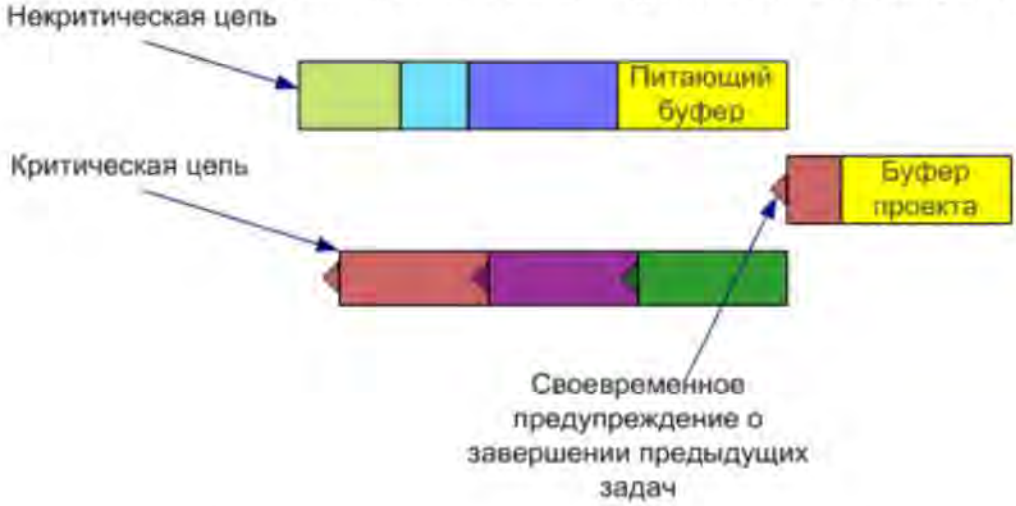
- Для работ критической цепи выделяется время, равное 50% или 0% (как это изображено на рисунке) вероятности завершения.



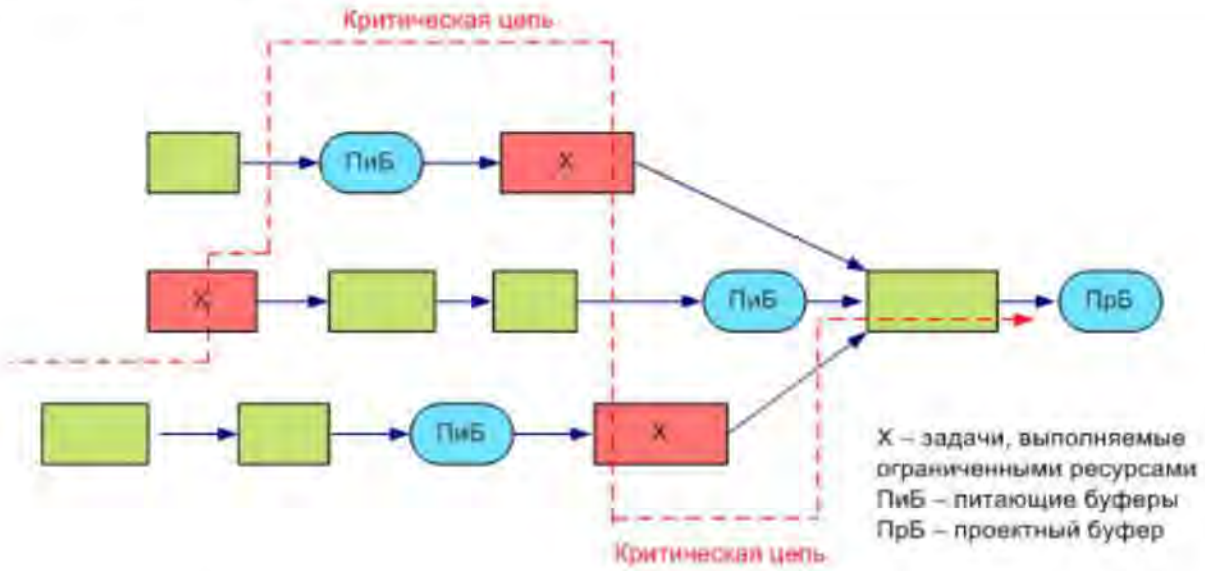
- Для работ критической цепи установленное время завершения не является обязательным.
- Оставшееся время работ критической цепи собирается в единый резерв всей цепи, который используется при невозможности завершить работу к заданному времени.

Такие правила позволяют собрать вместе временные резервы всех задач цепи в единый резерв и расходовать его по мере необходимости.

Для некритической цепи вводится питающий буфер. Этот буфер снижает риск влияния позднего завершения задач, которые не находятся на критической цепи, на критическую цепь проекта. Руководитель проекта вставляет эти буферы в те точки расписания проекта, в которых выходы задач, которые не находятся на критической цепи, являются входами задач критической цепи проекта.



Для ресурсов организуется ресурсный буфер – это предупреждение, которое направляется критическим ресурсам для того, чтобы они своевременно завершили свои текущие задачи и начали подготовку к реализации задачи критической цепи. Это делается для того, чтобы работа над критической задачей началась сразу после того, как будет выполнена работа над текущей задачей ресурса.



Тема №8. Часть 2. Риски проекта.

Риск – это неопределенное событие, которое может произойти с некоторой вероятностью и повлиять на параметры проекта: сроки исполнения операций, бюджет, качество работ, состав исполнителей и т.п. Воздействие риска может быть отрицательным, препятствующим успешному завершению проекта. Такие риски называются негативными. Риск может оказать благоприятное воздействие на проект. Такие риски называются позитивными.

Воздействие риска можно выразить через величину, в которой исчисляется значение соответствующего параметра проекта. При влиянии на длительность – в днях, на трудоемкость – в трудозатратах, на стоимость – в денежных единицах. Размер риска определяется как произведение вероятности возникновения события на степень его воздействия на параметры проекта.

Риски можно разделить на две категории – известные и неизвестные. Известные риски – это риски, которые идентифицированы при применении методов идентификации рисков. Для них можно оценить вероятность возникновения, степень воздействия на проект, величину риска, а также подготовить при необходимости план реагирования на риск, который осуществляется при реализации риска. Для неизвестных рисков подготовить план реагирования невозможно.

В качестве предупредительных мер для таких рисков создаются резервы времени и стоимости операций, фаз и проекта в целом. Резервы могут включаться в базовый план проекта или не включаться в него. В первом случае они носят характер резервов на покрытие неопределенности. Во втором случае они образуют управленческий резерв, который поступает в распоряжение менеджера проекта и расходуется по мере выполнения мероприятий по реагированию на риски.

Основной целью управления рисками является их своевременное обнаружение, снижение вероятности и величины негативных рисков, повышение вероятности и величины позитивных рисков.

Для работы с рисками в проекте используется классификация рисков. Это иерархическая структура, в которую заносятся и распределяются по категориям причины возникновения рисков. Структура предназначена для использования во всех, реализуемых организацией, проектах одной предметной области. Например, разработка программного обеспечения. Классификацию рисков можно считать элементом суммарного опыта всех предшествующих проектов. Такая классификация облегчает процесс идентификации и анализа рисков в новых проектах.

Пример подобной классификации источников рисков проектов разработки программного обеспечения, используемой в Microsoft Solutions Framework (MSF) приведен на рисунке.

К методам идентификации рисков относятся анализ документации, методы сбора информации, анализ контрольных списков, анализ допущений, методы отображения с помощью диаграмм.

Анализ документации – это поиск противоречий в проектной документации, которые сами по себе могут служить источниками рисков. Одним из основных источников являются принятые в проекте ограничения и допущения.

К методам сбора информации относятся мозговой штурм, метод Дельфи, опросы, идентификация основной причины и SWOT-анализ.

Мозговой штурм проводится с целью генерации идей относительно всех видов рисков проекта. В нем участвуют наиболее опытные сотрудники команды

проекта с возможным участием сторонних специалистов-экспертов. Генерация идей относительно рисков происходит под руководством ведущего. Запрещается критика высказываемых предположений. Генерация идей основывается на предварительно составленной классификации рисков. Все высказанные идеи протоколируются и тщательно изучаются после общего собрания на предмет обоснованности. На основании проведенного анализа формируется первоначальный реестр рисков.

Метод Дельфи – это метод анонимного опроса с целью достижения консенсуса между экспертами. Взаимодействие с экспертами ложится на роль ведущего. Предварительно составляется опросный лист, который раздается экспертом. Эксперты не знают кто еще участвует в опросе. Это делает их более свободными в своих оценках, которые не подвергаются воздействию стороннего авторитетного мнения. Ведущий собирает ответы экспертов, составляет резюме и знакомит с ним всех участников опроса. При этом каждый участник вправе скорректировать свой первоначальный ответ. Консенсус достигается в течение нескольких итераций описанного процесса.

Опросы предполагают проведение опросов с целью выявления рисков. Опросы проводятся среди опытных сотрудников, принимающих участие в проекте, участников проекта, экспертов в данной области.

Идентификация основной причины – выявление наиболее значимых причин возникновения рисков. Позволяет дать более точные определения рискам и сгруппировать риски по вызывающим их причинам. Реагирование на риски может быть эффективным только тогда, когда оно направлено на устранение основной причины возникновения риска.

SWOT-анализ – сильные и слабые стороны, возможности и угрозы. Его изображают в виде матрицы из 4-х равных клеток, обозначающих эти понятия:

Сильные стороны	**Слабые стороны**
Возможности	**Угрозы**

Сильные и слабые стороны — это внутренние параметры проекта, на которые имеется возможность влияния у команды проекта. Параметры, характеристики, условия проекта, на которые команда проекта повлиять не может, относятся к возможностям и угрозам. Обычно возможности и угрозы определяются внешним окружением проекта.

При анализе контрольных списков используются предварительно подготовленные контрольные списки, впитавшие в себя опыт реализации всех предшествующих проектов. Источниками для их создания является как опыт отдельно взятой организации, так и опыт профессиональных сообществ, опубликованный и обсуждаемый в специализированных изданиях и социальных сетях. Контрольный список позволяет легко проконтролировать все ли действия по идентификации рисков выполнены, нет ли упущенных моментов. По завершении проекта списки подвергаются переосмыслению и, при

необходимости дополняются и корректируются, впитывая в себя опыт только что завершенного проекта.

Задачей анализа допущений является проверка их обоснованности, полноты и непротиворечивости, а также соответствия текущего плана и хода реализации проекта принятым ранее допущениям. Подобный анализ нередко позволяет выявить потенциальные источники рисков.

Методы отображения с помощью диаграмм предполагают использование диаграмм причинно-следственных связей и диаграмм зависимостей процесса.

Результатом идентификации рисков является реестр рисков, в который входят следующие данные.

- Список идентифицированных рисков. Это перечень событий, влекущих за собой риски проекта, оценка вероятности реализации события и степени воздействия риска на параметры проекта.
- Основные причины возникновения риска – причины возникновения соответствующих событий.
- План реагирования на риски. Может представлять из себя как определенные разовые мероприятия, так и резервный план реализации всего проекта.

Анализ рисков принято разделять на качественный и количественный. В задачу качественного анализа входит ранжирование рисков по степени важности и потенциального воздействия на проект, выделение рисков для своевременного реагирования и для наблюдения. Количественный анализ проводится только для тех рисков, которые на этапе качественного анализа были признаны существенными и требующими реагирования. Его задача – определить влияние риска в величинах параметров проекта (днях, трудозатратах, денежных единицах).

Для качественного анализа рисков используют следующие методы: определение вероятности и воздействия рисков, матрица вероятностей и последствий, оценка качества данных риска, оценка срочности риска.

Определение вероятности и воздействия рисков – проведение анализа для определения уровня вероятности возникновения того или иного специфического риска проекта. Определяется потенциальный эффект, который риск может оказать на цели проекта (время, стоимость, содержание или качество), включая отрицательные воздействия для угроз и положительные воздействия для благоприятных возможностей. Вероятность и воздействие оцениваются для каждого выявленного риска.

Матрица вероятностей и последствий содержит комбинации вероятностей и уровня воздействия, при помощи которых рискам присваивается определенный интегральный ранг: низкий, средний или высокий приоритет. Определяет, какие комбинации вероятности и воздействия соответствуют высокому риску, среднему риску или малому риску.

Оценка качества данных риска – проверка достоверности информации. Использование данных о риске из недостоверных источников может привести к

ошибкам на этапе качественного анализа. В результате несущественный риск можно принять за значимый и, наоборот, значимый риск – за не существенный.

Оценка срочности риска. Риски, требующие незамедлительного реагирования, рассматриваются как наиболее срочные для принятия ответных мер. Для таких рисков следует разработать мероприятия по реагированию и приступить к осуществлению этих мероприятий.

К методам количественного анализа относят опрос, анализ ожидаемой денежной стоимости, моделирование и имитацию, дерево решений.

Опрос используются для оценки значения вероятности наступления и воздействия рисков на цели проекта. Пример оценок по трем точкам для стоимостной оценки приведен в таблице.

Диапазон оценок стоимости риска			
Элемент ИСР	Низкая	Наиболее вероятно	Высокая
Проектирование	4	6	10
Программирование	16	20	35
Тестирование	11	15	23
Итог для проекта	31	41	68

Анализ ожидаемой денежной стоимости (ОДС) проводится в точках, имеющих несколько вариантов развития событий.. Пусть имеется несколько таких вариантов. Событие S1 имеет вероятность P1, S2 – вероятность P2, S3 – вероятность P3. Тогда ОДС вычисляется по формуле

$$ОДС = S1*P1 + S2*P2 + S3*P3.$$

ОДС благоприятных возможностей положительно. ОДС неблагоприятных возможностей отрицательно.

При моделировании и имитации используется имитационная модель для определения последствий от воздействия рисков на результаты проекта в целом. Проводится многократное моделирование, в результате которого рассчитывается некоторое итоговое распределение вероятностей целевых показателей (например, общая стоимость или дата завершения проекта).

Дерево решений отображает рассматриваемую ситуацию с учетом каждой из имеющихся возможностей выбора и возможного сценария. Пример дерева решений изображен на рисунке. Здесь изображена альтернатива из двух решений: разрабатывать новый продукт или модифицировать старый. Для каждой из альтернатив вычисляется ОДС. ОДС для варианта модификации старого продукта оказался выше. Следует принять решение – модифицировать старый продукт.

Стратегиями реагирования на негативные риски являются эскалация, уклонение, передача, снижение и принятие.

Эскалация – команда признает, что риск находится вне сферы ее влияния, и передает ответственность за риск на более высокий уровень организации, где управление риском будет более результативным.

Уклонение – полное исключение воздействия риска на проект за счет изменений характера проекта или плана управления проектом. Некоторые риски, возникающие на ранних стадиях проекта, например, из-за отсутствия четкого определения требований заказчика, можно избежать, затратив дополнительное время и увеличив трудозатраты на их выявление. Эта стратегия не может полностью исключить риск.

Передача – исключает угрозу риска путем передачи негативных последствий риска с ответственностью за реагирование на риск на третью сторону. Передача риска обычно сопровождается выплатой премии за риск стороне, принимающей риск и ответственность за его управление. Сам риск при этом не устраняется.

Снижение – предполагает действия, направленные на понижение вероятности и/или последствий риска до приемлемых пределов. Используется включение в план проекта дополнительной работы, которая будет выполняться независимо от возникновения риска. Например, проведение дополнительного тестирования функциональности информационной системы, разработка прототипа системы, дополнительное подключение к работе опытных сотрудников.

Принятие. Решение команды не уклоняться от риска. При пассивном принятии риска команда ничего не предпринимает в отношении риска и в случае его возникновения разрабатывает способ его обхода или исправления

последствий. При активном принятии риска план действий разрабатывается до того, как риск может произойти и называется планом действий в непредвиденных обстоятельствах.

Стратегиями реагирования на позитивные риски являются эскалация, использование, совместное использование, усиление и принятие.

Эскалация – команда признает, что риск находится вне сферы ее влияния, и передает ответственность за риск на более высокий уровень организации, где управление риском будет более результативным.

Использование – устранение всех неопределенностей, связанных с риском, при помощи мер, обеспечивающих обязательное появление данной благоприятной возможности в различных формах. Привлечение к участию в проекте более талантливого и квалифицированного персонала, чтобы максимально эффективно использовать имеющуюся благоприятную возможность.

Совместное использование – передача ответственности третьей стороне, которая способна наилучшим образом и с наибольшей выгодой использовать благоприятную возможность в интересах проекта.

Усиление. При применении этой стратегии изменяется "размер" благоприятной возможности путем увеличения вероятности возникновения и/или положительного воздействия, а также путем выявления и максимизации основных источников этих положительных рисков.

Принятие. Команда не имеет способов усиления действия благоприятной возможности и принимает ее такой, какая она есть. При пассивном принятии риска команда ничего не предпринимает в отношении риска и в случае его возникновения разрабатывает способ использования создавшихся возможностей. При активном принятии риска план действий разрабатывается до того, как риск может произойти и называется планом действий в непредвиденных благоприятных обстоятельствах.

Тема №13. Управление качеством проекта.

Качество – это степень, в которой совокупность внутренних характеристик объекта соответствует заданным требованиям.

Управление качеством проекта направлено как на управление проектом, так и на поставляемые результаты проекта.

Выделяются следующие процессы управления качеством.

1. Планирование качества – определение каким стандартам качества должен удовлетворять проект и его результаты, составление плана мероприятий по обеспечению и контролю качества.
2. Процесс обеспечения качества – реализация запланированных мероприятий с целью обеспечения соответствия проекта и его результатов намеченным стандартам и показателям качества.
3. Процесс контроля качества – проверка и мониторинг операций проекта и результатов проекта на соответствие установленным стандартам и

показателям качества, выявление фактов и идентификация причин несоответствия, устранение вызвавших несоответствие причин.

Современные подходы к управлению качеством проекта исходят из следующих положений.

1. Удовлетворение потребностей заказчика. Следует понимать, что формальным выражением потребностей заказчика выступает описание требований, включенное в техническое задание по проекту. Во многих случаях это описание не является полным и не полностью соответствовать реальным потребностям потребителей. Результаты проекта должны сочетать как выполнение заявленных требований, так и пригодность к использованию, т.е. удовлетворять реальным потребностям потенциальных пользователей.
2. Предотвращение важнее инспектирования. Затраты на превентивные меры, направленные на ликвидацию источников низкого качества и возникновения ошибок, всегда значительно ниже затрат по устранению обнаруженных ошибок в ходе проведения инспекций качества.
3. Ответственность руководства. Для успеха проекта необходимо участие и максимальная отдача всей команды проекта. Обязанность руководства проекта – обеспечение его всеми необходимыми ресурсами, которые могут понадобиться для достижения наилучшего результата.
4. Постоянное совершенствование. Непрерывно повторяющийся цикл "планирование – исполнение – проверка – воздействие" – это основа повышения качества. Этот цикл предполагает проведение регулярных мероприятий по планированию, исполнению и контролю за самим процессом. В случае обнаружения ошибок и неудовлетворительного исполнения отыскиваются и устраняются причины. Тем самым организация совершенствует свой процесс, повышая качество как текущего, так и всех последующих проектов.

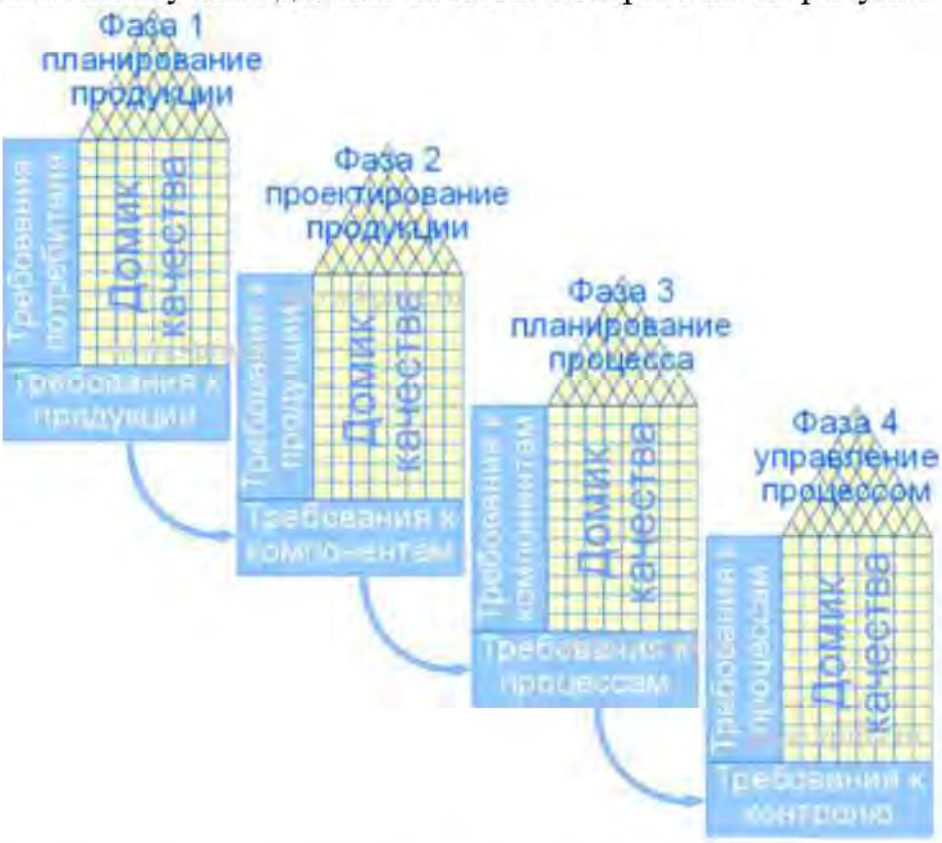
В современных ИТ-проектах активно применяется технология развертывания функций качества QFD.

Развертывание функций качества – это системный подход к проектированию, задачей которого является изучение и ранжирование реальных потребностей потребителей, определение какие из технических характеристик результатов проекта наилучшим образом удовлетворяют этим потребностям, планирование и управление процессом таким образом, чтобы реализовать наиболее важные для потребителя технические характеристики.

Развертывание функций качества позволяет решить ряд важных задач при создании продукта проекта.

- Ранжировать пожелания и ожидания потребителя, выделить наиболее существенные из них.
- Определить перечень технических спецификаций, наилучшим образом соответствующих наиболее важным пожеланиям и ожиданиям потенциальных пользователей.
- Создать такой результат проекта, который будет соответствовать установленным техническим спецификациям, а посредством них – наиболее важным пожеланиям потребителей.

Технология QFD включает в себя 4 фазы, на каждой из которых применяется матричная диаграмма особого вида – домик качества. Обобщенная схема QFD и используемые домики качества изображены на рисунке.



На каждой фазе производится сопоставление входных параметров с выходными. Оценивается приоритет входных параметров, степень их взаимосвязи с входными параметрами, насколько тот или иной выходной параметр позволяет удовлетворить требования, сформулированные в виде набора входным параметров. На основании оценок подобной взаимосвязи производится ранжирование выходных параметров по важности. Наиболее важные из них переходят на следующую фазу в качестве входных параметров ее домика качества.

Технология QFD позволяет сформулировать требования к продукции на основе пожеланий пользователей, требования к компонентам на основании требований к продукции, требования к процессам на основании требований к компонентам и требования к контролю на основании требований к процессам.

Фаза 1 – планирование продукции. Домик качества помогает перевести пожелания и требования потребителя в технические характеристики изделия. Здесь должны участвовать специалисты, знакомые с потребностями конечных потребителей, а также знакомые с методами изучения пользовательского спроса. Эта фаза очень важна, так как от полученных результатов будет зависеть как эффективность применения технологии QFD в целом, так и полезность созданного проектом конечного продукта.

Фаза 2 – проектирование продукции. Здесь за дело берутся технические специалисты (проектировщики, архитекторы ПО, разработчики ПО). Их задача

– выработать общую концепцию будущего продукта, выделить его основные части (компоненты), определить порядок взаимодействия компонент и интерфейсы между ними. Второй домик качества используется для отображения входных технических характеристик продукта в выходные технические требования и спецификации компонентов.

Фаза 3 – планирование процесса. В ней участвуют прежде всего менеджеры проекта, а также разработчики и тестировщики программного обеспечения. Задача данной фазы – спланировать и организовать процесс разработки так, чтобы наиболее оптимальным образом, с наименьшими затратами времени и финансов, но при достаточном уровне качества, реализовать компоненты с заданными техническими характеристиками. Третий домик позволяет увязать исходные параметры компонентов с параметрами и параметрами и планом процесса разработки.

Фаза 4 – управление процессом. На этом этапе в работах участвуют менеджеры проекта, менеджеры по качеству, разработчики и тестировщики ПО. На основании требований к процессам разрабатываются требования и процедуры контроля, которые будут регулярно применяться для мониторинга качества при реализации проекта.

Рассмотрим подробный пример построения домика качества для фазы 1 на примере проектирования печатающего устройства.

Шаг 1. Определение требований потребителя.

Требования вносятся в матрицу домика качества в раздел требований потребителя. Требования могут быть структурированы по видам, например, эргономичность, исполнение и т.п. Результат приведен в таблице.

Эргоном	Легко вставить картридж							
	Легко активировать							
	Легко подключить							
	Не требует настроек							
Исполне	Малый вес							
	Разные цвета корпуса							
	Безопасность в работе							
	Не ломается при падении							

Шаг 2. Определение важности требований для потребителей.

Для определения рейтинга важности может применяться шкала от 1 до 5, где 5 означает максимальную важность, а 1 минимальную важность. Чтобы ранжировать требования потребителей по степени важности применяют матрицу приоритетов или метод консенсуса. Результат заносится в домик качества.

Эргоном	Легко вставить картридж	3						
	Легко активировать	3						
	Легко подключить	5						
	Не требует настроек	4						
Ис	Малый вес	2						
	Разные цвета корпуса	1						

	Безопасность в работе	4								
	Не ломается при падении	3								

Шаг 3. Определение конкурентного рейтинга потребителя.

Конкурентный рейтинг дает возможность установить конкурентные преимущества разрабатываемого продукта или услуги в сравнении с аналогичными. Для сравнения выбираются продукты нескольких компаний и проводится их сравнительная оценка. Результат заносится в домик качества в раздел рейтинга потребителя.

										Рейтинг потребителя				
										Н – наша продукция А–компания А Б–компания Б				
										1	2	3	4	5
Эргоном	Легко вставить картридж	3								Б	Н	А		
	Легко активировать	3									А		Н	Б
	Легко подключить	5								Н			Б	А
	Не требует настроек	4								Б	Н		А	
Исполне	Малый вес	2								Б		А	Н	
	Разные цвета корпуса	1									Б	А	Н	
	Безопасность в работе	4									Н	Б	А	
	Не ломается при падении	3								А		Б	Н	

Шаг 4. Определение технических требований.

На данном шаге в домик качества включается перечень технических требований к проектируемому изделию.

Шаг 5. Построение матрицы взаимосвязи.

Для выявления силы взаимосвязи применяется шкала значений 9, 3, 1, где 9 означает сильную взаимосвязь, 3- среднюю, 1 – слабую.

Результатом всех пяти шагов является полностью заполненный домик качества, приведенный в таблице. Далее после суммирования столбцов важности технических требований ранжируем эти требования по набранной сумме. Имеем.

1. Соответствие стандартам – 24.
2. Количество цветов – 9.
3. Усилие включения – 9.
4. Количество типоразмеров – 5.
5. Количество систем – 4.
6. Количество интерфейсов – 3.
7. Температурный диапазон – 3.
8. Вес материалов – 2.

К методам процесса обеспечения качества относятся следующие.

1. Аудит качества. Независимая экспертная оценка, определяющая, насколько операции проекта соответствуют установленным в рамках проекта или организации правилам, процессам и процедурам. Выполняется сторонней организацией, стоит дорого, но позволяет получить независимую объективную оценку качества проекта.
2. Анализ процесса. Предусматривает выполнение действий, направленных на выявление нуждающихся в улучшении моментов с технической и организационной точек зрения. Выполняется командой проекта, вследствие чего стоит существенно дешевле, нежели аудит качества.

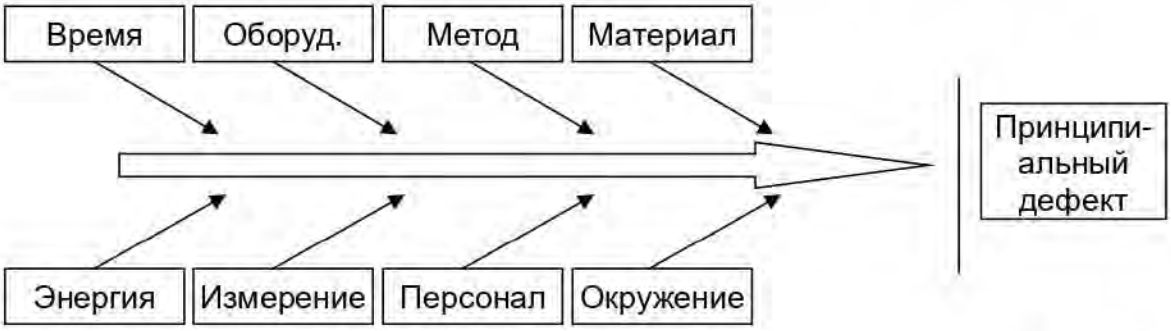
		Важность потребителей	Программные средства			Размеры		Особенности			Рейтинг потребителя				
			Соответствие стандартам	Кол-во интерфейсов	Кол-во систем	Кол-во цветов	Кол-во типоразмеров	Температурный диапазон	Вес материалов	Усилие включения	Н – наша продукция А–компания А Б–компания Б				
											1	2	3	4	5
Эргономичность	Легко вставить картридж	3					1				Б	Н	А		
	Легко активировать	3	3		1					9		А		Н	Б
	Легко подключить	5	9	3	3						Н			Б	А
	Не требует настроек	4	9								Б	Н		А	
Исполнение	Малый вес	2					1		1		Б		А	Н	
	Разные цвета корпуса	1				9						Б	А	Н	
	Безопасность в работе	4	3				3	3				Н	Б	А	
	Не ломается при падении	3							1		А		Б	Н	

К методам процесса контроля качества относятся.

1. Диаграммы причинно-следственных связей.
2. Контрольные диаграммы.
3. Диаграммы зависимостей.

4. Гистограмма.
5. Диаграмма Парето.
6. Диаграмма разброса.
7. Выборочные оценки.
8. Инспекция.
9. Проверка исправления дефектов.
10. Тестирование и верификация программного обеспечения.

Диаграмма причинно-следственных связей изображает связь разнообразных ранее установленных факторов с возможными проблемами или эффектами. Эту диаграмму называют также диаграммой рыбьего скелета или диаграммой Ишикавы. По одну сторону диаграммы изображаются потенциальные причины, вызвавшие когда-либо в прошлом некий принципиальный дефект. По другую сторону – сам принципиальный дефект, когда-либо проявившийся при реализации проекта. Данная диаграмма выступает в качестве средства накопления опыта и знаний организации в процессе многолетней работы над ИТ проектами. Диаграмма изображена на рисунке.



Контрольные диаграммы используются в тех случаях, когда имеется некоторый процесс, в ходе которого требуется отслеживать изменение значений некоторой величины. Диаграмма позволяет получить визуальное представление о развитии процесса в течение определенного времени. Помогает выяснить, как произведенные изменения повлияли на улучшение или ухудшение процесса. Пример диаграммы приведен на рисунке.

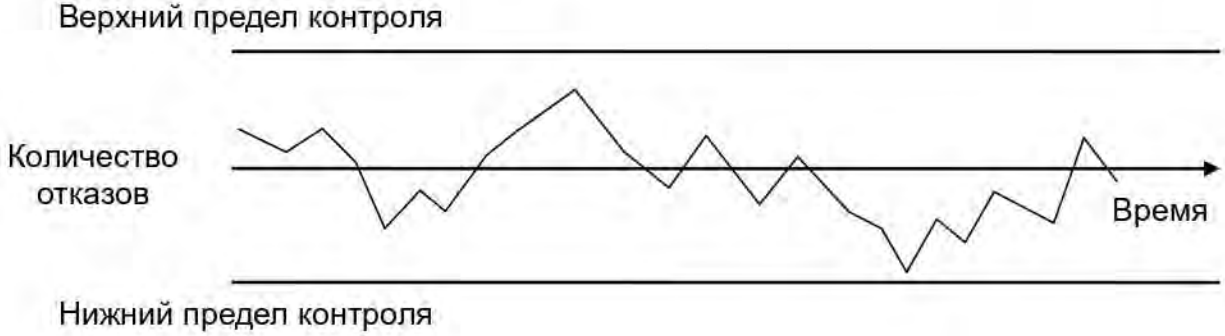
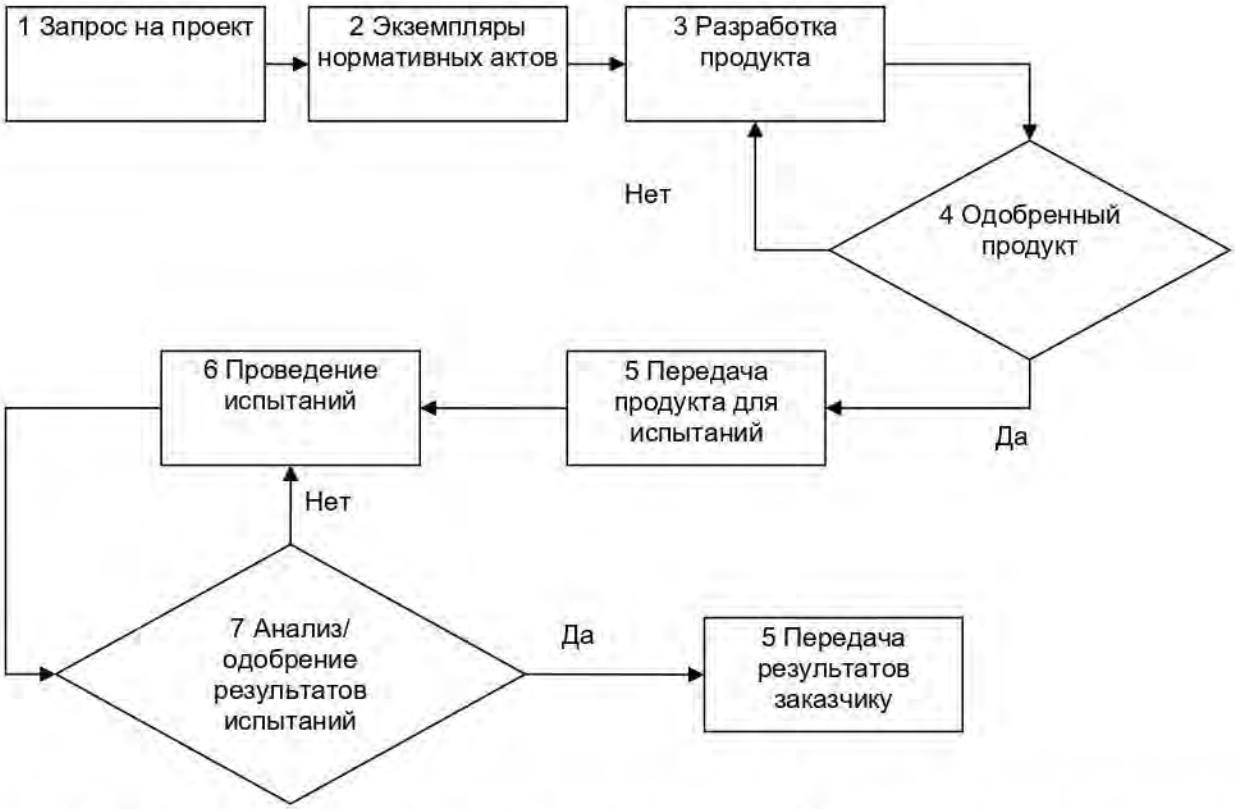


Диаграмма зависимостей – это графическое изображение процесса, в котором используются операции, точки принятия решений и последовательность

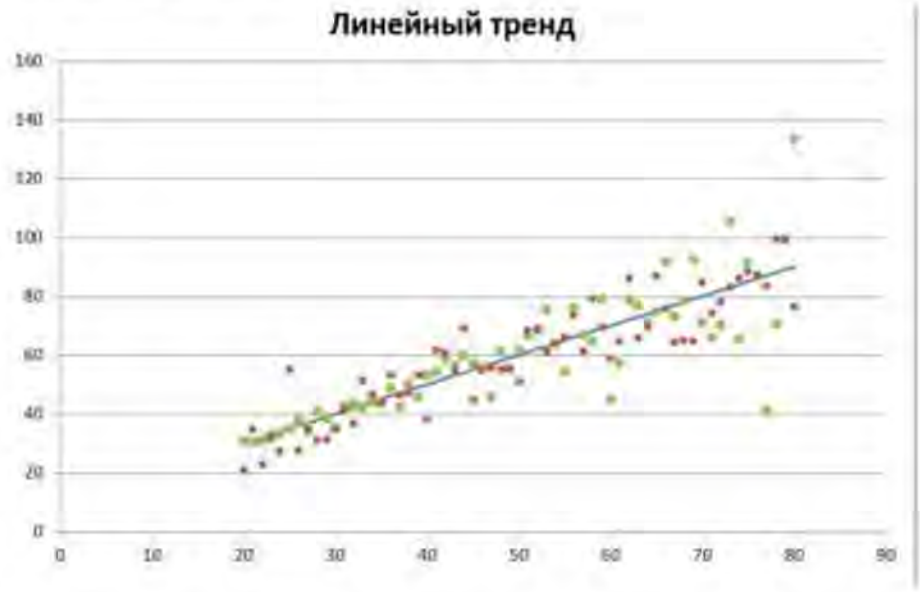
обработки данных. Дает представление о взаимодействии элементов системы между собой. Пример диаграммы приведен на рисунке.



Гистограмма - столбиковая диаграмма, отображающая распределение некоторых переменных. Позволяет графически отобразить частоту или удельный вес наблюдающихся значений друг относительно друга.

Диаграмма Парето – гистограмма, упорядоченная по частоте возникновения. Связана с законом Парето (принцип 80/20), согласно которому 80 процентов проблем создается 20 процентами причин. Диаграмма Парето позволяет выявить наиболее часто встречающиеся причины неудовлетворительного качества с тем, чтобы направить основные ресурсы команды проекта на устранение именно этих причин, не отвлекаясь на ликвидацию более мелких и несущественных проблем.

Диаграмма разброса отображает модель взаимодействия между двумя величинами. При помощи этого инструмента можно проводить изучение и определять возможные зависимости между значениями двух переменных. На рисунке приведена диаграмма разброса, показывающая наличие прямой линейной зависимости между двумя исследуемыми величинами.



Выборочные оценки. Базируется на использовании методов математической статистики. Имеется некоторая достаточно большая совокупность объектов, качество которых должно быть проверено и измерено. Для сокращения времени и затрат на проверку из этой совокупности объектов случайным образом выбирается определенное подмножество, которое и подлежит проверке. Результат проверки может быть экстраполирован на всю совокупность с некоторой доверительной вероятностью. Значение доверительной вероятности определяет размер выборки, которая подлежит реальной проверке.

Инспекция – изучение работы продукта или процесса его разработки с целью определить соответствуют ли они установленным стандартам качества. При этом следует понимать, что предотвращение возникновения дефектов гораздо дешевле инспектирования с последующим их устранением.

Проверка исправления дефектов предпринимается с целью убедиться, что дефекты продукта или процесса, обнаруженные в ходе инспектирования, были исправлены.

Все виды тестирования программного обеспечения можно условно разделить на следующие группы:

1. Функциональные
2. Нефункциональные
3. Связанные с изменениями

Функциональные тесты основаны на знаниях какие функции выполняет ПО, какие бизнес-процессы автоматизируются, какие исходные данные использует ПО и какие результаты должны быть получены, как ПО взаимодействует со смежным ПО, оборудованием и операционной системой. К ним относятся:

- функциональное тестирование,
- тестирование безопасности,
- тестирование взаимодействия.

Нефункциональное тестирование содержит тесты, задачей которых является определение как работает система и выразить результаты через некоторые количественные характеристики: время отклика, время обработки запроса, количество одновременно работающих пользователей и т.п. При этом измерению подвергаются не функции, выполняемые системой, а количественные характеристики ее функционирования. К нефункциональным видам тестирования относятся:

- все виды тестирования производительности,
- тестирование установки,
- тестирование удобства пользования,
- тестирование на отказ и восстановление,
- конфигурационное тестирование.

Задачей тестирования, связанного с изменениями, является проверка работы системы после того как в нее внесены какие-либо изменения (исправление ошибок, расширение функциональности, переработка каких-либо компонент). К такому виду тестирования относятся:

- дымовое тестирование,
- регрессионное тестирование,
- тестирование сборки,
- санитарное тестирование.

Одним из основных источников для составления функциональных тестов являются техническое задание на разработку программной системы и описание требований к системе SRS. Именно эти документы описывают разнообразные условия, требования к функционалу, порядок применения и функционирования программной системы, которые должны быть протестированы с применением функционального тестирования. Тестирование функциональности может проводиться в двух аспектах: требований и бизнес-процессов.

Тестирование в аспекте требований использует спецификацию функциональных требований к системе как основу для дизайна тестовых случаев. В этом случае необходимо сделать список того, что будет тестироваться, а что нет, расставить приоритеты требований, а на их основе определить приоритеты тестовых сценариев. Такой подход позволит не упустить при тестировании наиболее важный функционал.

Тестирование в аспекте бизнес-процессов использует знание самих бизнес-процессов, которые описывают сценарии ежедневного использования системы. Здесь тестовые сценарии основываются на практических вариантах использования системы.

Преимуществом функционального тестирования является то, что оно имитирует фактическое использование системы. Недостатки заключаются в том, что при таком тестировании можно упустить логические ошибки в программном обеспечении, если пропустить какой-либо сценарий. При функциональном тестировании возможно избыточное тестирование функционала, задействованного в нескольких текстовых сценариях.

Тестирование безопасности – это стратегия тестирования, используемая для проверки безопасности системы, а также для анализа рисков, связанных с обеспечением целостного подхода к защите приложения, атак хакеров, вирусов, несанкционированного доступа к конфиденциальным данным. Общая стратегия безопасности основывается на трех основных принципах: конфиденциальность, целостность, доступность.

Конфиденциальность – это сокрытие определенных ресурсов или информации, т.е. ограничение доступа к ресурсу некоторой категории пользователей.

Существует два основных критерия при определении понятия целостности: доверие; повреждение и восстановление.

Доверие – ожидается, что ресурс будет изменен только соответствующим способом определенной группой пользователей.

Повреждение и восстановление – в случае, когда данные повреждаются или неправильно меняются авторизованным или не авторизованным пользователем, мы должны определить на сколько важной является процедура восстановления данных.

Доступность представляет собой требования о том, что ресурсы должны быть доступны авторизованному пользователю, внутреннему объекту или устройству. Как правило, чем более критичен ресурс, тем выше должен быть уровень его доступности.

Тестирование взаимодействия проверяет насколько успешно и эффективно тестируемое ПО взаимодействует со смежным программным обеспечением, оборудованием и компонентами операционной системы, предусмотренными в техническом задании. Высокая степень взаимодействия облегчает процессы интеграции и внедрения разработанного ПО при минимальном уровне времени и затрат. Тестирование взаимодействия включает в себя тестирование совместимости и интеграционное тестирование.

Нагрузочное тестирование или тестирование производительности – это автоматизированное тестирование, имитирующее работу определенного количества бизнес-пользователей на каком-либо общем ресурсе.

Задачей тестирования производительности является определение масштабируемости приложения под нагрузкой, при этом происходит:

- измерение времени выполнения выбранных операций при определенных интенсивностях выполнения этих операций,
- определение количества пользователей, одновременно работающих с приложением,
- определение границ приемлемой производительности при увеличении нагрузки (при увеличении интенсивности выполнения этих операций),
- исследование производительности на высоких, предельных, стрессовых нагрузках.

Стрессовое тестирование позволяет проверить насколько приложение и система в целом работоспособны в условиях стресса и также оценить способность системы к регенерации, т.е. к возвращению к нормальному состоянию после прекращения воздействия стресса. Стрессом может быть

повышение интенсивности выполнения операций до очень высоких значений или аварийное изменение конфигурации сервера.

Задачей объемного тестирования является получение оценки производительности при увеличении объемов данных в базе данных приложения.

Задачей тестирования стабильности (надежности) является проверка работоспособности приложения при длительном (многочасовом) тестировании со средним уровнем нагрузки.

Тестирование установки направленно на проверку успешной инсталляции и настройки, а также обновления или удаления программного обеспечения.

Тестирование удобства пользования - это метод тестирования, направленный на установление степени удобства использования, обучаемости, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий.

Тестирование удобства пользования дает оценку уровня удобства использования приложения по следующим пунктам:

- производительность, эффективность – сколько времени и шагов понадобится пользователю для завершения основных задач приложения, например, размещение новости, регистрации, покупка и т.д.;
- правильность – сколько ошибок сделал пользователь во время работы с приложением;
- активизация в памяти – как много пользователь помнит о работе приложения после приостановки работы с ним на длительный период времени (повторное выполнение операций после перерыва должно проходить быстрее чем у нового пользователя);
- эмоциональная реакция – как пользователь себя чувствует после завершения задачи (растерян, испытал стресс), порекомендует ли он систему своим друзьям (положительная реакция - лучше).

Тестирование на отказ и восстановление проверяет тестируемый продукт с точки зрения способности противостоять и успешно восстанавливаться после возможных сбоев, возникших в связи с ошибками программного обеспечения, отказами оборудования или проблемами связи. Такое тестирование очень важно для систем, работающих по принципу “24x7”.

Методика подобного тестирования заключается в симулировании различных условий сбоя и последующем изучении и оценке реакции защитных систем. В процессе подобных проверок выясняется, была ли достигнута требуемая степень восстановления системы после возникновения сбоя.

Конфигурационное тестирование — специальный вид тестирования, направленный на проверку работы программного обеспечения при различных конфигурациях системы (заявленных платформах, поддерживаемых драйверах, при различных конфигурациях компьютеров и т.д.).

Понятие дымового тестирования пошло из инженерной среды: «При вводе в эксплуатацию нового оборудования считалось, что тестирование прошло удачно, если из установки не пошел дым».

В области программного обеспечения, дымовое тестирование рассматривается как короткий цикл тестов, выполняемый для подтверждения того, что после сборки кода (нового или исправленного) устанавливаемое приложение стартует и выполняет основные функции.

Регрессионное тестирование – это тестирование, которое проводится после внесения в систему каких-либо изменений с целью устранения ошибок, доработки, расширения функциональности и т.п. Задачей такого тестирования является проверка правильности работы ранее уже отлаженного функционала. Все то, что успешно работало до внесения изменений, должно также успешно работать и после них. Регрессионное тестирование обычно проводится в автоматизированном режиме. Для этого заранее разрабатываются наборы автоматизированных тестов, проверяющие различные функции системы. После внесения изменений такие тесты дают полное представление о работе всех функциональных модулей и наличии каких-либо проблем.

Как правило, для регрессионного тестирования используются тест кейсы, написанные на ранних стадиях разработки и тестирования. Это дает гарантию того, что изменения в новой версии приложения не повредили существующую функциональность.

Выделяют 3 основных типа регрессионного тестирования:

- регрессия багов – попытка доказать, что исправленная ошибка на самом деле не исправлена;
- регрессия старых багов – попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок, т.е. старые баги стали снова воспроизводиться;
- регрессия побочного эффекта – попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

Тестирование сборки – тестирование, направленное на определение соответствия выпущенной версии критериям качества до начала детального тестирования. По своим целям является аналогом дымового тестирования, направленного на приемку новой версии в дальнейшее тестирование или эксплуатацию. Вглубь оно может проникать дальше, в зависимости от требований к качеству выпущенной версии.

Санитарное тестирование решает задачу доказать, что конкретно взятая функция или программный модуль работает именно так, как этого требует техническое задание и описание требований к системе. Обычно выполняется с применением ручного тестирования.

Тестирование на разных уровнях производится на протяжении всего жизненного цикла разработки и сопровождения программного обеспечения. Уровень тестирования определяет то, над чем производятся тесты: над отдельным модулем, группой модулей или системой, в целом. Проведение тестирования на всех уровнях системы – это залог успешной реализации и сдачи проекта.

Выделяют следующие уровни тестирования:

- компонентное или модульное тестирование - проверяет функциональность и ищет дефекты в частях приложения;

- интеграционное тестирование – предназначено для проверки связи между компонентами, а также взаимодействия с различными частями системы;
- системное тестирование – проверка как функциональных, так и не функциональных требований в системе в целом;
- приемочное тестирование – формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью заключения удовлетворяет ли система приемочным критериям или вынесения решения заказчиком или другим уполномоченным лицом принимается приложение или нет.

Верификация программного продукта является более общим понятием, нежели тестирование. Имеет целью доказательство того, что выполнены все формальные условия, которые были сформулированы в ТЗ и описании требований. Тестирование, в конечном итоге, выступает как инструмент, который используется при верификации.

Тема №14. Управление стоимостью проекта. Методы анализа эффективности исполнения IT-проекта.

К процессам управления стоимостью проекта относятся.

1. Планирование управления стоимостью. Планируются подходы к оценке стоимости проекта, формированию его бюджета, организации контроля за исполнением бюджета и организации корректирующих мероприятий в случае существенных отклонений от бюджета.
2. Стоимостная оценка. Предполагает оценку затрат на выполнение операций проекта, которая сводится к суммированию стоимости всех используемых операциями видов ресурсов: трудовых, материальных и финансовых.
3. Разработка бюджета расходов. Стоимости отдельных операций складываются в стоимости этапов и фаз проекта, которые, в свою очередь, складываются в общую стоимость всего проекта. Бюджет расходов предполагает распределение общей стоимости проекта на календарные периоды реализации тех или иных операций, этапов и фаз.
4. Управление стоимостью. Мониторинг расхода средств между проектными работами с непрерывной оценкой эффективности потребления бюджета. Выполнение корректирующих мероприятий при значительных отклонениях фактического потребления бюджета от плана.

Разработка плана управления стоимостью базируется на следующих исходных пунктах.

- Устав проекта. В нем фиксируется размер суммарного бюджета проекта и допустимый уровень его превышения. Планирование стоимости выполняется исходя именно из этих величин.
- План управления расписанием. Определяет распределение стоимости проектных операций по календарным периодам, что, в свою очередь, определяет распределение бюджета во времени.

- План управления рисками. Определяет размеры и порядок использования стоимостных резервов, а также стоимость мероприятий по реагированию на риски.
- Факторы среды предприятия. Оказывают влияние как собственно на планирование бюджета расходов, так и на его потребление. Например, оплата накладных расходов организации за организационное сопровождение и реализацию проекта силами ее подразделений в размере 20% от стоимости проекта.
- Активы процессов организации. Сюда включается весь ранее накопленный опыт организации по планированию, выполнению и контролю бюджета проекта, а также сформированные для этого организационные процедуры и регламенты.

При разработке плана управления стоимостью используется набор инструментов, включающий в себя следующие.

- Экспертная оценка. Позволяет оценить стоимостные параметры проекта и отдельных его операций на основании исторической информации. Проведение подобной оценки возможно только специалистами, имеющими значительный успешный опыт участия в управлении стоимостью рядом предыдущих проектов.
- Анализ альтернатив. Позволяет оценить возможности привлечения финансирования проекта из источников, не включенных в устав проекта.
- Совещания участников проекта (руководитель и спонсор проекта, ключевые члены команды проекта, стейкхолдеры и другие заинтересованные стороны). Совещания проводятся для составления плана управления бюджетом проекта, концепции формирования и расходования управленческих резервов стоимости отдельных его фаз.
- Связи организационных процедур. Предусматривают использование внутриорганизационных регламентов в рамках проекта для решения задач управления стоимостью.

В плане управления стоимостью указываются.

- Единицы измерения для каждого ресурса, например, человеко-часы или человеко-дни для оценки времени.
- Степень точности, указывающая приемлемый диапазон (например, плюс/минус 10 %), который будет использоваться в рамках реалистичных оценок стоимости.
- Контрольные пороги отклонений, устанавливающие заранее согласованную величину, при отклонении от которой необходимо предпринимать какие-либо действия. Пороги обычно выражаются в виде процентных отклонений от базового плана.
- Методы освоенного объема для управления стоимостью (взвешенные контрольные события, фиксированные значения, процент выполнения и т.д.)
- Форматы отчетности, определяющие форму и частоту составления различных отчетов о стоимости.
- Процедура документирования стоимости проекта.

К методам стоимостной оценки относятся следующие:

- оценка по аналогам,
- определение ставок стоимости ресурсов,
- оценка «снизу вверх»,
- оценка «сверху вниз»,
- параметрическая оценка,
- анализ предложений исполнителей,
- анализ резервов.

При оценке стоимости по аналогам в качестве основы принимается стоимость ранее реализованного проекта или отдельной операции. Например, проект автоматизации вуза А стоил N рублей. При оценке стоимости проекта автоматизации вуза В можно сопоставить с величиной N в зависимости от соотношения количества сотрудников и студентов в этих вузах.

Данный метод может быть эффективно применен на начальных фазах проекта в условиях отсутствия точной информации о содержании и объемах предстоящих работ. Метод прост в использовании и не требует каких-либо существенных затрат. Однако, для его успешного применения требуется наличие детальной информации по ранее реализованным проектам, наличие экспертов, которые могли бы сопоставить текущий проект с предыдущими. Наиболее точные оценки получаются в тех случаях, когда сопоставляемые проекты близки не только по форме, но и по содержанию проектных работ.

При определении ставок стоимости ресурсов устанавливаются ставки стоимости за единицу трудоемкости ресурса (например, почасовая зарплата сотрудников проекта). Стоимость каждой операции проекта вычисляется как произведение ставки на трудоемкость или длительность операции. Далее стоимости отдельных операций складываются в соответствии с ИСР в стоимость всего проекта. Такая схема реализована, например, в системе Microsoft Project и позволяет достаточно точно оценить стоимость всего проекта и каждой отдельной фазы или операции еще на этапе планирования работ.

При оценке стоимости «снизу-вверх» сначала выполняется оценка стоимости отдельных пакетов работ или отдельных операций с максимальной степенью детализации. Эта стоимость затем суммируется на более высоких уровнях ИСР. Точность такой оценки обычно зависят от размеров и сложности конкретных операций или пакетов работ. Обычно чем ниже трудоемкость операций, тем выше точность их стоимостной оценки.

Параметрическая оценка стоимости используется тогда, когда можно выделить один или несколько параметров, для которых можно построить аналитическую зависимость между значениями параметров и стоимостью операции. Для конкретной операции задаются значения параметров, а ее стоимость рассчитывается по заданной формуле. При этом для различных операций могут быть использованы разные параметры и разные математические зависимости, а для некоторых операций вообще не удается подобрать подходящий параметр и его взаимосвязь со стоимостью.

Например, стоимость разработки одной страницы сайта – 10000р. Проект предполагает разработку сайта, состоящего из 10 страниц. Стоимость проекта – 10 * 10000 = 100000р.

Стоимостная оценка «сверху-вниз» применяется в условиях отсутствия детальной ИСР, нехватки информации о ресурсах и материалах, необходимых для реализации работ. Сначала дается укрупненная оценка всего пакета работ, а затем она детализируется и декомпозируется на отдельные элементы (по работам, исполнителям и др.). Метод имеет право на жизнь на ранних этапах проекта, когда выполняется оценка его жизнеспособности и непонятно, следует ли расходовать ресурсы на более детальное планирование и оценку.

Стоимостная оценка методом анализа предложений исполнителей чаще всего используется в тех случаях, когда на проект объявлен тендер, проектная организация планирует участвовать в конкурсе и оценивает предполагаемую стоимость работ только на основании тендерной документации. Команда проекта, в условиях недостатка детальной информации, проводит укрупненное разбиение проекта на основные этапы работ, закрепляет за этими этапами группы исполнителей и собирает их предложения по стоимости и срокам выполнения. Следует учитывать, что исполнители склонны завышать стоимостную оценку операций. В связи с этим этот способ может потребовать несколько итераций сбора предложений исполнителей.

Анализ резервов предполагает, что в стоимость операций включаются резервы (средства на непредвиденные обстоятельства). При этом возникает проблема возможного завышения стоимостной оценки операции.

Резерв на непредвиденные обстоятельства – это часть стоимости, используемая по усмотрению менеджера проекта в случае возникновения возможных, но не определенных событий (например, рисков). Резервы могут включаться в базовый план проекта или не включаться в него. В первом случае они носят характер резервов на покрытие неопределенности. Во втором случае они образуют управленческий резерв, который поступает в распоряжение менеджера проекта и расходуется по мере выполнения мероприятий по реагированию на риски. Наиболее распространен следующий способ управления резервами на непредвиденные обстоятельства. В конце фазы создается фиктивная работа. Длительность этой работы равна нулю, а стоимость – размеру стоимостного резерва всей фазы, который образуется суммированием стоимостных резервов входящих в нее операций. Если какая-то операция не вписывается в отведенные для нее пределы стоимости, общий резерв фазы расходуется на покрытие превышения ее стоимости. Такой подход позволяет более эффективно управлять стоимостными резервами операция, перебрасывая их на те работы, где в этом возникает необходимость.

Сравнение методов стоимостной оценки, с позиций когда и при каких условиях можно применять тот или иной метод, приведено в таблице.

Метод оценки	Основания для применения и сфера (этап)	Необходимые условия

Параметрическая оценка	Наличие оценок объемов работ и нормативной стоимости отдельных элементов работ. Применяется на любых этапах проекта. Точность зависит от точности оценок объемов работ и их нормативной стоимости.	Наличие возможности нормирования стоимости работ. Возможность расчета оценок исходя из объемных параметров работ. Наличие нормативов стоимости отдельных типовых операций.
Оценка по аналогам	Недостаток детальной информации. Применяется на ранних фазах проекта.	Схожесть работ по содержанию и типу. Наличие информации о фактической стоимости работы-аналога. Наличие опыта у участников.
Оценка «сверху вниз»	Необходимость быстрой укрупненной оценки стоимости. Применяется для фазы замысла (идеи) проекта.	Возможность укрупненной оценки стоимости всего проекта.
Оценка «снизу вверх»	Необходимость в уточненной оценке стоимости. Повторная оценка стоимости. Рекомендуется для фазы детального планирования.	Невысокие трудоемкость и объем работ отдельных операций. Наличие достаточно точных оценок необходимых ресурсов для отдельных операций. Историческая информация о стоимости отдельных типовых операций. Наличие нормативов затрат. Тщательно проработанная ИСР.
Анализ предложений исполнителей	Закупка оборудования у поставщиков. Организация тендера. Наличие возможности выполнить работы силами внешних организаций.	Качественная тендерная (конкурсная) документация. Детализированные предложения конкурсантов (оферты). Доступность экспертной оценки.
Анализ резервов	Управление стоимостью операций	Наличие операций с риском превышения плановой стоимости.

Разработка бюджета расходов – это объединение оценок стоимости отдельных операций или пакетов работ для создания общего базового плана по стоимости.

При разработке бюджета расходов используются методы.

1. Суммирование стоимости.
2. Анализ резервов.
3. Параметрическая оценка.
4. Согласование объемов финансирования.

При построении ИСР процесс декомпозиции происходит «сверху-вниз», разбивая элементы верхних уровней на более мелкие элементы. Конечным этапом декомпозиции являются пакеты работ. При суммировании стоимости процесс идет в обратном направлении «снизу-вверх». Сначала определяется стоимость пакетов работ, которая суммируется на более высоких уровнях ИСР, образуя стоимость этапов, подпроектов, промежуточных результатов и т.п. В конечном итоге все компоненты стоимости складываются в общую стоимость всего проекта. Если при этом учесть расписание операций проекта с привязкой к календарю, получим распределенный по времени бюджет расходов.

Резервы на непредвиденные обстоятельства – это бюджет, зарезервированный на случай незапланированных, но возможных изменений содержания и стоимости проекта. Для расходования этого резерва менеджер проекта должен всегда получать одобрение. Эти резервы не входят в базовый план по стоимости, но включаются в бюджет проекта. Они не распределяются по проекту, как бюджет, и поэтому не учитываются при расчете по методике освоенного объема.

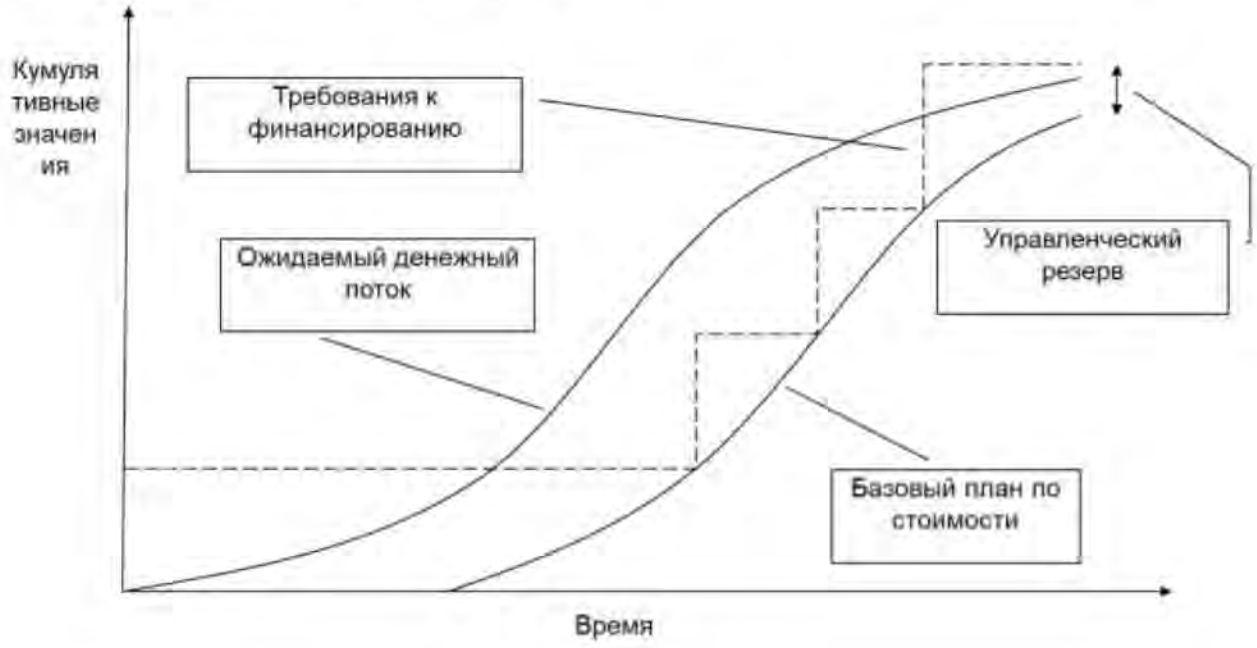
Параметрическая оценка используется тогда, когда можно построить некую математическую модель зависимости стоимости от параметров проекта и исходной задачи. Значения параметров являются исходными данными для такой модели. Сама же модель позволяет как оценить стоимость проекта при заданном наборе параметров, так и подобрать такие значения параметров, которые «впишутся» в установленный бюджет проекта. К сожалению, подобрать адекватную математическую модель стоимости удается далеко не для всех проектов.

Согласование объемов финансирования предполагает, что большие колебания объемов расходов по периодам крайне нежелательны. Возникает необходимость в согласовании объемов расходуемых средств по проекту с объемами финансирования, установленными заказчиком. Для этого требуется, чтобы расписание работ и порядок выплат были составлены так, чтобы они носили плавный характер, без резких колебаний, т.е. чтобы выплаты производились по требуемым датам, которым в расписании проекта соответствуют завершение определенных пакетов работ, элементов ИСР или контрольные события расписания.

Результатами разработки бюджета расходов являются базовый план по стоимости и требования к финансированию проекта.

Базовый план по стоимости. Представляет собой распределенный по календарю бюджет, по которому выполняется сверка, мониторинг и контроль использования денежных средств проекта в целом. Он разрабатывается суммированием оценок стоимости по временным периодам и отображается в виде S-кривой, изображенной на рисунке.

Требования к финансированию проекта формируются на основании базового плана стоимости. Финансирование представляет собой инкрементные суммы, нарастание которых происходит не постоянно, поэтому они изображаются в виде ступенчатой функции. Общее количество требуемых средств – это сумма средств, заданных в базовом плане по стоимости, и резерва на непредвиденные обстоятельства.



Метод освоенного объёма основан на четырех основных величинах.

- Базовая стоимость запланированных работ (БСЗР, PV). Обозначает общую стоимость работ, которые должны быть завершены к текущему моменту (каковы должны быть затраты на проект по базовому плану).
- Фактическая стоимость выполненных работ (ФСВР, AC). Обозначает общую фактическую стоимость трудозатрат на текущий момент (сколько фактически потрачено на проект к текущему моменту).
- Базовая стоимость выполненных работ (БСВР, EV). Обозначает запланированную по базовому плану стоимость фактически выполненных работ (сколько планировалось потратить на трудозатраты, которые были фактически осуществлены).
- Бюджет по завершении (БПЗ, BAC). Плановые затраты на проект (или задачу) согласно базовому плану.

На основе приведенных основных величин вычисляются производные показатели методики освоенного объема, которые позволяют как оценить состояние хода выполнения проекта на текущий момент времени, так и произвести прогноз итоговых затрат или сроков завершения проекта в будущем, в предположении что работа над проектом будет выполняться также, как она выполнялась до сих пор.

Производные показатели и их интерпретация приведены в таблице.

Название	Формула	Значение	Трактовки

Отклонение от календарного плана (ОКП, SV)	ОКП = БСВР – БСЗР	<0 =0 >0	Отставание от плана Выполнение в срок Опережение плана
Отклонение по стоимости (ОПС, CV)	ОПС = БСВР – ФСВР	<0 =0 >0	Превышение затрат Затраты по плану Экономия средств
Относительное отклонение по стоимости (ООПС)	ООПС = ОПС / БСВР * 100	<0 =0 >0	Превышение затрат Затраты по плану Экономия средств
Индекс отклонения стоимости (ИОС, CPI)	ИОС = БСВР / ФСВР	<1 =1 >1	Превышение затрат Затраты по плану Экономия средств
Относительное отклонение от календарного плана (ООКП)	ООКП = ОКП / БСЗР * 100	<0 =0 >0	Отставание от плана Выполнение в срок Опережение плана
Индекс отклонения от календарного плана (ИОКП, SPI)	ИОКП = БСВР/ БСЗР	<1 =1 >1	Отставание от плана Выполнение в срок Опережение плана
Предварительная оценка по завершении (ПОПЗ, EAC)	ПОПЗ = ФСВР + (БПЗ – БСВР) / ИОС	<БПЗ =БПЗ >БПЗ	Экономия средств Затраты по плану Превышение затрат
Отклонение по завершении (ОПЗ, VAC)	ОПЗ = БПЗ – ПОПЗ	<0 =0 >0	Превышение затрат Затраты по плану Экономия средств
Показатель эффективности выполнения (ПЭВ)	ПЭВ = (БПЗ – БСВР)/(БПЗ – ФСВР)	<1 =1 >1	Средства экономятся Ход работ по плану Превышение затрат
Прогноз до завершения (ПДЗ, ETC)	ПДЗ = ПОПЗ - ФСВР		Ожидаемая стоимость выполнения оставшейся части проекта

Рассмотрим пример анализа хода выполнения проекта по методу освоенного объема. Пусть на текущий момент времени по базовому плану запланировано выполнить работ общей стоимостью в 1000 условных денежных единиц. Стоимость всех фактически выполненных работ на текущий момент оценивается в 800 условных денежных единиц. А по базовому плану на выполнение фактически выполненных работ предполагалось потратить 1100

единиц. При этом на весь проект по базовому плану предполагается потратить 10000 единиц.

Расчет показателей метода освоенного объема и их интерпретация приведены в таблице.

Показатель	Интерпретация
БСЗР = 1000	На момент оценки по проекту запланировано работ на эту сумму
ФСВР = 900	На момент оценки по проекту выполнено работ на эту сумму
БСВР = 1100	Те работы, которые выполнены к моменту оценки, по базовому плану стоят столько
БПЗ = 10000	Столько планируется потратить на все работы проекта
ОКП = БСВР – БСЗР= 1100 – 1000 = 100	Отклонение от календарного плана > 0 – опережение плана
ОПС = БСВР – ФСВР= 1100 – 900 = 200	Отклонение по стоимости > 0 – экономия средств, сэкономлено 200р
ООПС = ОПС / БСВР * 100 = 200/1100*100 = 18,2	Экономия средств в процентах Сэкономлено 18,2% средств от запланированного объема
ИОС = БСВР / ФСВР = 1100 / 900 = 1,2	Индекс отклонения по стоимости. Экономия затрат на 20%
ООКП = ОКП / БСЗР * 100 = 100/1000*100=10	Относительное отклонение от календарного плана: опережение графика работ на 10%
ИОКП = БСВР/ БСЗР = 1100/1000 = 1,1	Индекс отклонения от календарного плана: опережение графика работ на 10%
ПОПЗ = ФСВР + (БПЗ – БСВР) / ИОС = 900 + (10000 - 1100)/1,2 = 8316	Столько будет фактически стоить весь проект, если мы будем работать также как работали до момента оценки
ОПЗ = БПЗ – ПОПЗ = 10000 – 8316 = 1684	Отклонение по завершении: К концу проекта мы сэкономим столько средств, если будем работать также как работали до момента оценки
ПЭВ = (БПЗ – БСВР)/(БПЗ – ФСВР) = (10000 - 1100) / (10000 – 900) = 0,98	Показатель эффективности выполнения: Проект обойдется нам в 98% запланированной ранее суммы, если будем работать также как работали до момента оценки

ПДЗ = ПОПЗ – ФСВР = 8316 – 900 = 7416	Прогноз до завершения: На оставшуюся часть проекта потребуется потратить эту сумму, если будем работать также как работали до момента оценки

Тема №15. Управление командой проекта.

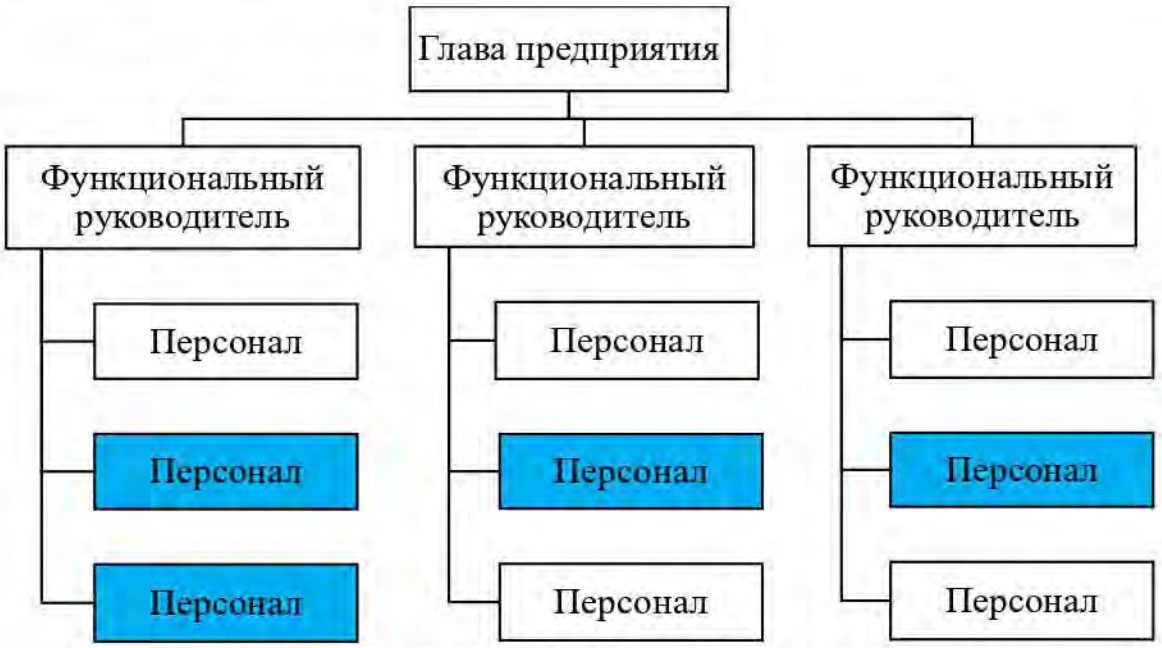
Для выполнения проекта создается команда проекта - новый временный рабочий коллектив, состоящий из специалистов различных структурных подразделений компаний со стороны Исполнителя и со стороны Заказчика. Для членов команды проекта необходимо определить проектные роли (временные должности), функции, обязанности, ответственность, полномочия и правила взаимодействия, а также организационную схему, отражающую отношения подчиненности.

Организационная структура проекта – это соответствующая проекту временная организационная структура, включающая всех его участников и создаваемая для успешного управления и достижения целей проекта. Поскольку проект – это временное предприятие, то по его окончании команда проекта распускается и специалисты приступают к своим функциональным обязанностям в соответствии со штатной организационной структурой компании или переходят на следующий проект, где их функции и полномочия могут быть другими.

Проекты всегда реализуются в рамках некоторой организации или предприятия, которые имеют свою организационную структуру. Команда проекта должна быть вписана в эту структуру, а ее функционирование во многих случаях определяется особенностями этой организационной структуры.

Рассмотрим основные виды организационных структур и их особенности с точки зрения работы проектной команды: функциональная, проектная, сбалансированные организации, слабая и сильная матричные организации.

Функциональная организация – иерархически выстроенная организация, в которой у каждого сотрудника есть один прямой начальник, сотрудники разделены на группы (отделы) по областям специализации. Каждая группа (отдел) управляется одним человеком - функциональным руководителем (руководителем отдела). Пример функциональной организации изображен на рисунке.

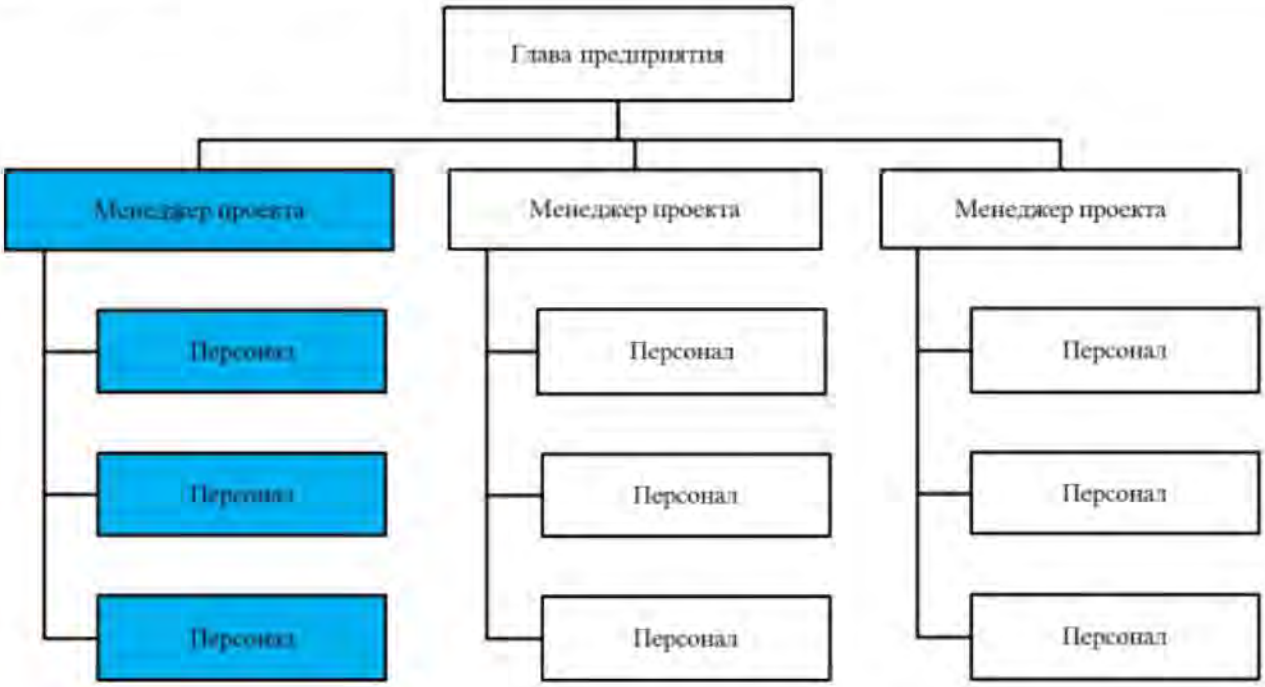


Каждый работник имеет одного четко выделяемого руководителя. Персонал группируется по специальностям, например, кодирование, маркетинг, финансы, разработка документации, тестирование. Команда проекта оказывается разбросанной по различным подразделениям. На рисунке члены команды отмечены заливкой.

Когда возникает вопрос, касающийся специалистов другого отдела, работники подают запрос своему руководителю, который консультируется с руководителем другого отдела. Руководитель этого отдела передает запрос своим подчиненным. Ответ подчиненных транслируется по обратной цепочке.

Недостатком такой организации является отсутствие выраженного руководителя проекта. Дополнительная нагрузка ложится на функциональных руководителей, которым помимо своих основных обязанностей приходится выполнять роль координации проекта. В силу этого в рамках функциональной организации невозможно реализовать большое количество проектов, а проектная деятельность является всего лишь побочной, второстепенной деятельностью функциональной организации.

Противоположностью функциональной является проектная организация, структура которой изображена на рисунке.

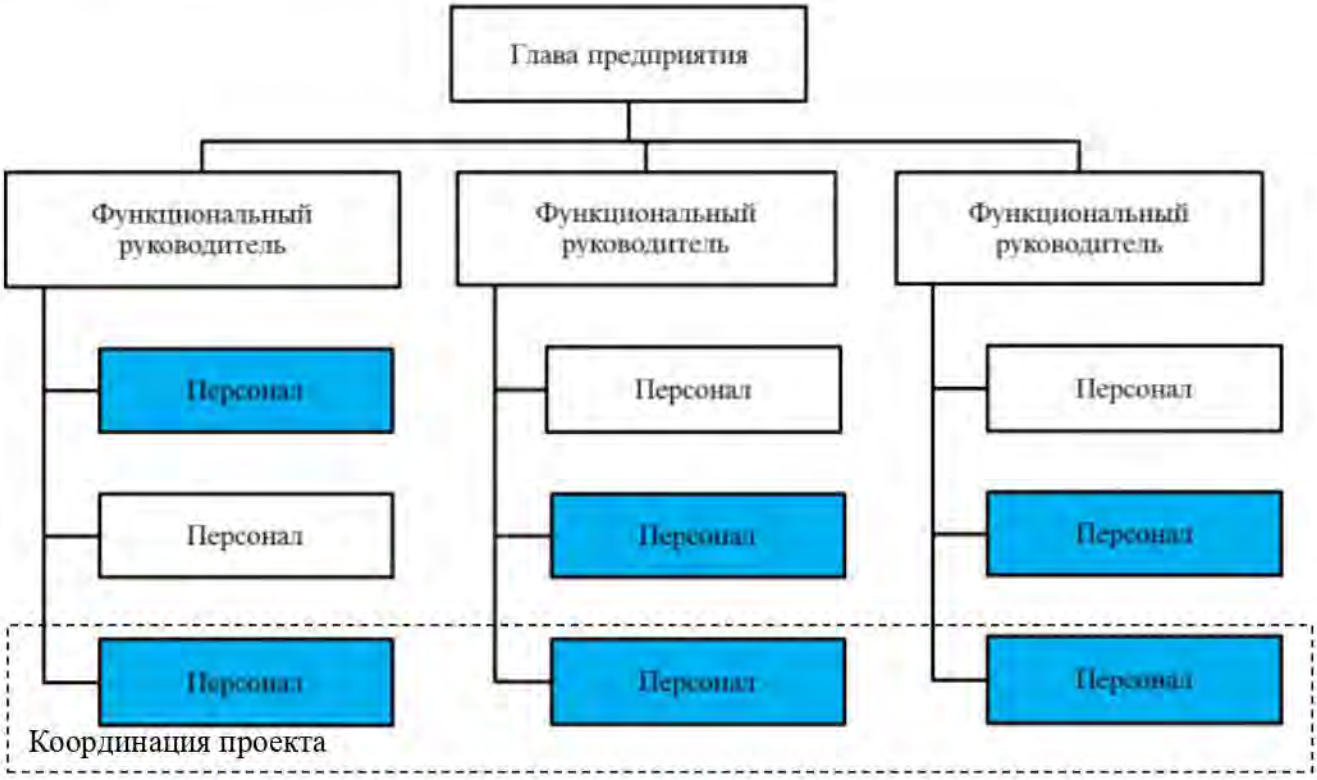


В такой организации проектная деятельность является основным видом деятельности. Руководителю предприятия непосредственно подчинены менеджеры проектов, каждый из которых обладает полным спектром полномочий в рамках своего проекта по установлению приоритетов, использованию ресурсов и руководству работой лиц, назначенных на исполнение проекта, а также финансовыми полномочиями в рамках бюджета проекта.

Команда проекта собрана вместе в одно подразделение, которое подчинено менеджеру проекта. Большая часть ресурсов организации задействована в работах проектов. Менеджеры проектов в значительной степени независимы и обладают большими полномочиями. Проектные организации часто имеют подразделения, называемые отделами, но эти подразделения подотчетны непосредственно менеджеру проекта.

К недостаткам проектной организации можно отнести тот факт, что работники работают по временным трудовым договорам и по окончании работ по проекту могут покинуть организацию вместе с накопленным опытом.

В слабой матричной организации нет явно выраженного руководителя проекта. Но для разгрузки функциональных руководителей в каждом подразделении из членов команды проекта назначаются координаторы, на плечи которых ложится взаимодействие с работниками других подразделений. Структура слабой матричной организации приведена на рисунке.

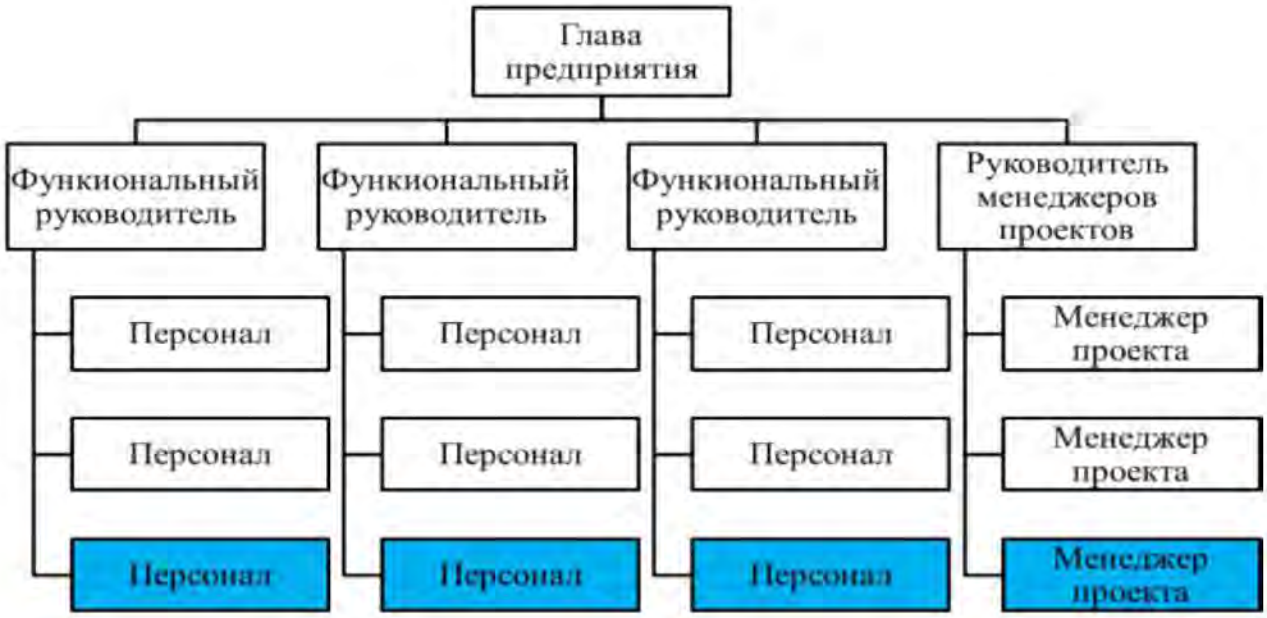


Такая организационная структура сохраняет многие характеристики функциональной организации. Функции менеджера проекта в ней скорее соответствуют функциям координатора или диспетчера проектов, а не менеджера. Окончательные решения принимаются функциональными руководителями по представлению соответствующих координаторов.

В сильной матричной организации руководитель проекта разделяет с функциональными руководителями (руководителями отделов) ответственность по заданию приоритетов и управлению работой лиц, назначенных на исполнение проекта. Такая организация обладает многими характеристиками проектных организаций. В ней могут быть штатные менеджеры проектов с широкими полномочиями и входящий в штат управленческий персонал проектов.

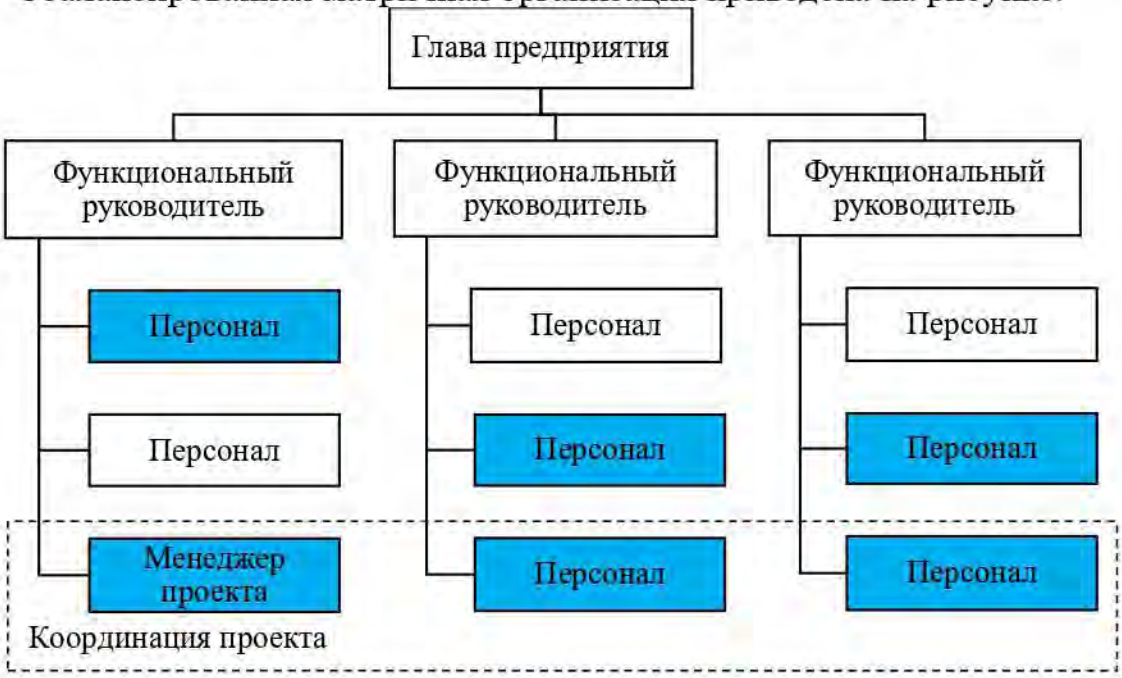
В проекте участвует персонал, подчиненный различным функциональным руководителям. Но функциональный руководитель отвечает не за вопросы реализации проекта, а за вопросы организации работ по видам производственной деятельности. Например, постановка задачи и разработка ТЗ, разработка ПО, тестирование ПО и т.п.

Структура сильной матричной организации изображена на рисунке.



В сбалансированной матричной организации руководитель проекта наравне разделяет с функциональными руководителями (руководителями отделов) ответственность по заданию приоритетов и управлению работой лиц, назначенных на исполнение проекта. В качестве руководителя проекта назначается один из координаторов проекта от структурных подразделений.

Сбалансированная матричная организация приведена на рисунке.



В приведенной ниже таблице приведено сравнение характеристик проектов, реализуемых в организациях с различными видами организационных структур.

Характеристики	Виды организационных структур

проекта	Функцио-нальная	Матричные			Проектная
		Слабая	Сбалансир	Сильная	
Полномочия менеджера проекта	Незначит. или нет	Ограничено	Средний уровень	Высокий уровень	Полный контроль
Наличие ресурсов	Незначительно или нет	Ограничено	Средний уровень	Высокий уровень	Полный контроль
Контроль бюджета проекта	Функцион. рук.	Функцион. рук.	Смешан.	Менеджер проекта	Менеджер проекта
Роль менеджера проекта	Частичная занятость в проекте	Частичная занятость в проекте	Полная занятость в проекте	Полная занятость в проекте	Полная занятость в проекте
Администр. персонал проекта	Частичная занятость в проекте	Частичная занятость в проекте	Частичная занятость в проекте	Полная занятость в проекте	Полная занятость в проекте

Процессы управления человеческими ресурсами состоят из четырех процессов.

1. Планирование человеческих ресурсов – определение и документальное оформление ролей, ответственности и подотчетности, а также создание плана обеспечением проекта персоналом.
2. Набор команды проекта – привлечение человеческих ресурсов, необходимых для выполнения проекта.
3. Развитие команды проекта – повышение квалификации членов команды проекта и укрепление взаимодействия между ними с целью повышения эффективности исполнения проекта.
4. Управление командой проекта – контроль за эффективностью членов команды проекта, обеспечение обратной связи, решение проблем и координация изменений, направленных на повышение эффективности исполнения проекта.

Форматами документирования ролей сотрудников в проекте могут выступать иерархическая организационная диаграмма, разнообразные виды матричных диаграмм (в том числе – матрица ответственности) и произвольный текстовый формат.

Одним из наиболее распространенных форматов является матрица ответственности, которая используется для отображения связей между пакетами работ или операциями и членами команды проекта. Пример матрицы ответственности приведен в таблице. В ней использованы обозначения: Исп – исполнитель, Согл – согласование, Утв – утверждение, Н – наблюдение.

Проектные работы	Проектные роли

	Руководитель проекта	Бизнес-менеджер	Менеджер по конфигурации	Администратор	Бизнес-аналитик	Системный архитектор	Разработчик	Тестировщик
Управление программой	Исп							
Планирование проекта	Исп	Согл						
Управление рисками и дефектами	Исп		Исп	Н				Исп
Управление коммуникациями	Исп	Согл		Н				
Управление предложениями	Исп	Согл						
Управление качеством	Исп							Исп
Обучение	Утв			Н	Исп	Исп		
Анализ бизнес-процессов	Утв				Исп			
Требования к конфигурации	Утв		Согл		Исп	Исп		
Сбор бизнес-требований					Исп			
Анализ соответствия решения задаче			Согл		Исп	Исп		
Написание дополнительного кода		Утв					Исп	
Тестирование и обеспечение качества		Утв						Исп
Инфраструктура		Утв				Исп		
Интеграция и интерфейсы	Утв					Исп	Исп	
Миграция данных	Утв					Исп	Исп	

Другими видами матричного формата являются матрица подотчетности, потребность работ в ресурсах, реестр навыков и другие.

Матрица подотчетности задает отношение подотчетности между членами команды. Пример фрагмента такой матрицы приведен ниже.

Матрица потребности работ в ресурсах формируется на основании расписания проекта. В ней перечисляются проектные работы и для каждой из них указывается трудоемкость и роли, которые исполняют данную работу. Пример приведен в таблице.

Готовят отчет	Получают отчет

	Руководитель проекта	Бизнес-менеджер	Менеджер по конфигурации	Разработчик	Тестировщик	Менеджер по качеству
Руководитель проекта	–	По необх.	По необх.	Никогда	Никогда	По необх.
Бизнес-менеджер	По необх.	–	Никогда	Никогда	Никогда	Никогда
Менеджер по конфигурации	Каждую неделю	Каждую неделю	–	По необх.	По необх.	По необх.
Разработчик	По необх.	По необх.	По необх.	–	По необх.	По необх.
Тестировщик	По необх.	По необх.	По необх.	По необх.	–	По необх.
Менеджер по качеству	Каждую неделю	Каждую неделю	По необх.	По необх.	По необх.	–

Задача	Трудоемкость, человеко-дни	Роли исполнителя
Управление программой	26	Руководитель проекта
Планирование проекта	4	Руководитель проекта Менеджер по качеству Менеджер по конфигурации
Управление рисками и дефектами	8	Руководитель проекта Менеджер по качеству Менеджер по конфигурации
Управление коммуникациями	7	Руководитель проекта Менеджер по конфигурации
Управление предложениями	4	Руководитель проекта
Управление качеством	3	Руководитель проекта Менеджер по качеству
Обучение	3	Менеджер по конфигурации Системный архитектор Администратор проекта
Анализ бизнес-процессов	1	Менеджер по конфигурации
Требования к конфигурации	7	Менеджер по конфигурации Системный архитектор
Сбор бизнес-требований	2	Менеджер по конфигурации

Анализ соответствия решения задаче	3	Менеджер по конфигурации
Написание дополнительного кода	1	Разработчик
Тестирование и обеспечение качества	2	Менеджер по качеству Тестировщик
Инфраструктура	3	Системный архитектор
Интеграция и интерфейсы	2	Системный архитектор Разработчик Тестировщик
Миграция данных	12	Системный архитектор Разработчик Тестировщик

Реестр навыков формируется для каждого класса персонала. В нем перечисляются ключевые навыки, которыми должен владеть персонал, их важность (рейтинг критичности) и минимально необходимый уровень владения навыком (рейтинг способности).

Результаты планирования человеческих ресурсов должны отражать следующие моменты.

1. Распределение ролей и ответственности.

A. Роль. Обозначение должности или участия члена команды проекта в его реализации. Каждая роль должна характеризоваться набором возлагаемых полномочий и степенью принимаемой ответственности.

B. Полномочия. Предполагают возможность распоряжения и управления определенными ресурсами проекта, принимать и/или утверждать проектные решения, выполнять другие действия, направленные на реализацию проекта. Уровень полномочий должен соответствовать степени ответственности.

ответственности.

C. Ответственность. Перечень тех работ, которые закреплены за конкретной ролью в рамках реализации проекта.

D. Квалификация. Набор знаний, умений и навыков, которые требуются от роли для успешной реализации задач проекта.

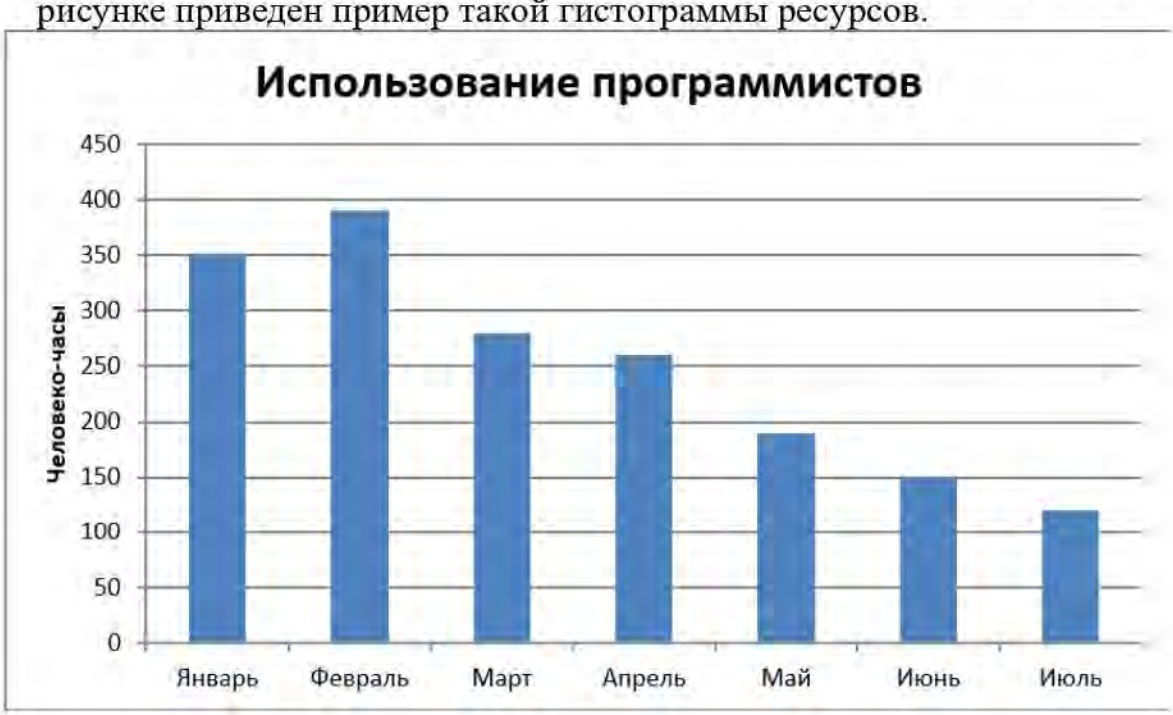
2. Организационная диаграмма проекта. Это иерархическая диаграмма, определяющая подотчетность каждого члена команды.

3. План управления обеспечением проекта персоналом. Описывает подходы к распределению ролей проекта между сотрудниками организации и мероприятия по набору и развитию команды.

В плане обеспечения проекта персоналом должны быть отражены следующие сведения:

- Набор персонала. Как будет набираться команда проекта. Какие роли будут выполнять штатные сотрудники, на какие роли будут приглашаться совместители или набираться дополнительный штат сотрудников. Какова заработная плата сотрудников различного уровня квалификации. Как будет организована работа команды: будет ли использоваться выделенный офис или создается работающая удаленно виртуальная команда.

- Расписание. Задает временные интервалы потребности проекта в том или ином трудовом ресурсе. Определяет, когда следует начать и завершить операции привлечения сторонних участников к работам проекта. Одним из инструментов графического отображения расписания является гистограмма ресурсов. По горизонтальной оси откладываются календарные периоды, а по вертикальной – количество трудозатрат ресурса, необходимое для реализации работ проекта. На диаграмме может быть приведена линия, обозначающая предельную загрузку ресурса. В случае превышения этой линии в некоторые периоды требуется проводить выравнивание ресурсов с возможным изменением расписания проекта.На рисунке приведен пример такой гистограммы ресурсов.



- Критерии освобождения ресурсов. График освобождения сотрудников от участия в проекте. По мере завершения работ над проектом участие определенных ролей и соответствующих сотрудников прекращается. Это приводит к прекращению выплаты заработной платы. Работники должны знать график освобождения ресурсов, чтобы иметь возможность заранее планировать свою дальнейшую карьеру. Считается, что плановый переход освобождающихся сотрудников в другие проекты благоприятно сказывается на общем микроклимате в организации, повышает доверие и отдачу работников организации.
- Обучение персонала. Требуется когда квалификации участвующих в проекте сотрудников недостаточно для его успешного выполнения, или заказчику требуется документальное подтверждение квалификации в форме соответствующих сертификатов. Составляется план обучения персонала с указанием программ дополнительного образования, сертификационных курсов, их сроков и стоимости.

- Поощрение и премирование. Четко прописанные и понятные правила получения дополнительного вознаграждения за повышенную эффективность и качество работы, задачей которых является стимулирование членов команды проекта на повышение производительности труда. Наиболее действенными являются правила премирования, которые основаны на показателях, входящих в сферу ответственности конкретного лица.
- Соответствие. Режим работы персонала должен соответствовать законодательству в сфере труда и социальной политики, соглашениям с профсоюзами, условиям контракта и другим установленным правилам.
- Безопасность. Система мероприятий по защите здоровья сотрудников на рабочих местах. Одним из элементов такой системы является своевременный инструктаж по технике безопасности и другие профилактические мероприятия.

Набор команды проекта – это процесс привлечения извне сотрудников, квалификация которых востребована для выполнения проекта.

При наборе команды учитываются следующие факторы:

- Доступность. Какие человеческие ресурсы доступны сейчас и какие будут доступны в будущем?
- Способность. Какая у этих людей квалификация?
- Опыт работы. Имеют ли эти люди опыт такой или подобной работы? Каковы их прошлые успехи?
- Заинтересованность. Интересно ли людям работать над данным проектом?
- Стоимость. Сколько надо будет платить каждому члену команды, особенно если они нанимаются со стороны по контракту?

При наборе команды используются следующие методы.

1. Предварительное назначение. В некоторых случаях члены команды заданы заранее, т.е. они предварительно уже назначены на определенные должности. Такая ситуация может возникнуть, если определенным квалифицированным специалистам было заранее обещано участие в проекте, когда успех реализации проекта зависит от их знаний или если их назначение на определенные должности предусмотрено Уставом проекта.
2. Переговоры. Команде управления проектом могут понадобиться переговоры:
   - с функциональными руководителями – чтобы гарантировать, что проект будет обеспечен соответствующим штатом квалифицированных сотрудников на заданный период времени и чтобы члены команды проекта могли работать в проекте до полного окончания возложенных на них работ;
   - с другими командами управления проектом в рамках исполняющей организации – чтобы обеспечить проект дефицитными ресурсами или узкоспециализированными сотрудниками.
3. Набор персонала. Если в организации для реализации проекта не хватает штатных специалистов, то требуемые услуги можно получить из сторонних

источников. Это может выражаться в найме консультантов или передаче работ сторонним организациям на условиях субподряда.

4. Виртуальные команды. Виртуальные команды – это группы людей, объединенных общей целью, причем каждый член группы выполняет работу при минимальном личном контакте с другими членами или при полном его отсутствии. Работа таких команд стала возможной благодаря распространению электронных средств коммуникации. Формат виртуальных команд предоставляет возможность:
   - формировать команды из состава сотрудников одной компании, проживающих в различных регионах;
   - добавлять в команду специалистов, даже если они находятся в другом регионе;
   - привлекать в проект сотрудников, работающих дома;
   - формировать команды из исполнителей, работающих в разные смены или в разные часы;
   - привлекать к участию в проекте инвалидов;
   - браться за выполнение проектов, реализация которых в иных условиях была бы невозможна из-за высоких командировочных расходов.

Результатами набора команды проекта являются.

1. Назначение персонала в проекте. Проект считается укомплектованным штатом, когда для работы над ним назначены соответствующие люди.
2. Доступность ресурсов. Для указания доступности ресурсов документально фиксируется временной интервал, в течение которого каждый сотрудник проекта может принимать участие в выполнении работ по проекту. Чтобы создать достоверное окончательное расписание необходимо обладать информацией о всех особенностях расписания по каждому конкретному человеку, включая отпуска и обязательства по участию в других проектах.

Развитие команды проекта означает повышение квалификации сотрудников проекта и укрепление взаимодействия между ними для повышения эффективности исполнения проекта.

Целями развития команды проекта являются:

- Повышение навыков членов команды для повышения их способности выполнять операции проекта.
- Укрепление чувства доверия и сплоченности среди членов команды для повышения продуктивности работы команды.

Задачами развития команды являются:

- Повышение уровня знаний и навыков членов команды.
- Повышение чувства доверия и сплоченности среди членов команды.
- Создание динамичной и сплоченной командной культуры.

Тема №16. Часть 1. Гибкие методологии.

Классический подход к управлению проектами разработки программного обеспечения может быть представлен последовательностью следующих этапов.

Этап 1. Инициация. Определяются общие требования и параметры проекта, которые согласовываются с заказчиком. Команда проекта вырабатывает общее видение конечного продукта проекта, последовательности его разработки, требования к программному и аппаратному обеспечению и квалификации команды разработчиков.

Этап 2. Планирование. Составляется план работ по созданию требуемого продукта. Определяется состав операций, их исполнители и потребности в ресурсах, идентифицируются возможные риски, выявляются заинтересованные стороны. Составляется детальный календарный план работ с учетом резервов на непредвиденные обстоятельства. На основе календарного плана просчитывается бюджет проекта.

Этап 3. Проектирование. Прорабатывается архитектура разрабатываемого решения, выбирается наиболее подходящий комплекс инструментальных средств. При необходимости решаются задачи разработки новых или применения существующих математических моделей и методов. Разрабатываются алгоритмы функционирования отдельных модулей и их взаимодействия между собой.

Этап 4. Реализация и тестирование. На этом этапе создается готовый продукт проекта в том виде, в котором он был спроектирован и спланирован на предыдущих этапах проекта. Создается программный код или техническое устройство. Созданный продукт подвергается всестороннему тестированию с целью поиска возможных недоработок, которые оперативно устраняются. Продукт проходит процедуру верификации установленных заказчиком требований.

Этап 5. Мониторинг и завершение проекта. Мониторинг проекта выполняется в течение всех фаз его жизненного цикла от начала до окончания. Его задача – отслеживание возможных отклонений и внесение своевременных изменений в проектные планы и решения. Завершение проекта обычно связано с проведением приемосдаточных испытаний по заранее подготовленным заказчиком наборам тестовых заданий. После успешного проведения испытаний продукт передается заказчику в эксплуатацию.

Классический подход обладает рядом достоинств, которые обуславливают его распространенность и популярность вплоть до настоящего времени.

- Требует от Заказчика и руководства компании определить, что же они хотят получить, уже на первом этапе проекта.
- Раннее включение привносит стабильность в работу проекта.
- Планирование позволяет упорядочить реализацию проекта.
- Этот подход подразумевает мониторинг показателей и тестирование, что совершенно необходимо для реальных проектов различного масштаба.
- Позволяет избежать стрессов ввиду наличия запасного времени на каждом этапе, заложенного на случай каких-либо осложнений и реализации рисков.

- С правильно проведённым этапом планирования руководитель проектов всегда знает, какими ресурсами он обладает. Даже если эта оценка не всегда точная.

Тем не менее классический подход имеет и ряд существенных недостатков, являющихся причиной появления новых схем и подходов к управлению проектами, особенно в сфере разработки ПО. Важнейшим недостатком является неприспособленность к изменениям. Если в Вашем проекте ресурсы и время не являются ключевыми ограничениями, а содержание проекта подвержено изменениям – возможно вам стоит присмотреться к гибким технологиям управления проектами.

По данным исследования, проведенного на ресурсе ИНФОСТАРТ (https://infostart.ru/1c/articles/985232/), выделены следующие проблемы проектов автоматизации предприятий и их частота:

- изменение требований в процессе реализации проекта– 91%;
- нечеткие требования заказчика – 88%;
- разрастание объема работ в ходе проекта – 69%;
- несоблюдение сроков – 59%;
- сложности при согласовании расписания и бюджета – 32%;
- сложности взаимопонимания внутри проектной команды – 24%;
- другие – 6%.

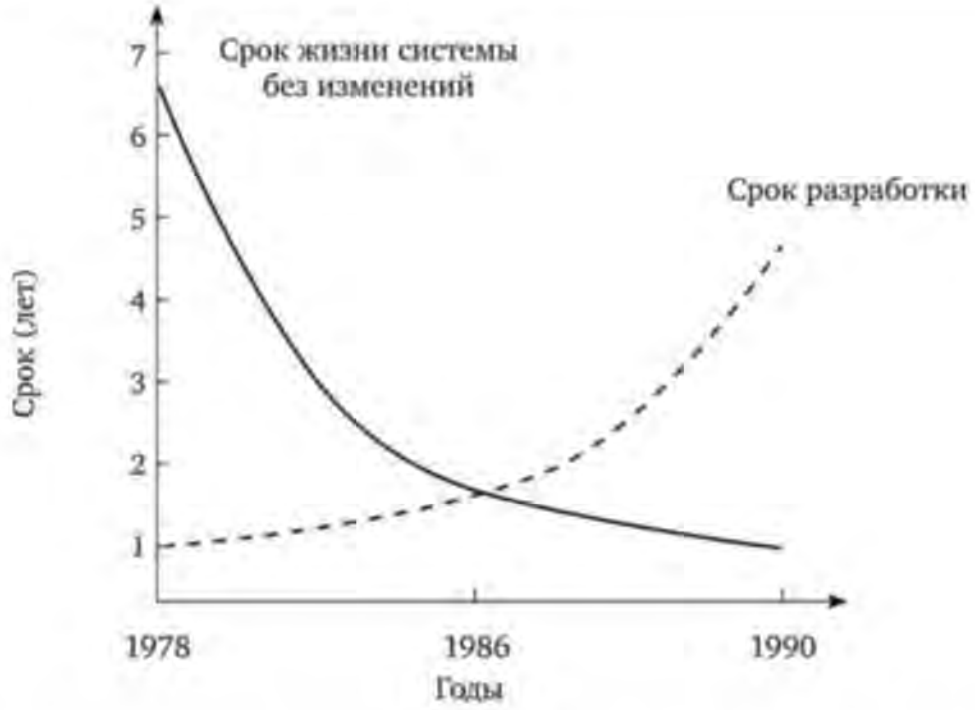
Из перечисленного перечня две наиболее часто встречающиеся проблемы напрямую противоречат идеям классического управления проектами. Именно для таких проектов и применяются гибкие методологии, основанные на принципах Agile.

Гибкие методологии начали зарождаться в конце 20-го столетия, чему способствовало появление ряда причин. В качестве основных причин можно назвать противоречие между ростом времени на первичное создание автоматизированных систем и необходимостью их быстрого эволюционного развития и непрерывное усложнение автоматизированных систем.

На рисунке представлены графики двух кривых: срок жизни системы без изменений и срок разработки системы.

В 1970-е гг средний срок разработки системы составлял один год, а срок эксплуатации такой системы без необходимости внесения изменений – 7 лет. Примерно в 1986г эти сроки сравнялись, т. е. сразу же после внедрения новой системы возникает потребность ее обновления. Начиная с 1990-х годов срок длительность разработки промышленных прикладных систем стала превышать срок их жизни без изменений. Другими словами, изменение требований к системе происходит уже до окончания ее разработки.

Современную большую прикладную систему очень сложно разработать и внедрить, пользуясь традиционным водопадным подходом — требования к системам в большинстве случаев начинают изменяться задолго до завершения проекта.

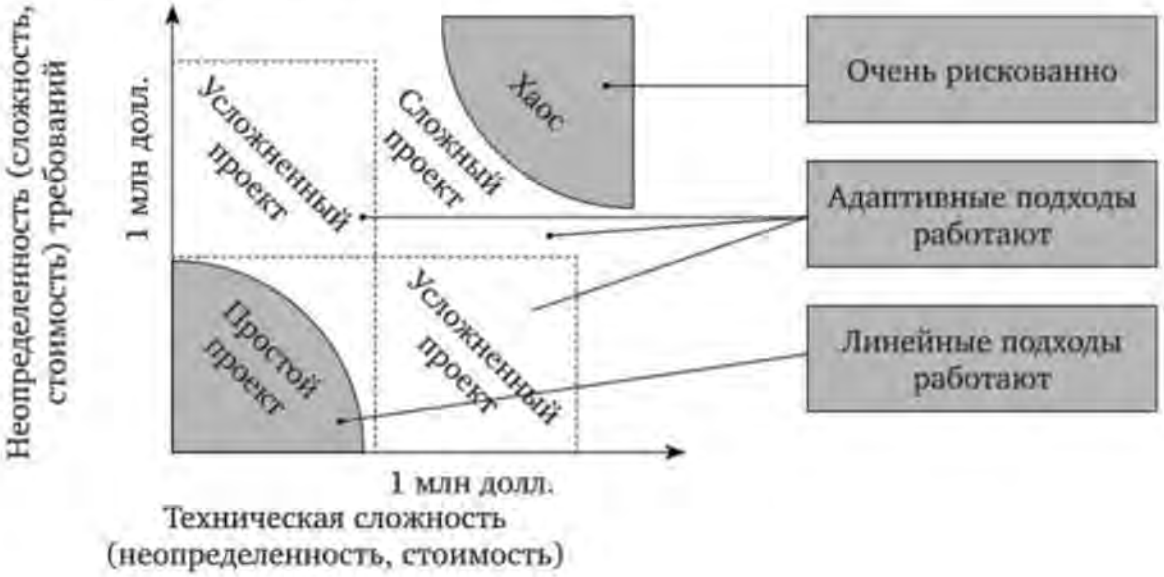


Производительность компьютеров растет, а вместе с ней увеличивается количество обрабатываемых объектов. Темпы роста составляют около 1000 раз за 15 лет. Автоматизированная система из обособленной программной системы превращается в сложный вычислительный комплекс с применением распределенной обработки и облачных вычислений. Это значительно усложняет процесс взаимодействия между компонентами системы, но позволяет автоматизировать гораздо более широкий круг задач и процессов предприятия.

Параметры и перечень функциональных свойств разрабатываемой системы практически невозможно полностью описать в момент начала ее разработки. Многочисленные уточнения и дополнения появляются по ходу разработки на любых ее этапах. Не исключены изменения требований даже в фазах тестирования и опытной эксплуатации. Кроме того, конкурентная среда не стоит на месте, она постоянно изменяется. Повышается вероятность появления до завершения работ по проекту конкурирующих разработок, способных оказать существенное влияние на изменение требований к еще разрабатываемому продукту.

На рисунке приведена схема распределения подходов к управлению проектами в зависимости от масштаба и стоимости проекта.

По данным проводимых исследований проекты стоимостью до 0,4 млн долл., реализуемые с подходом водопад, успешны в 44 % случаев. Проекты стоимостью от 1,3 млн долл., реализуемые с подходом водопад, успешны всего в 7 % случаев. В этих условиях наиболее эффективны адаптивные, гибкие методологии.



Рассмотрим подробно четыре ценности, провозглашенные манифестом Agile.

1. Люди и их взаимодействие важнее процессов и инструментов.

Чтобы люди работали эффективнее, процессы и инструменты не должны их ограничивать. В Agile ни процесс, ни тем более программный инструмент не диктует, что людям делать. Более того, они сами решают, как менять процессы и инструменты в своей работе. Чтобы ускорить процесс разработки, люди также должны взаимодействовать напрямую (без посредников в виде документов или других людей), активно общаться между собой лично, а не письменно. Правда, в современном бизнесе общение часто вынуждено переходить в онлайн. Но тогда это должна быть видеосвязь с интерактивными онлайн-досками, а не только письма и чаты.

2. Работающий продукт важнее исчерпывающей документации.

Чтобы клиенты были довольны, им нужен именно работающий продукт. Поэтому разработчики продукта должны фокусироваться именно на том, чтобы продуктом можно было как можно скорее воспользоваться, а не на составлении списков, диаграмм, требований, отчетов перед заказчиком. Чтобы укладываться в сжатые сроки с минимумом затрат, зачастую не стоит связывать себя документацией. Поддержка документации в адекватном продукту состоянии нередко замедляет разработку и требует неоправданно больших затрат.

3. Сотрудничество с заказчиком важнее согласований условий контракта.

Чтобы на выходе получить продукт, действительно ценный для заказчика, стоит отказаться от излишних деталей в контракте между подрядчиком и заказчиком (равно как и в требованиях внутреннего заказчика к внутреннему разработчику продукта). Будучи жестко заданы на старте, детали контракта мешают учитывать новые данные и приоритеты, появляющиеся лишь во время разработки. Чтобы бизнес-ценность продукта быстро росла, заказчик с разработчиком должны плотно общаться по ходу работы. В этом случае все возникающие изменения и проблемы оперативно обрабатываются обеими

сторонами. Чтобы такое сотрудничество исполнителя и заказчика стало возможным, нужно выстраивать их доверие друг к другу.

4. Готовность к изменениям важнее, чем следование плану.

Чтобы не откладывать риски проектов на последние стадии разработки (когда будет уже поздно уменьшать содержание работы, сдвигать срок или усиливать команду), Agile предлагает не только итеративность работы, но и готовность к изменениям на всех стадиях. Чтобы в первую очередь делалось самое ценное, текущее видение бизнес-ценности и позиционирования продукта должно быть прозрачно для разработчиков, а процесс их работы должен позволять вносить существенные изменения в прежние планы. В том числе, разработчики должны быть готовы добавлять в продукт незапланированные новые возможности, если они стали ценными в изменившейся ситуации. Готовность заказчика оперативно жертвовать какой-то частью запланированного также нужна в ситуации, когда исполнители столкнулись с непредвиденными проблемами в ходе разработки.

Схема работы по Agile изображена на рисунке. Проект разбивается не на последовательные фазы, а на маленькие подпроекты, которые затем «собираются» в готовый продукт. Инициация и верхнеуровневое планирование проводятся для всего проекта, а последующие этапы: разработка, тестирование и прочие проводятся для каждого мини-проекта отдельно. Это позволяет передавать результаты этих мини-проектов, так называемые инкременты, быстрее, а приступая к новому подпроекту (итарации) в него можно внести изменения без больших затрат и влияния на остальные части проекта.

Сильными сторонами Agile являются.

- Гибкость и адаптивность. Методология может подстроиться под практически любые условия и процессы организации.
- Именно быстрая и относительно безболезненная реакция на изменения является причиной тому, что многие крупные компании стремятся сделать свои процессы более гибкими.
- Agile отлично подходит для проектов с «открытым концом» — например, запуску сервиса или блога.
- Сфера Agile – разработка новых, инновационных продуктов. В проектах по разработке таких продуктов высока доля неопределённости, а информация о продукте раскрывается по ходу проекта.

Слабые стороны Agile.

- Agile — это всего лишь набор принципов и ценностей.
- Каждой команде придётся самостоятельно составлять свою систему управления, руководствуясь принципами Agile.
- Это непростой и длительный процесс, который потребует изменений всей организации, начиная процедурами и заканчивая базовыми ценностями. Это тернистый путь и не всем организациям он под силу.

Наиболее формализованной реализацией Agile является фреймворк Scrum. Фреймворк разработан в 1986 году. Сочетает в себе элементы классического процесса и идеи гибкого подхода к управлению проектами.

В Scrum работа ведется спринтами — одинаковыми по продолжительности короткими итерациями. Вся работа выполняется силами небольшой (до 10 человек) команды, в которую входят разработчики, владелец продукта (отвечающий за успех продукта) и скрам-мастер (отвечающий за эффективность и правильное применение Scrum). Команда самостоятельно решает, кто, что, когда и как делает.

Рассмотрим основные элементы Scrum.

Спринт – итерация разработки, в ходе которой создается новый результат. Жестко фиксирован во времени – от 1 до 4 недель. Чем короче спринт, тем гибче процесс разработки, поскольку после каждого спринта требования к системе могут корректироваться на основании обратной связи от заказчика. С другой стороны, при более длительном спринте снижаются издержки на совещания, и больше остается времени на решение задач проекта.

Беклог продукта – журнал пожеланий к продукту. Это список требований к системе, упорядоченный по приоритету – важности реализации. Приоритет исчисляется в очках. Система очков устанавливается командой самостоятельно. Журнал пожеланий могут дополнять все участники процесса.

Беклог спринта –журнал пожеланий для реализации в ходе спринта. Содержит функциональность, отобранную владельцем проекта для реализации на текущем спринте.

Встреча по упорядочиванию беклога продукта – эта встреча аналогична фазе планирования в классическом проектном управлении, и проводится в первый день каждого спринта. На ней рассматривается – что уже было сделано по проекту в целом, что ещё осталось сделать и принимается решение о том, что же делать дальше. Владелец продукта определяет, какие задачи на данном этапе являются наиболее приоритетными. Данный процесс определяет эффективность спринта, ведь именно от него зависит, какую ценность получит заказчик по итогам спринта.

Планирование спринта – после того, как Владелец продукта определил приоритеты, команда совместно решает, что же конкретно они будут делать во время грядущей итерации, как достигнуть поставленные перед спринтом цели. Команды могут применять различные инструменты планирования и оценки на данном этапе, лишь бы они не противоречили принципам и логике Scrum. Планирование спринта проводится в самом начале итерации, после встречи по упорядочиванию беклога продукта.

Ежедневные летучки – проводится в одно и то же время, в одном и том же месте длительностью не более 15 минут. Каждому участнику нужно ответить на 3 вопроса: что я сделал вчера, что я планирую сделать сегодня, что мне мешает.

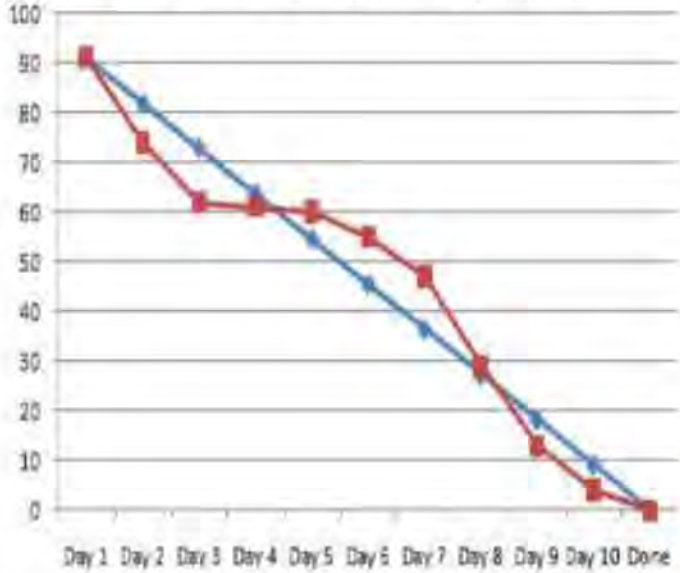
Подведение итогов спринта –собрание в конце спринта, длительностью не более четырех часов. Команда демонстрирует результаты спринта всем заинтересованным лицам. Привлекается максимальное количество зрителей. Все члены команды участвуют в демонстрации и показывают, что было сделано в течение спринта. Нельзя демонстрировать незавершенную функциональность.

Ретроспектива спринта – заключительное собрание, проводится по окончании подведения итогов спринта длительностью до трех часов. Обсуждаются результаты спринта. Члены команды высказывают своё мнение о прошедшем спринте и отвечают на два основных вопроса: что было сделано хорошо в прошедшем спринте, что надо улучшить в следующем? На этом собрании члены команды осуществляют улучшение процесса разработки: разрешают вопросы и фиксируют удачные решения.

Диаграмма сгорания задач –демонстрирует объем сделанной и оставшейся работы относительно сроков проекта. Диаграмма актуализируется ежедневно. Предусмотрены два вида диаграмм.

- Диаграмма сгорания для спринта — показывает, сколько задач сделано и сколько остаётся сделать в текущем спринте.
- Диаграмма сгорания всего проекта — показывает, сколько задач сделано и сколько остаётся сделать до завершения проекта.

Пример диаграммы сгорания приведен на рисунке.



Важным фактором успешной работы в Scrum является разделение ролей между владельцем продукта, Scrum-мастером и командой разработки.

Владелец продукта – это член команды, который отвечает за ценность проделываемой командой работы, несет ответственность перед заинтересованными лицами, является представителем заказчика. Его обязанности: своевременное предоставление требований к продукту, определение дат и содержания релизов. Является единственным человеком в

команде, кто отвечает за создание и контроль беклога продукта, за расстановку приоритетов элементов беклога.

Управление беклогом продукта включает в себя:

- описание элементов беклога ясным и понятным образом;
- управление порядком элементов беклога для наилучшего достижения целей и миссий;
- оптимизацию ценности работы, выполняемой разработчиками;
- обеспечение доступности, прозрачности и ясности беклога для всех участников процесса;
- понимание разработчиками элементов беклога.

Решения владельца продукта по исполнению тех или иных задач должны выполняться. Никто не может заставить команду разработки работать над другим набором требований. Он не может влиять на работу команды разработки, однако он всегда должен находиться рядом для разрешения возникающих вопросов.

Scrum мастер выполняет следующий набор функций:

- обучение команды особенностям Scrum–методологии;
- устранение проблем, образующихся внутри команды;
- выявление рисков и проблем, устранение препятствий, мешающих прогрессу работы;
- создание дружественных отношений в команде;
- слежение за процессами и их выполнением (разработки, тестирования);
- смена статусов задач в спринте;
- проведение ежедневных летучек;
- организация встреч перед спринтами;
- помощь владельцу продукта с беклогом продукта.

Перечень услуг Scrum мастера, предоставляемых владельцу продукта, команде разработки и организации в целом, приведены в таблице.

Владелец продукта	Команда разработки	Организация в целом
Помогает найти наиболее эффективные методы для управления беклогом продукта	Обучает команду самоорганизации и многофункциональности	Адаптирует Scrum в соответствии с потребностями организации
По необходимости помогает в организации процессов	Устраняет внешние препятствия	Помогает понять внешним к команде людям Scrum
При необходимости может выступить ведущим мероприятий Scrum.	По необходимости помогает а организации процессов	Обменивается лучшими практиками с другими Scrum мастерами

Команда разработки – самоорганизующаяся и самоуправляемая команда. Размер команды в идеале составляет от 5 до 9 человек. Это основная движущая сила, которая выполняет все технические задачи по разработке. Никто (и даже

Scrum мастер) не может указывать команде, как правильно превращать беклог продукта в инкременты работающей функциональности.

Каждый член команды имеет специализированные знания, но ответственность всегда на всей команде в целом. В команде есть только понятие разработчик продукта и исключены все должности, несмотря на то, чем занимается конкретный человек. Команда кроссфункциональна, и обладают всеми навыками, необходимыми для разработки инкремента продукта. Команда не имеет иерархии или подотделов. Всё решается внутри команды.

Ключевыми обязанностями команды являются:

- занимается непосредственной разработкой для заказчика;
- следит за собственной результативностью (совместно с Scrum Master);
- берет на себя решение по разработке и дизайну;
- создает беклог спринта;
- обеспечивает в конце спринта реализацию потенциально поставляемой функциональности;
- оценивает элементы беклога продукта;
- несет ответственность за результат перед владельцем продукта.

Команда должна быть уполномочена определять:

- что она создаст по окончании спринта;
- как ожидаемые результаты должны быть разбиты на задачи;
- кто выполнит задачу и в каком порядке они будут выполнены.

Помимо достоинств у Scrum есть и недостатки. К ним относятся.

- Высокая требовательность к команде: кроссфункциональность, члены команды должны быть «командными игроками», активно брать на себя ответственность и уметь самоорганизовываться. Подобрать такую зрелую команду очень непросто.
- Чрезмерная ориентированность на набор очков. Разработчики стараются набрать побольше очков во время спринта, что не приводит к улучшению кодовой базы, не делает её проще.
- Длительные митапы. Работникам приходится встраивать многочасовые совещания в свое расписание.
- Неизменность элементов беклога во время спринта. У программистов нет возможности перераспределить работу при обнаружении новых особенностей. Scrum не позволяет «перестраивать корабль прямо во время плавания», поэтому приходится ждать окончания спринта, чтобы внести изменения.

Еще одним вариантом практической реализации принципов Agile является Kanban. Эта методология очень похожа на схему промышленного производства. На входе в этот процесс попадает кусочек металла, а на выходе получается готовая деталь. Также и в Kanban, инкремент продукта передаётся вперёд с этапа на этап, а в конце получается готовый к поставке элемент.

Создатель Kanban вдохновлялся супермаркетами, а именно их принципом – «держи на полках только то, что нужно клиенту». А потому в Kanban разрешается оставить неоконченную задачу на одном из этапов, если её

приоритет изменился и есть другие срочные задачи. Неотредактированная статья для блога, подвешенная без даты публикации или часть кода функции, которую возможно не будут включать в продукт – всё это нормально для работы по Kanban.

Kanban намного менее строгий, нежели Scrum – он не ограничивает время спринтов, нет ролей, за исключением владельца продукта. Kanban даже позволяет члену команды вести несколько задач одновременно, чего не позволяет Scrum. Также никак не регламентированы встречи по статусу проекта – можно делать это как вам удобно, а можно не делать вообще.

Для работы с Kanban необходимо определить этапы потока операций. В Kanban они изображаются как столбцы, а задачи обозначают специальные карточки. Карточка перемещается по этапам, подобно детали на заводе, переходящей от станка к станку, и на каждом этапе процент завершения становится выше. На выходе мы получаем готовый к поставке заказчику элемент продукта. Доска со столбцами и карточками может быть как настоящей, так и электронной.

Схема работы по Kanban изображена на рисунке.

Kanban базируется на четырех основных элементах.

- Карточки. Каждая задача представляется в системе в виде собственной карточки, в которую заносится вся необходимая для работы информация о задаче: что нужно сделать, кто исполнитель, перечень имеющихся ресурсов, дедлайн, ограничения и т. п. При таком подходе карточка содержит все данные, которые могут потребоваться при выполнении задачи.
- Ограничение на количество задач на этапе. Устанавливается строгое ограничение на количество карточек, которые могут находиться на одном этапе. Это ограничение вполне естественно, поскольку работник не может выполнять одновременно большое количество задач. Благодаря ему наглядно выявляются «узкие места», в которых скапливается большое количество требуемых выполнения задач. Наличие такого «узкого места» должно служить поводом к реорганизации текущего процесса для ликвидации возможного «затора» по причине перегрузки некоторых исполнителей.

- Непрерывный поток. Как только на некотором этапе завершается работа над какой-либо задачей, ее карточка перемещается на следующий этап. Освободившееся место должна занять карточка другой задачи, ждущей своей очереди. Если очередь ожидают несколько задач, выбирается карточка с наивысшим приоритетом. При таком подходе обеспечена постоянная загруженность исполнителя и непрерывная работа над текущими, наиболее приоритетными, задачами.
- Постоянное улучшение: Залогом повышения качества результата является качество процесса. Kanban предполагает постоянное совершенствование процесса по результатам анализа текущей деятельности. Если что-то требует улучшения, оно должно быть улучшено.

Kanban имеет и ряд недостатков. К ним относятся:

- чем больше команда, тем более громоздкой и неудобной становится Kanban-доска;
- оптимальным является состав команды не более 5 человек;
- ориентирован только на управление текущим потоком задач, не имеет средств долгосрочного планирования;
- система выливается в нескончаемый поток задач.