Учреждение образования
«Гомельский государственный университет имени Франциска Скорины»

Факультет математики и технологий программирования
кафедра фундаментальной и прикладной математики

СОГЛАСОВАНО

Заведующий кафедрой
фундаментальной
и прикладной математики
Л. Н. Марченко
_____
24 02 2022 г.

СОГЛАСОВАНО

Декан
факультета математики
и технологий программирования
С. П. Жогаль
_____
24 02 2022 г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС
ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

МЕНЕДЖМЕНТ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1-31 03 07-01 Прикладная информатика
(программное обеспечение компьютерных систем)

Составители:
Л. Н. Марченко, кандидат технических наук, доцент

Рассмотрено и утверждено
на заседании кафедры фундаментальной
и прикладной математики
24. 02. 2022
Протокол № 7

Рассмотрено и утверждено
на заседании научно - методического совета университета
03 03 2022 г.,
протокол № 3

ОГЛАВЛЕНИЕ

1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к электронному учебно-методическому комплексу дисциплины «МЕНЕДЖМЕНТ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ» для специальности 1-31 03 07-01 Прикладная информатика (программное обеспечение компьютерных систем)

Электронный учебно-методический комплекс дисциплины «Менеджмент программного обеспечения» представляет собой комплекс систематизированных учебных, методических и вспомогательных материалов, предназначенных для использования в образовательном процессе специальности 1-31 03 07-01 Прикладная информатика (программное обеспечение компьютерных систем) и разработан в соответствии со следующими нормативными документами.

1. Положением об учебно-методическом комплексе на уровне высшего образования, утвержденном постановлением Министерства образования Республики Беларусь от 26.07.2011 №167.

2. Образовательный стандарт Республики Беларусь «Высшее образование. Первая ступень. Специальность 1-31 03 07 Прикладная информатика (по направлениям)».

3. Учебный план специальности 1-31 03 07 Прикладная информатика (по направлениям), направление специальности 1-31 03 07 01 Прикладная информатика (программное обеспечение компьютерных систем), утвержденный 15.04.2020 (регистрационный номер № G 31-01-20/УП).

4. Порядок разработки и утверждения учебных программ и программ практики для реализации содержания образовательных программ высшего образования, утвержденный Министром образования Республики Беларусь от 27.05.2019.

5. Типовая программа специальности 1-31 03 07 Прикладная информатика (по направлениям), утвержденная 20.04.2020 (регистрационный № ТД-G.641.тип.)

Данные документы предусматривают получение студентами теоретических и практических основ деятельности в области управления в сфере информационных технологий; формирование умений и навыков использования инструментов управления разработкой программного обеспечения. В них заложена задача формирования профессиональных и личностных компетенций в области экономической деятельности, связанной с созданием и продвижением информационных технологий и продуктов.

В Республике Беларусь возрастает количественная и качественная потребность в подготовке квалифицированных специалистов для существующих ИТ-компаний, в формировании новой предпринимательской среды (возникновение стартапов, инновационного предпринимательства, инфраструктуры коммерциализации и трансфера технологий). Специалистам в

сфере информационных технологий необходимы знания и практические навыки в области управления проектами, особенно для продвижения собственных ИТ-продуктов. Изучение дисциплины «Менеджмент программного обеспечения» позволит студентам быстрее адаптироваться к ведению бизнеса по своей профессии, принимать более обоснованные решения по выбору будущего конкретного направления своей деятельности, проанализировать свой предпринимательский потенциал.

Целью учебно-методического комплекса «Менеджмент программного обеспечения» является обеспечение теоретической и практической подготовки елей, активизации их учебно-познавательной деятельности по вопросам менеджмента программного обеспечения, развитие профессионально личностных компетенций в данной сфере, совершенствование умений и навыков самостоятельной учебной работы.

Задачами учебно-методического комплекса являются:

- раскрытие требований к содержанию учебной дисциплины «Менеджмента программного обеспечения»;
- обеспечение успешного усвоение теоретического материала по дисциплине,
- актуализирование использования традиционных форм и методов контроля знаний и стимулировать инновационные подходы к проверке и оценке знаний, умений и навыков студентов;
- повышение уровня творческой активности студентов через выполнение различных видов творческих заданий и проблемно-поисковых задач.

В структурном отношении учебно-методический комплекс «Менеджмент программного обеспечения» включает в себя четыре раздела: теоретический, практический, раздел контроля знаний, вспомогательный.

Теоретический раздел содержит основные положения, выносимые на лекции и предназначенные как для аудиторной работы со студентами, так и для самостоятельного изучения за рамками аудиторных часов. Посредством этих тем студенты могут получить представление об основных направлениях в области менеджмента программного обеспечения; об инструментах менеджера; об управлении рисками проекта, о проблемах формировании команды. При подготовке теоретической части широко использовалась учебная и научная литература по управлению проектами, а также информация из Интернет-ресурсов.

Практический раздел включает в себя в соответствии с учебным планом дисциплины практические занятия. Каждая тема занятия состоит из перечня вопросов для устного обсуждения содержания темы занятия (уровень узнавания), перечня задач (уровень формирования умений и навыков), заданий для письменного контроля знаний (уровень воспроизведения), заданий творческого характера (уровень применения знаний на практике). Дается список основной и дополнительной литературы для самостоятельного изучения и более глубокой подготовки к практическим занятиям. Предполагаются

различные формы работы с студентами на практических занятиях: устный опрос, защита рефератов, тестовые задания, групповая дискуссия, подготовка презентаций и другие. Особое место занимает использование метода проектов. Студентам при разработке проекта предлагалось раскрыть следующие моменты: бизнес-идея; команда, роли, мназвание; цель, миссия, предмет и объект; маркетинговые исследования в выбранной сфере деятельности; проект и его описание; планирование работ проекта (этапы); представление и защита проекта. Работа над проектами требует от студентов дополнительного самообучения, креативного мышления, актуализации полученных знаний из различных областей в решении конкретной задачи. Указанные формы работы способствуют не только усвоению знаний и их репродуктивному воспроизведению, но и видеть закономерности управления проектами.

Раздел контроля знаний представлен критериями оценок результатов учебной деятельности студентов по дисциплине. Здесь приводятся вопросы к зачету, критерии оценивания, примерные тестовые задания, критерии оценивания компьютерного тестирования. В данном разделе размещен также образец тестовых заданий, предназначенных для проверки уровня академических компетенций студентов по дисциплине. Они составлены в логической последовательности, и охватывают все темы учебной дисциплины. Представленные тестовые задания являются авторскими, и составлены на основе содержащегося в учебно-методическом комплексе материала к лекциям, а также дополнительных источников, рекомендуемых для самостоятельного изучения. Педагогические тесты включают в себя элемент выборы - варианты для выбора правильных ответов, которые формулируются в форме истинных или ложных высказываний.

Вспомогательный раздел содержит необходимые элементы учебно-программной документации: учебную программу по дисциплине «Менеджмент программного обеспечения» учреждения образования с пояснительной запиской и содержанием учебного материала.

Учебно-методический комплекс учебной дисциплины «Менеджмент программного обеспечения» предназначен для студентов 2-го курса

Дисциплина изучается студентами 2-го курса направления специальности 1-31 03 07-01 Прикладная информатика (программное обеспечение компьютерных систем) в 4-м семестре дневной формы получения высшего образования (первая ступень).

Общее количество часов – 54 (2 зачетных единиц); аудиторное количество часов – 34, из них лекций – 18 (из них управляемая самостоятельная работа студентов – 6 часов), лабораторные занятия – 16. Форма отчётности – зачет (4 семестр).

2 ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

2.1 Тематические планы лекций

ТЕМА 1 ОСНОВЫ МЕНЕДЖМЕНТА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1 История и основные понятия.
2 Отличия программной инженерии от других отраслей.
3 Эволюция подходов к управлению программными проектами.
4 Модели процесса разработки ПО.

ТЕМА 2 ОРГАНИЗАЦИЯ ГРУППОВОГО ВЗАИМОДЕЙСТВИЯ

1 Разработка плана управления человеческими ресурсами.
2 Управление командой.
3 Разрешение конфликтов.
4 Теории мотивации.

ТЕМА 3 МЕНЕДЖМЕНТ И МЕТОДОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1 Последовательный и итерационный процесс.
2 Принципы совмещения методологий.
3 Принципы работы с требованиями к программному обеспечению.

ТЕМА 4 УПРАВЛЕНИЕ ПРОГРАММНЫМИ ПРОЕКТАМИ

1 Жизненный цикл проекта.
2 Законы Брукса в жизненном цикле разработки программного обеспечения.
3 Функции управления разработкой ПО.
4 Фазы и продукты.

ТЕМА 5 ПЛАНИРОВАНИЕ ПРОЕКТА

1 Классификационные, иерархические признаки планирования.
2 Основные принципы планирования.
3 Технологии и системы планирования: ERP, CSRP.
4 Планирование и контроль менеджмента.

ТЕМА 6 СТАНДАРТЫ И СТАНДАРТИЗАЦИЯ В ОБЛАСТИ РАЗРАБОТКИ ПО

1 Понятие о стандартизации в области менеджмента.
2 Стандарт, свод правил, свод знаний в области ПО.
3 Элементы классификации стандартов в области ПО.

ТЕМА 7 ПРОДВИЖЕНИЕ ВЫСОКОТЕХНОЛОГИЧНОЙ ПРОДУКЦИИ НА ДЕЛОВОЙ И ПОТРЕБИТЕЛЬСКИЕ РЫНКИ

1 Системы маркетинговых исследований и маркетинговой информации.
2 Особенности рынка программного обеспечения B2B и его отличия от рынка B2C.
3 Продвижение программного обеспечения: стратегия коммуникации и стимулирования спроса.

4 Интернет-маркетинг

5 Понятие о стандартизации в области менеджмента.

ТЕМА 8 ПРОГНОЗИРОВАНИЕ

1 Основные методы прогнозирования.

2 Элементы технического анализа.

3 Экспертные и эвристические методы прогнозирования.

4 Оценка качества прогнозов

ТЕМА 9 УПРАВЛЕНИЕ РИСКАМИ

1 Понятия о рисках в области разработки ПО.

2 План управления рисками.

3 Идентификация рисков.

4 Работа с рисками: исключение, уменьшение, переключение, предупреждение ущерба.

5 Методический аппарат анализа риска.

2.2 Краткий конспект лекций

Тема 1 Основы менеджмента программного обеспечения

1 История и основные понятия

Программная инженерия есть применение определенного систематического измеримого подхода при разработке, эксплуатации и поддержке программного обеспечения.

Термин software (программное обеспечение, ПО) ввел в 1958 году всемирно известный статистик Джон Тьюкей (John Tukey). Термин software engineering (программная инженерия) впервые появился в названии конференции НАТО, состоявшейся в Германии в 1968 году и посвященной так называемому кризису программного обеспечения. С 1990-го по 1995 год велась работа над международным стандартом, который должен был дать единое представление о процессах разработки программного обеспечения. В результате был выпущен стандарт ISO/IEC 12207 [2]. В 2004 году в отрасли был создан основополагающий труд «Руководство к своду знаний по программной инженерии» (SWEBOK) [3], в котором были собраны основные теоретические и практические знания, накопленные в этой отрасли.

Во избежание двусмысленностей, но, не претендуя на академичность, позволю себе ввести рабочие определения ряда терминов, которые я буду в дальнейшем активно использовать.

Программирование - процесс отображения определенного множества целей на множество машинных команд и данных, интерпретация которых на компьютере или вычислительном комплексе обеспечивает достижение поставленных целей.

Цели могут быть любые: воспроизведение звука в динамике ПК, расчет траектории полета космического аппарата на Марс, печать годового балансового отчета и т.д. Важно то, что они должны быть определены. Это звучит банально, но сколько бы раз об этом не твердили ранее, по-прежнему, приходится сталкиваться с программными проектами, в которых отсутствуют какие-либо определенные цели.

Это отображение может быть очень простым, например, перфорирование машинных команд и данных на перфокартах. А может быть многоступенчатым и очень сложным, когда сначала цели отображаются на требования к системе, требования - на высокоуровневую архитектуру и спецификации компонентов, спецификации - на дизайн компонентов, дизайн - на исходный код. Далее исходный код при помощи компиляторов и сборщиков отображается на код развертывания, код развертывания - на вызовы функций ПО окружения (ОС, промежуточное ПО, базы данных), которое может располагаться на множестве

компьютеров, объединенных в сеть, и только после этого - в машинные команды и данные.

Профессиональное программирование (синоним производство программ) - деятельность, направленная на получение доходов при помощи программирования.

Принципиальным отличием от просто программирования является то, что имеется или, по крайней мере, предполагается некоторый потребитель, который готов платить за использование программного продукта. Отсюда следует важный вывод о том, что профессиональное производство программ это всегда коллективная деятельность, в которой участвуют минимум два человека: программист и потребитель

Профессиональный программист - человек, который занимается профессиональным программированием.

Профессионального программиста следует отличать от профессионала (мастера в программировании). Разброс профессионального мастерства в программировании достаточно широк и далеко не каждый, кто зарабатывает на жизнь программированием, является мастером, но об этом позже.

Программный продукт - совокупность программ и сопроводительной документации по их установке, настройке, использованию и доработке.

Согласно стандарту [EEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes] жизненный цикл программы, программной системы, программного продукта включает в себя разработку, развертывание, поддержку и сопровождение. Если программный продукт не коробочный, а достаточно сложный, то его развертывание у клиентов, как правило, реализуется отдельными самостоятельными проектами внедрения. Сопровождение включает в себя устранение критических неисправностей в системе и реализуется часто не как проект а, как процессная деятельность. Поддержка заключается в разработке новой функциональности, переработке уже существующей функциональности, в связи с изменением требований, и улучшением продукта, а также устранение некритических замечаний к ПО, выявленных при его эксплуатации (рисунок 1.1). Жизненный цикл программного продукта завершается выводом продукта из эксплуатации и снятием его с поддержки и сопровождения.

Рисунок 1.1 - Жизненный цикл программного продукта

Процесс разработки ПО - совокупность процессов, обеспечивающих создание и развитие программного обеспечения.

Самый распространенный процесс разработки ПО, который пришлось наблюдать за годы работы в отрасли, можно назвать «как получится». Это не означает, что процесса как такового нет. Он есть и, как правило, обеспечивает разработку ПО при приемлемых затратах и качестве, но этот процесс не документирован, является «знанием стаи», держится на людях

и передается из поколения в поколение. Целенаправленная работа по оценке эффективности и улучшению процесса не ведется.

Модель процесса разработки ПО - формализованное представление процесса разработки ПО. Часто при описании процессов вместо слова модель употребляется термин методология, что приводит к неоправданному расширению данного понятия.

Согласно SWEBOK 2004, программная инженерия включает в себя 10 основных и 7 дополнительных областей знаний, на которых базируются процессы разработки ПО. К основным областям знаний относятся следующие области:

1. Software requirements - программные требования.
2. Software design - дизайн (архитектура).
3. Software construction - конструирование программного обеспечения.
4. Software testing - тестирование.
5. Software maintenance - эксплуатация (поддержка) программного обеспечения.
6. Software configuration management - конфигурационное управление. Software engineering management - управление в программной инженерии.
7. Software engineering process - процессы программной инженерии.
8. Software engineering tools and methods - инструменты и методы.
9. Software quality - качество программного обеспечения.

Дополнительные области знаний включают в себя:

1. Computer engineering - разработка компьютеров.
2. Computer science - информатика.
3. Management - общий менеджмент.
4. Mathematics - математика.
5. Project management - управление проектами.
6. Quality management - управление качеством.
7. Systems engineering - системное проектирование.

Все это необходимо знать и уметь применять, для того чтобы разрабатывать ПО. Как видим, управление проектами, о котором мы будем говорить далее, лишь одна из 17 областей знаний программной инженерии, и то вспомогательная. Однако основной причиной большинства провалов программных проектов является именно применение неадекватных методов управления разработкой.

2 Отличия программной инженерии от других отраслей

Standish Group, проанализировав работу сотен американских корпораций и итоги выполнения нескольких десятков тысяч проектов, связанных с разработкой ПО пришла к следующим выводам:

- только 35 % проектов завершились в срок, не превысили запланированный бюджет и реализовали все требуемые функции и возможности;
- 46 % проектов завершились с опозданием, расходы превысили запланированный бюджет, требуемые функции не были реализованы в полном объеме;
- среднее превышение сроков составило 120%, среднее превышение затрат 100%, обычно исключалось значительное число функций;
- 19 % проектов полностью провалились и были аннулированы до завершения.

То, что производят программисты нематериально - это коллективные мысли и идеи, выраженные на языке программирования. Просто в силу уникальности отрасли опыт профессионалов, накопленный в материальном производстве и изложенный в стандарте PMI PMBOK [«PMBOK. Руководство к Своду знаний по управлению проектами», 3-е изд., PMI, 2004], мало способствует успеху в управлении программным проектом. Управлять разработкой ПО надо иначе.

Творчество - это интеллектуальная деятельность человека, законы которой нам неизвестны. Если бы мы знали законы творчества, то и картины, и стихи, и музыку, и программы уже давно бы создавали компьютеры. Творческое начало это то, что роднит программирование с наукой и искусством.

Творчество в программировании начинается с определения целей программы и заканчивается только тогда, когда в ее коде, написанном на каком-либо языке программирования, поставлена последняя точка. Попытки разделять программистов на творческую элиту, архитекторов и проектировщиков, и нетворческих программистов-кодеров не имеют под собой объективных оснований. Даже если алгоритм программы строго определен математически, два разных программиста его закодируют по-разному, и полученная программа будет иметь разные потребительские качества.

Творчество неразрывно связано с вдохновением, И чем сложнее задача, тем труднее извлечь это вдохновение из подсознания. Иногда для этого требуются часы, а иногда недели.

Программирование - это не искусство, в том смысле, что оно не является творческим отражением и воспроизведением действительности в художественных образах. Об искусстве в программировании можно и должно говорить только в смысле умения, мастерства, знания дела, как и в любой другой профессии. И как в любой другой профессии программистское мастерство может доставлять истинное эстетическое наслаждение, но только для людей, причастных к этой профессии.

Программирование - это не наука. Наработки математиков в области логики, теории информации, численных методов, реляционной алгебры, теории графов и некоторых других дисциплинах на долю процента не покрывают сложность программистских задач. В программировании нет системы знаний о закономерностях создания программ. Даже выдающиеся программисты не возьмут на себя смелость утверждать об архитектуре новой программной системы то, что она будет успешной. Хотя в программировании уже накоплен определенный опыт провалов, который может позволить искушенному программисту увидеть в архитектуре новой системы антипаттерны - источники серьезных будущих проблем. Но не более того.

Существующее состояние программной инженерии напоминает большую поваренную книгу с многочисленными описаниями рецептов однажды успешно приготовленных блюд из ингредиентов, которых в будущем уже не будет. Завтра в новой системе будут другие вычислительные машины, технологии, языки программирования, инструменты и окружающее ПО, новые проблемы взаимодействия с которыми обязательно придется решать.

Профессиональное творчество программиста принципиально отличается от творчества в науке и искусстве. Программистские задачи с каждым годом становятся все сложнее и объемнее, а сроки, за которые требуется решить эти задачи, наоборот, с каждым годом сокращаются. Поэтому современные программы создаются коллективами от нескольких до тысяч программистов, в то время как творческие деятели науки и искусства работают, как правило, в одиночку.

Программист тоже работает с абстракциями, но ему приходится держать в голове гораздо больше абстракций, чем любому ученому. Абстракции сопутствуют программисту на всех уровнях разработки программы от описания ее целей до исполняемого машинного кода. И этих уровней могут быть десятки. И на каждом уровне абстракций их деталей становится все больше и больше.

Дополнительно к абстрактному мышлению, программист должен обладать сильно выраженным системным мышлением, чтобы удерживать многочисленные взаимосвязи, существующие на всех уровнях программистских абстракций, а также взаимосвязи между этими уровнями. Еще одной сложностью является то, что все эти абстракции и взаимосвязи между ними изменяются во времени, и программист должен учитывать эту динамику.

Кроме того, программист должен обладать маниакальной усидчивостью, сосредоточенностью и упорством для перебора всех возможных вариантов поведения своих абстракций и доскональной проработки всех деталей. Проработка должна быть абсолютно точной и не должна содержать ни одной ошибки, неправильного, лишнего или отсутствующего символа исходного кода (а это порой миллионы строк). Инструменты программирования: синтаксические анализаторы, компиляторы и проч., - лишь

незначительно помогают в этой работе.

Еще одна особенность, которая присуща программистскому творчеству, это постоянное обновление информационных технологий, которые программисту необходимо знать и успешно применять в своей работе. Поэтому профессиональный программист должен, как сказал один из наших прежних вождей, «учиться, учиться и учиться». Программист должен удерживать в голове, постоянно пополнять и активно применять на практике гигабайты профессиональной информации. Это устройство компьютеров, компьютерных сетей и сетевые протоколы. Это операционные системы и языки программирования. Это программные интерфейсы промежуточного ПО и прикладных библиотек с особенностями и багами их реализации в конкретных продуктах. Это технологические стандарты, технологии разработки и инструменты, которые их поддерживают. Это архитектуры программных систем, паттерны и антипаттерны проектирования и много-много другой информации.

Программирование - это проектирование и только проектирование. Роль конструкторского бюро для программного проекта выполняют компилятор и сборщик программ. А программистским аналогом завода, который переводит конструкторскую документацию в продукт, доступный потребителю, служит вычислительный комплекс, на котором развертывается и выполняется созданная программа.

3 Эволюция подходов к управлению программными проектами

За 50 лет развития программной инженерии накопилось большое количество моделей разработки ПО. Интересно провести аналогию между историей развития методов, применяемых в системах автоматического управления летательными аппаратами, и эволюцией подходов к управлению программными проектами.

«Как получится». Разомкнутая система управления. Полное доверие техническим лидерам. Представители бизнеса практически не участвует в проекте. Планирование, если оно и есть, то неформальное и словесное. Время и бюджет, как правило, не контролируются. Аналогия: баллистический полет без обратной связи. Можно, но недалеко и неточно.

«Водопад» или каскадная модель. Жесткое управление с обратной связью. Расчет опорной траектории (план проекта), измерение отклонений, коррекция и возврат на опорную траекторию. Лучше, но не эффективно.

«Гибкое управление». Расчет опорной траектории, измерение отклонений, расчет новой попадающей траектории и коррекция для выхода на нее. «Планы - ничто, планирование - все» (Эйзенхауэр, Дуайт Дэвид)

«Метод частых поставок». Самонаведение. Расчет опорной траектории, измерение отклонений, уточнение цели, расчет новой попадающей траектории и коррекция для выхода на нее.

Классические методы управления перестают работать в случаях, когда структура и свойства управляемого объекта нам не известны и/или изменяются со временем. Эти подходы так же не помогут, если текущие свойства объекта не позволяют ему двигаться с требуемыми характеристиками. Например, летательный аппарат не может развить требуемое ускорение или разрушается при недопустимой перегрузке. Аналогично, если рабочая группа проекта не может обеспечить требуемую эффективность и поэтому постоянно работает в режиме аврала, то это приводит не к росту производительности, а к уходу профессионалов из проекта.

Когда структура и свойства управляемого объекта нам не известны, необходимо использовать адаптивное управление, которое, дополнительно к прямым управляющим воздействиям, направлено на изучение и изменение свойств управляемого объекта. Продолжая аналогию с управлением летательными аппаратами - это расчет опорной траектории, измерение отклонений, уточнение цели, уточнение объекта управления, адаптация (необходимое изменение) объекта управления, расчет новой попадающей траектории и коррекция для выхода на нее.

Для того чтобы понять структуру и свойства объекта и воздействовать на него с целью

их приведения к желаемому состоянию, в проекте должен быть дополнительный контур обратной связи - контур адаптации.

4 Модели процесса разработки ПО

Модели (или, как еще любят говорить, методологии) процессов разработки ПО принято классифицировать по «весу» - количеству формализованных процессов (большинство процессов или только основные) и детальности их регламентации. Чем больше процессов документировано, чем более детально они описаны, тем больше «вес» модели.

ГОСТ 19 «Единая система программной документации» и ГОСТ 34 «Стандарты на разработку и сопровождение автоматизированных систем» ориентированы на последовательный подход к разработке ПО. Разработка в соответствии с этими стандартами проводится по этапам, каждый из которых предполагает выполнение строго определенных работ, и завершается выпуском достаточно большого числа весьма формализованных и обширных документов. Таким образом, строгое следование этим гостам не только приводит к водопадному подходу, но и требует очень высокой степени формализованности разработки. На основе этих стандартов разрабатываются программные системы по госзаказам в России.

SW-CMM

В середине 80-х годов минувшего столетия Министерство обороны США крепко задумалось о том, как выбирать разработчиков ПО при реализации крупномасштабных программных проектов. По заказу военных Институт программной инженерии, входящий в состав Университета Карнеги-Меллона, разработал SW-CMM, Capability Maturity Model for Software в качестве эталонной модели организации разработки программного обеспечения.

Данная модель определяет пять уровней зрелости процесса разработки ПО.

1. Начальный — процесс разработки носит хаотический характер. Определены лишь немногие из процессов, и успех проектов зависит от конкретных исполнителей.
2. Повторяемый — установлены основные процессы управления проектами: отслеживание затрат, сроков и функциональности. Упорядочены некоторые процессы, необходимые для того, чтобы повторить предыдущие достижения на аналогичных проектах.
3. Определенный — процессы разработки ПО и управления проектами описаны и внедрены в единую систему процессов компании. Во всех проектах используется стандартный для организации процесс разработки и поддержки программного обеспечения, адаптированный под конкретный проект.
4. Управляемый — собираются детальные количественные данные по функционированию процессов разработки и качеству конечного продукта. Анализируется значение и динамика этих данных.
5. Оптимизируемый — постоянное улучшение процессов основывается на количественных данных по процессам и на пробном внедрении новых идей и технологий.

Документация с полным описанием SW-CMM занимает около 500 страниц и определяет набор из 312 требований, которым должна соответствовать организация, если она планирует аттестоваться по этому стандарту на 5-ый уровень зрелости.

RUP

Унифицированный процесс (Rational Unified Process, RUP) был разработан Филиппом Крачтеном (Philippe Kruchten), Иваром Якобсоном (Ivar Jacobson) и другими сотрудниками компании "Rational Software" в качестве дополнения к языку моделирования UML.. Модель RUP описывает абстрактный общий процесс, на основе которого организация или проектная команда должна создать конкретный специализированный процесс, ориентированный на ее потребности. Именно эта черта RUP вызывает основную критику - поскольку он может быть чем угодно, его нельзя считать ничем определенным. В результате такого общего построения RUP можно использовать и как основу для самого что ни на есть традиционного водопадного стиля разработки, так и в качестве гибкого процесса.

MSF

Microsoft Solutions Framework (MSF) - это гибкая и достаточно легковесная модель,

построенная на основе итеративной разработки. Привлекательной особенностью MSF является большое внимание к созданию эффективной и небюрократизированной проектной команды. Для достижения этой цели MSF предлагает достаточно нестандартные подходы к организационной структуре, распределению ответственности и принципам взаимодействия внутри команды.

PSP/TSP

Одна из последних разработок Института программной инженерии Personal Software Process / Team Software Process. Personal Software Process определяет требования к компетенциям разработчика. Согласно этой модели каждый программист должен уметь:

- □ учитывать время, затраченное на работу над проектом;
- □ учитывать найденные дефекты;
- □ классифицировать типы дефектов;
- □ оценивать размер задачи;
- □ осуществлять систематический подход к описанию результатов тестирования;
- □ планировать программные задачи;
- □ распределять их по времени и составлять график работы.
- □ выполнять индивидуальную проверку проекта и архитектуры;
- □ осуществлять индивидуальную проверку кода;
- □ выполнять регрессионное тестирование.

Team Software Process делает ставку на самоуправляемые команды численностью 320 разработчиков. Команды должны:

- □ установить собственные цели;
- □ составить свой процесс и планы;
- □ отслеживать работу;
- □ поддерживать мотивацию и максимальную производительность.

Последовательное применение модели PSP/TSP позволяет сделать нормой в организации пятый уровень CMM.

Agile

Основная идея всех гибких моделей заключается в том, что применяемый в разработке ПО процесс должен быть адаптивным. Они декларируют своей высшей ценностью ориентированность на людей и их взаимодействие, а не на процессы и средства. По сути, так называемые, гибкие методологии это не методологии, а набор практик, которые могут позволить (а могут и нет) добиваться эффективной разработки ПО, основываясь на итеративности, инкрементальности, самоуправляемости команды и адаптивности процесса.

Тяжелые и легкие модели производственного процесса имеют свои достоинства и свои недостатки (таблица 1.1).

Таблица 1.1 - Плюсы и минусы тяжелых и легких моделей процессов разработки ПО

Вес модели	**Плюсы**	**Минусы**
Тяжелые	Процессы рассчитаны на среднюю квалификацию исполнителей. Большая специализация исполнителей. Ниже требования к стабильности команды. Отсутствуют ограничения по объему и сложности выполняемых проектов.	Требуют существенной управленческой надстройки. Более длительные стадии анализа и проектирования. Более формализованные коммуникации.

Легкие	Меньше непроизводительных расходов, связанных с управлением проектом, рисками, изменениями, конфигурациями. Упрощенные стадии анализа и проектирования, основной упор на разработку функциональности, совмещение ролей. Неформальные коммуникации.	Эффективность сильно зависит от индивидуальных способностей, требуют более квалифицированной, универсальной и стабильной команды. Объем и сложность выполняемых проектов ограничены.

Алистер Коуберн, один из авторов «Манифеста гибкой разработки ПО» проанализировал очень разные программные проекты, которые выполнялись по разным моделям от совершенно облегченных и «гибких» до тяжелых (CMM-5) за последние 20 лет. Он не обнаружил корреляции между успехом или провалом проектов и моделями процесса разработки, которые применялись в проектах. Отсюда он сделал вывод о том, что эффективность разработки ПО не зависит от модели процесса, а также о том, чт :

- □ У каждого проекта должна быть своя модель процесса разработки.
- □ У каждой модели - свое время.

Это означает, что не существует единственного правильного процесса разработки ПО, в каждом новом проекте процесс должен определяться каждый раз заново, в зависимости от проекта, продукта и персонала, в соответствие с «Законом 4-х П» (рисунок 1.2). Совершенно разные процессы должны применяться в проектах, в которых участвуют 5 человек, и в проектах, в которых участвуют 500 человек. Если продуктом проекта является критическое ПО, например, система управления атомной электростанцией, то процесс разработки должен сильно отличаться от разработки, например, сайта «отдохни.ру». И, наконец, по-разному следует организовывать процесс разработки в команде вчерашних студентов и в команде состоявшихся профессионалов.

Рисунок 1.2 - «Закон 4-х П». Процесс в проекте должен определяться в зависимости от проекта, продукта и персонала

Команда, которая начинала проект, не остается неизменной, она проходит определенные стадии формирования и, как правило, количественно растет по мере развития проекта. Поэтому процесс должен постоянно адаптироваться к этим изменениям. Главный принцип: не люди должны строиться под выбранную модель процесса, а модель процесса

должна подстраиваться под конкретную команду, чтобы обеспечить ее наивысшую эффективность.

Что надо делать для успеха программного проекта

Стив Макконнелл в своей книге приводит тест программного проекта на выживание. Этот чек-лист из 33-х пунктов, который я считаю необходимым процитировать с небольшими корректировками. Руководитель программного проекта должен его периодически использовать для внутреннего аудита своих процессов.

Чтобы программный проект стал успешным, необходимо:

1. Четко ставить цели.

1.1. Концепция определяет ясные недвусмысленные цели
1.2. Все члены команды считают концепцию реалистичной.
1.3. У проекта имеется обоснование экономической эффективности.
1.4. Разработан прототип пользовательского интерфейса.
1.5. Разработана спецификация целевых функций программного продукта.
1.6. С конечными пользователями продукта налажена двухсторонняя связь

2. Определять способ достижения целей.

2.1. Имеется детальный письменный план разработки продукта.
2.2. В список задач проекта включены «второстепенные» задачи (управление конфигурациями, конвертация данных, интеграция с другими системами).
2.3. После каждой фазы проекта обновляется расписание и бюджет.
2.4. Архитектура и проектные решения документированы.
2.5. Имеется план обеспечения качества, определяющий тестирование и рецензирование.
2.6. Определен план многоэтапной поставки продукта.
2.7. В плане учтены обучение, выходные, отпуска, больничные.
2.8. План проекта и расписание одобрен всеми участниками команды.

3. Контролировать и управлять реализацией.

3.1. У проекта есть куратор. Это такой топ-менеджер исполняющей компании, который лично заинтересован в успехе данного проекта.
3.2. У проекта есть менеджер, причем только один!
3.3. В плане проекта определены «бинарные» контрольные точки.
3.4. Все заинтересованные стороны могут получить необходимую информацию о ходе проекта.
3.5. Между руководством и разработчиками установлены доверительные отношения.
3.6. Установлена процедура управления изменениями в проекте.
3.7. Определены лица, ответственные за решение о принятии изменений в проекте.
3.8. План, расписание и статусная информация по проекту доступна каждому участнику.
3.9. Код системы проходит автоматическое рецензирование.
3.10. Применяется система управления дефектами.

4. Анализировать угрозы и противодействовать им.

4.1. Имеется список рисков проекта. Осуществляется его регулярный анализ и обновление.
4.2. Руководитель проекта отслеживает возникновение новых рисков.
4.3. Для каждого подрядчика определено лицо, ответственное за работу с ним.

5. Создавать команду.

5.1. Опыт команды достаточен для выполнения проекта.
5.2. У команды достаточная компетенция в прикладной области.
5.3. В проекте имеется технический лидер.
5.4. Численность персонала достаточна.

5.5. У команды имеется достаточная сплоченность.
5.6. Все участники привержены проекту.

Оценка и интерпретация теста

Оценка: сумма баллов, каждый пункт оценивается от 0 до 3:

- 0 - даже не слышали об этом;
- 1 - слышали, но пока не применяем;
- 2 - применяется частично;
- 3 - применяется в полной мере.

Поправочные коэффициенты:

- для малых проектов (до 5 человек) - 1.5;
- для средних (от 5 до 20 человек) - 1.25.

Результат:

- <40 - завершение проекта сомнительно.
- 40-59 - средний результат. В ходе проекта следует ожидать серьезные проблемы.
- 60-79 - хороший результат. Проект, скорее всего, будет успешным.
- 80-89 - отличный результат. Вероятность успеха высока.
- >90 - великолепный результат. 100% шансов на успех.

Этот чек-лист перечисляет, что надо делать для успеха программного проекта, но не дает ответ на вопрос как это следует делать.

Выводы

То, что производят программисты нематериально - это коллективные мысли и идеи, выраженные на языке программирования. В силу уникальности отрасли опыт, накопленный в отраслях материального производства, мало способствует успеху в управлении программным проектом. Прямые аналогии с этими отраслями не работают. Управлять разработкой ПО надо иначе.

Не существует единственного правильного процесса разработки ПО. Эффективный производственный процесс должен основываться на итеративности, инкрементальности, самоуправляемости команды и адаптивности. Главный принцип: не люди должны строиться под выбранную модель процесса, а модель процесса должна подстраиваться под конкретную команду, чтобы обеспечить ее наивысшую производительность.

Чтобы программный проект стал успешным, необходимо:

1. Четко ставить цели.
2. Определять способ достижения целей.
3. Контролировать и управлять реализацией.
4. Анализировать угрозы и противодействовать им.
5. Создавать команду.

Тема 2 Организация группового взаимодействия

1 Разработка плана управления человеческими ресурсами

Управление человеческими ресурсами проекта включает в себя процессы по организации команды проекта и управления ей. Команда проекта состоит из людей, каждому из которых назначена определенная роль и ответственность за выполнение проекта. После распределения ролей и ответственности между членами команды проекта, они должны принимать активное участие в планировании проекта и принятии решений. Привлечение членов команды к участию на ранних стадиях проекта позволяет использовать имеющийся у них опыт при планировании проекта и укрепляет нацеленность команды на достижение результатов. По мере выполнения проекта профессиональный и численный состав членов команды проекта может меняться.

Участники проекта - лица или организации, предоставляющие услуги и осуществляющие

поставки в проекте, а также принимающие участие в управлении проектом (Например, Инвестор, Заказчик, Генподрядчик, Поставщик, Команда управления проектом и др.).

Процессы управления человеческими ресурсами проектов включают в себя следующее:

- планирование человеческих ресурсов – определение и документальное оформление ролей, ответственности и подотчетности, а также создание плана управления обеспечением проекта персоналом.

- набор команды проекта – привлечение человеческих ресурсов, необходимых для выполнения проекта.

- развитие команды проекта – повышение квалификации членов команды проекта и укрепление взаимодействия между ними с целью повышения эффективности исполнения проекта.

- управление командой проекта – контроль эффективности членов команды проекта, обеспечение обратной связи, решение проблем и координация изменений, направленных на повышение эффективности исполнения проекта.

Команда проекта в общем понимании – это группа специалистов, обладающих определенной квалификацией, знаниями, умениями, навыками и качествами, необходимыми для эффективного достижения поставленной перед ними общей цели.

Команда проекта создается руководителем проекта, задачей которого является подбор членов команды обеспечения:

- соответствия количественного и качественного состава команды целям и требованиям проекта;

- эффективной командную работу по управлению проектом;

- психологической совместимости членов команды и формирование единой «внутрипроектой» культуры;

- свободного внутрикомандного общения и выработки оптимального разрешения проблем, возникающих во время реализации проекта.

Критерии отбора в команду проекта:

- профессионализм;

- демонстрация способности работать в команде;

- желание брать на себя ответственность за принимаемые решения;

- самостоятельность, предприимчивость.

В жизненном цикле команды проекта можно выделить следующие этапы:

- формирование;

- этап конфликтов;

- установка норм общения;

- основной этап работы (рабочая стадия и реорганизация);

- заключительный этап (расформирование).

2 Управление коммуникациями проекта.

Коммуникация — информационное взаимодействие субъектов, характеризуемое следующими признаками: суверенитетом участников взаимодействия; суверенитетом их ценностных ориентации, интересов, представлений об объекте взаимодействия и отношения к нему; технологической обеспеченностью равноправного информационной: обмена; технологической обеспеченностью равного уровня информационной полноты знаний о ситуации и объекте взаимодействия. Управление коммуникациями проекта (управление взаимодействием, информационными связями) - управленческая функция, направленная на обеспечение своевременного сбора, генерации, распределения и сохранения необходимой проектной информации.

Управление коммуникациями проекта – это область знаний, включающая в себя процессы, необходимые для своевременного создания, сбора, распространения, хранения, получения и, в конечном итоге, использования информации проекта. Процессы управления

коммуникациями проекта предусматривают создание необходимых связей между людьми и информацией, которые требуются для успешного осуществления коммуникаций. Менеджеры проектов могут тратить чрезмерно много времени на коммуникации с командой проекта, участниками проекта, заказчиком и спонсором. Все, кто так или иначе вовлечен в проект, должны хорошо понимать, насколько коммуникации отражаются на протекании проекта в целом.

Процессы управления коммуникациями проекта включают в себя следующие элементы:

- планирование коммуникаций
- определение потребностей участников проекта в коммуникации и информации.
- распространение информации – своевременное предоставление необходимой информации участникам проекта.
- отчетность по исполнению – сбор и распространение информации о выполнении работ. Эта информация включает в себя отчеты о текущем состоянии, оценку прогресса и прогнозирование.

Управление коммуникациями проекта — управленческая функция, направленная на обеспечение своевременного сбора, генерации, распределения и сохранения необходимой проектной информации.

Под информацией понимают собранные, обработанные и распределенные данные. Чтобы быть полезной для принятия решений, информация должна быть предоставлена своевременно, по назначению и в удобной форме.

В качестве основных потребителей информации проекта выступают: проект-менеджер для анализа расхождений фактических показателей выполнения работ от запланированных и принятия решений по проекту; заказчик для осведомленности о ходе выполнения работ проекта; поставщики при возникновении потребности в материалах, оборудования и т. п., необходимых для выполнения работ; проектировщики, когда необходимо внести изменения в проектную документацию; непосредственные исполнители работ на местах. Управление коммуникациями обеспечивает поддержку системы связи между участниками проекта, передачу управленческой и отчетной информации, направленной на обеспечение достижения целей проекта. Каждый участник проекта должен быть подготовлен к взаимодействию в рамках проекта в соответствии с его функциональными обязанностями. Функция управления информационными связями включает в себя следующие процессы: планирование системы коммуникаций - определение информационных потребностей участников проекта; сбор и распределение информации - процессы регулярного сбора и своевременной доставки необходимой информации участникам проекта; отчетность о ходе выполнения проекта - обработка фактических результатов состояния работ проекта, соотношение с плановыми и анализ тенденций, прогнозирование; документирование хода работ - сбор, обработка и организация хранения документации по проекту.

План коммуникаций является составной частью плана проекта. Он включает в себя: план сбора информации, в котором определяются источники информации и методы ее получения; план распределения информации, в котором определяются потребители информации и способы ее доставки; детальное описание каждого документа, который должен быть получен или передан, включая формат, содержание, уровень детальности и используемые определения; план ввода в действие тех или иных видов коммуникаций; методы обновления и совершенствования плана коммуникаций.

План коммуникаций формализуется и детализируется в зависимости от потребностей проекта. В рамках проекта существует потребность в осуществлении различных видов коммуникаций: внутренние и внешние; формальные и неформальные; письменные и устные; вертикальные и горизонтальные.

Системы сбора и распределения информации должны обеспечивать потребности различных видов коммуникаций. Для этих целей могут использоваться автоматизированные и неавтоматизированные методы сбора, обработки и передачи информации.

Неавтоматизированные методы включают сбор и передачу данных на бумажных

носителях, проведение совещаний.

Автоматизированные методы предусматривают использование компьютерных технологий и современных средств связи для повышения эффективности взаимодействия: электронная почта, системы документооборота и архивирования данных.

Процессы сбора и обработки данных о фактических результатах и отображение информации о состоянии работ в отчетах обеспечивают основу для координации работ, оперативного планирования и управления. Отчетность о ходе выполнения включает; информацию о текущем состоянии проекта в целом и в разрезе отдельных показателей; информацию об отклонениях от базовых планов; прогнозирование будущего состояния проекта.

Основные промежуточные результаты хода работ должны быть формально задокументированы.

Документирование результатов хода работ включает в себя: сбор и верификацию окончательных данных; анализ и выводы о степени достижения результатов проекта и эффективности выполненных работ; архивирование результатов с целью дальнейшего использования.

Компьютерные системы ведения электронных архивов позволяют автоматизировать процессы хранения и индексации текстовых и графических документов, значительно облегчить доступ к архивной информации.

Информационная система управления проектом — организационнотехнологический комплекс методических, технических, программных и информационных средств, направленный на поддержку и повышение эффективности процессов управления проектом.

В процессе реализации проекта менеджерам приходится оперировать значительными объемами данных, которые могут быть собраны и организованы с использованием компьютера. Кроме того, многие аналитические средства, например, пересчет графика работ с учетом фактических данных, ресурсный и стоимостной анализ с подразумевают достаточно сложные для неавтоматизированного расчета алгоритмы.

Тема 3 Менеджмент и методологии разработки программного обеспечения

Процесс жизни любой системы или программного продукта может быть описан посредством модели жизненного цикла, состоящей из стадий. Модели могут использоваться для представления всего жизненного цикла от замысла до прекращения применения или для представления части жизненного цикла, соответствующей текущему проекту. Модель жизненного цикла представляется в виде последовательности стадий, которые могут перекрываться и (или) повторяться циклически в соответствии с областью применения, размером, сложностью, потребностью в изменениях и возможностях. Каждая стадия описывается формулировкой цели и выходов. Процессы и действия жизненного цикла отбираются и исполняются на этих стадиях для полного удовлетворения цели и результатам каждой стадии. Различные организации могут использовать различные стадии в пределах жизненного цикла. Однако каждая стадия реализуется организацией, ответственной за эту стадию, с надлежащим рассмотрением информации, имеющейся в планах жизненного цикла и решениях, принятых на предшествующих стадиях. Аналогичным образом организация, ответственная за текущую стадию, ведет записи принятых решений и записи допущений, относящихся к последующим стадиям данного жизненного цикла.

Под моделью жизненного цикла ПО понимается структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении ЖЦ. Модель ЖЦ зависит от спецификации, масштаба и сложности проекта и спецификации условий, в которых система создается и функционирует. Модель ЖЦ ПО включает в себя:

стадии, результаты выполнения работ на каждой стадии, ключевые события – точки завершения работ и принятия решений. Модель ЖЦ любого конкретного ПО определяет характер процесса его создания, который представляет собой совокупность упорядоченных во времени, взаимосвязанных и объединенных в стадии работ, выполнение которых необходимо и достаточно для создания ПО, соответствующего заданным требованиям. Под стадией понимается часть процесса создания ПО, ограниченная определенными временными рамками и 22 заканчивающаяся выпуском конкретного продукта (моделей, программных компонентов, документации), определяемого заданными для данной стадии требованиями. На каждой стадии могут выполнятся несколько процессов, определённых в стандарте ГОСТ Р ИСО/МЭК 12207-2010, и наоборот один и тот процесс может выполняться на различных стадиях. Соотношение между стадиями и процессами также определяется используемой моделью ЖЦ ПО. Далее рассмотрим модели и их классификации.

Каскадная модель

Первой моделью, получившей широкую известность и действительно структурирующей процесс разработки, является каскадная (водопадная) модель. Каждая стадия каскадной модели заканчивается получением некоторых результатов, которые служат в качестве исходных данных для следующей стадии. Требования к разрабатываемому ПО, определенные на стадии формирования требований, строго документируются в виде технического задания и фиксируются на все время разработки проекта.

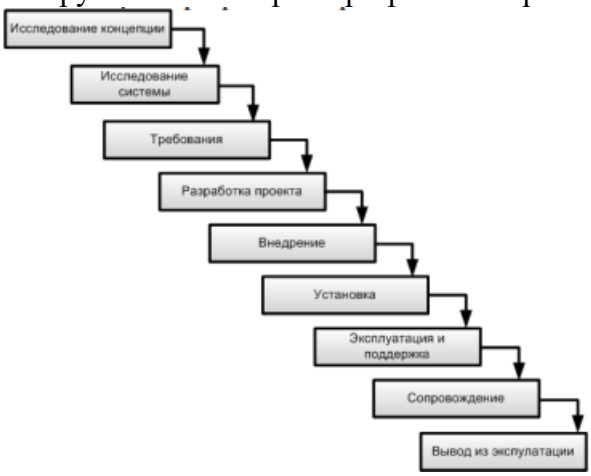

Рисунок 3.1 Классическая каскадная модель

Преимущества применения каскадной модели заключаются в следующем:

– на каждой стадии формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;

– выполняемые в логичной последователь стадии работ позволяют планировать сроки завершения всех работ и соответствующие затраты.

Каскадная модель может использоваться при создании ПО, для которого в самом начал разработки можно достаточно точно и полно сформулировать все требования. В то же время этот подход обладает рядом недостатков, вызванных прежде всего тем, чьл реальный процесс создания ПО никогда полностью не укладывался в такую жесткую схему.

Спустя непродолжительное время после появления на свет каскадная модель была доработана Уинстом Ройсом с учетом взаимозависимости этапов и необходимости возврата на предыдущие ступени, что может быть вызвано, например, неполнотой требований или ошибками в формировании задания. Процесс создания ПО носит как правило, итерационный характер: результаты очередной стадии часто вызывают изменения в проектных решениях, выработанных на более ранних стадиях. Таким образом, постоянно возникает потребность в возврате к предыдущим стадиям и уточнении или пересмотре ранее принятых решений. В

результате реальный процесс создания ПО принимает вид, изображенный на рисунке 3.2.

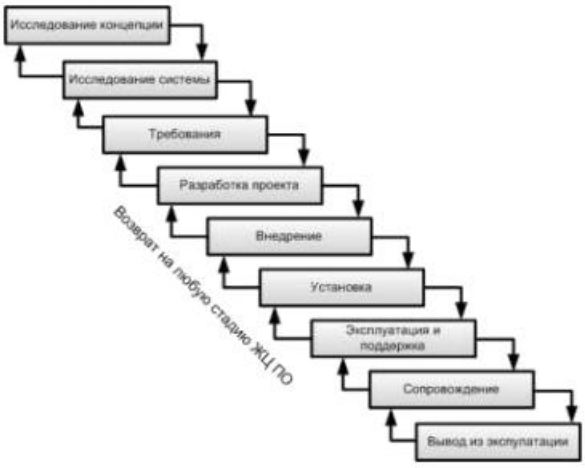

Рисунок 3.2 Модифицированная каскадная модель

Наиболее распространенным результатом каскадного похода к разработке ПО является поздняя неудача. Кажется, что проекты выполняются нормально, но только до тех пор, пока работы не вступят в завершающий этап, и тогда выясняется, что потребители недовольны созданным продуктом

V-образная модель, как разновидность каскадной модели

Основной принцип V-образной модели заключается в том, что детализация проекта возрастает при движении слева направо, одновременно с течением времени, и ни то, ни другое не может повернуть вспять. Итерации в проекте производятся по горизонтали, между левой и правой сторонами буквы.

V-модель – вариация каскадной модели, в которой задачи разработки идут сверху вниз по левой стороне буквы V, а задачи тестирования – вверх по правой стороне буквы V. Внутри V проводятся горизонтальные линии, показывающие, как результаты каждой из стадий разработки влияют на развитие системы тестирования на каждой из стадий тестирования. Модель базируется на том, что приемо-сдаточные испытания основываются, прежде всего, на требованиях, системное тестирование – на требованиях и архитектуре, комплексное тестирование – на требованиях, архитектуре и интерфейсах, а компонентное тестирование – на требованиях, архитектуре, интерфейсах и алгоритмах (рисунок 3.3).

Рисунок 3.3 V— образная каскадная модель

Особенностью данной модели является разбиение стадий на три логических этапа: проектирование (детализация требований), реализация, тестирование.

V-модель дает организациям и проектным группам руководство по выполнению и завершению проектов последовательным и воспроизводимым образом. Применение принципов V-модели гарантирует выявление и фиксацию требований пользователей. Утвержденные требования могут быть переведены в функции готового приложения, (и) приложение отражает требования пользователей.

Итеративный инкрементный подход к разработке (эволюционная модель)

1. Итеративная модель

Итеративная модель предполагает разбиение жизненного цикла проекта на последовательность итераций, каждая из которых напоминает «мини-проект», включая все фазы жизненного цикла в применении к созданию меньших фрагментов функциональности, по сравнению с проектом, в целом.

Цель каждой итерации – получение работающей версии программной системы, включающей функциональность, определенную интегрированным содержанием всех предыдущих и текущей итерации. Результата финальной итерации содержит всю требуемую функциональность продукта. Таким образом, с завершением каждой итерации, продукт развивается инкрементально.

Шансы успешного создания сложной системы будут максимальными, если она реализуется в серии небольших шагов и если каждый шаг заключает в себе четко определенный результат, а также возможность возврата к результатам предыдущей успешной итерации, в случае неудачи. Перед тем, как пустить в дело все ресурсы, предназначенные для создания ПО, разработчик имеет возможность получать обратную связь из реального мира (заказчиков, пользователей) и исправлять возможные ошибки в проекте.

Итеративная модель подразумевает возможность не только сборки работающей (с точки зрения результатов тестирования) версии системы - прототипа, но и её развертывания в реальных операционных условиях с анализом откликов пользователей для определения содержания и планирования следующей итерации. Поскольку на каждом шаге мы имеем работающую систему, то можно:

– очень рано начать тестирование пользователями;

– принять стратегию разработки в соответствии с бюджетом, полностью защищающую от перерасхода времени или средств (в частности, за счет сокращения второстепенной функциональности)

2 Инкрементная модель

Идея, лежащая в основе инкрементной модели, состоит в том, что программную систему следует разрабатывать по принципу приращений, так, чтобы разработчик мог использовать данные, полученные при разработке более ранних версий ПО. Новые данные получаются как в ходе разработки ПО, так и в ходе его использования, где это возможно. Ключевые этапы этого процесса – простая реализация подмножества требований к программе и совершенствование модели в серии последовательных релизов до тех пор, пока не будет реализовано ПО во всей полноте. В ходе каждой итерации организация модели изменяется, и к ней добавляются новые функциональные возможности. Для организации инкрементной разработки обычно выбирается характерный временной интервал, например, неделя. Затем в течение этого интервала происходит обновление проекта: добавляется новая документация как текстовая, так и графическая, расширяется набор тестов, добавляются новые программные коды и т. д. Теоретически шаги разработки могут выполняться и параллельно, но такой процесс очень сложно скоординировать. Инкрементная разработка проходит лучше всего, если следующая итерация начинается после того, как обновление всех артефактов в предыдущей итерации закончено, и существенно хуже, если время, требуемое на обновление

артефактов, значительно превышает выбранный интервал. В результате каждой итерации получается работающее, но не полнофункциональное ПО, которое еще не является программным продуктом и не подлежит распространению. В результате каждой итерации создается версия некоторой части ПО. Необходимо заметить, но как правило на каждой итерации определяются и реализуются новые требования, некоторые итерации могут быть целиком посвящены усовершенствованию существующей программы, например, с целью повышения ее производительности.

Спиральная модель, как разновидность эволюционной модели

В середине 1980-х годов Барри Боэм предложил свой вариант итерационной модели итеративной модели под названием «Спиральная модель». При использовании спиральной модели прикладное ПО создается в несколько итераций (витков спирали) методом прототипирования. Создание прототипов осуществляется в несколько итераций, или витков спирали. Каждая итерация соответствует созданию фрагмента или версии ПО, на ней уточняются цели и характеристики проекта, оценивается качество полученных результатов и планируются работы следующей итерации. На каждой итерации производится тщательная оценка риска превышения сроков и стоимости проекта, чтобы определить необходимость выполнения еще одной итерации, степень полноты и точности понимания требований к системе, а также целесообразность прекращения проекта.

Отличительной особенностью этой модели является специальное внимание рискам, влияющим на организацию жизненного цикла. Боэм формулирует 10 наиболее распространенных (по приоритетам) рисков:

– дефицит специалистов;

– нереалистичные сроки и бюджет;

– реализация несоответствующей функциональности;

– разработка неправильного пользовательского интерфейса;

– «золотая сервировка», перфекционизм, ненужная оптимизация и оттачивание деталей;

– непрекращающийся поток изменений;

– нехватка информации о внешних компонентах, определяющих окружение системы или вовлеченных в интеграцию;

– недостатки в работах, выполняемых внешними (по отношению к проекту) ресурсами;

– недостаточная производительность получаемой системы;

– «разрыв» в квалификации специалистов разных областей знаний.

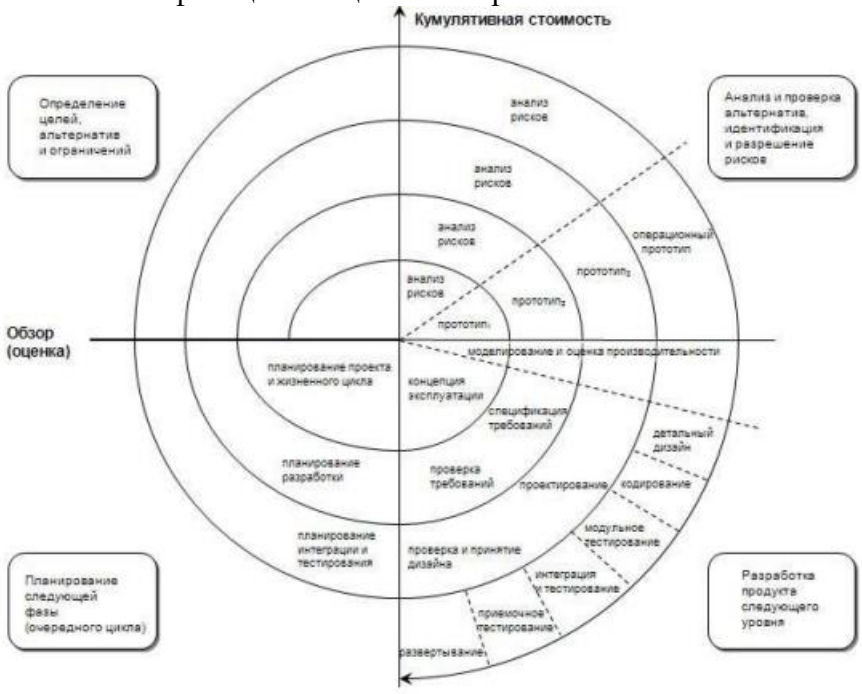

Рисунок 3.4 Оригинальная спиральная модель жизненного цикла разработки по Боэму

Основная проблема спирального цикла – определение момента перехода на следующую стадию. Для её решения необходимо ввести временные ограничения на каждую из стадий ЖЦ. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа закончена. План составляется на основе статических данных, полученных в предыдущих проектах, и личного опыта разработчика.

Достоинствами спиральной модели являются: ускорение разработки (ранее получение результата за счет прототипирования), постоянное участие заказчика в процессе разработки, разбиение большого объема работы на небольшие части, снижение риска.

2. Методологии разработки ПО

Методологии представляют собой ядро теории управления разработкой ПО. К существующей классификации в зависимости от используемой в ней модели жизненного цикла (каскадные и эволюционные) добавилась более общая классификация на прогнозируемые и адаптивные методологии. Прогнозируемые (предикативные) методологии фокусируются на детальном планировании будущего. Известны запланированные задачи и ресурсы на весь срок проекта. Команда с трудом реагирует на возможные изменения. План оптимизирован исходя из состава работ и существующих требований. Изменение требований может привести к существенному изменению плана, а также дизайна проекта. Адаптивные (гибкие) методологии нацелены на преодоление ожидаемой неполноты требований и их постоянного изменения. Когда меняются требования, команда разработчиков тоже меняется. Команда, участвующая в адаптивной разработке, с трудом может предсказать будущее проекта. Существует точный план лишь на ближайшее время. Более удаленные во времени планы существуют лишь как декларации о целях проекта, ожидаемых затратах и результатах. Среди адаптивных методологий: (Scrum, Crystal, Extreme Programming, Adaptive Software Development, DSDM, Feature Driven Development, Lean software development). Рассмотрим самые основные и популярные методологии.

1. RUP (Rational Unified Process)

Один из самых известных процессов, использующих итеративную модель разработки – RUP. Он был создан во второй половине 1990-х годов в компании Rational Software. Термином RUP обозначает как методологию, так и продукт компании IBM (ранее Rational) для управления процессом разработки. Методология RUP описывает абстрактный общий процесс, на основе которого организация или проектная команда должна создать специализированный процесс, ориентированный на ее потребности.

Основные характеристики:

– разработка требований, для описания требований в RUP используются прецеденты использования (use cases). Полный набор прецедентов использования системы вместе с логическими отношениями между ними называется моделью прецедентов использования. Каждый прецедент использования – это описание сценариев взаимодействия пользователя с системой, полностью выполняющего конкретную пользовательскую задачу. Согласно RUP все функциональные требования должны быть представлены в виде прецедентов использования.

– итеративная разработка, проект RUP состоит из последовательности итераций с рекомендованной продолжительностью от 2 до 6 недель. Перед началом очередной итерации определяется набор прецедентов использования, которые будут реализованы к её завершению.

2. Microsoft Solutions Framework (MSF)

Данная методология описывает подход и организацию работы при создании программных продуктов.

3. Scrum

Scrum предоставляет эмпирический подход к разработке ПО. Этот процесс быстр, адаптивен, умеет подстраиваться и отличен от каскадной модели. Scrum основан на повторяющихся циклах, это делает его более гибким и предсказуемым.

Для начала определим роли, которые участвуют в процессе: Scrum мастер (Scrum Master), Владелец продукта (Product Owner), Команда (Team).

Scrum Мастер - самая важная роль в методологии. Scrum Мастер отвечает за успех Scrum в проекте. Как правило, эту роль в проекте играет менеджер проекта или лидер команды (Team Leader). Важно подчеркнуть, что Scrum Мастер не раздает задачи членам команды. В Scrum команда является самоорганизующейся и самоуправляемой.

Основные обязанности Scrum Мастера таковы: – создает атмосферу доверия, – участвует в митингах в качестве фасилитатора - человека, обеспечивающий успешную групповую коммуникацию – устраняет препятствия – делает проблемы и открытые вопросы видимыми – отвечает за соблюдение практик и процесса в команде Scrum Мастер отслеживает прогресс команды при помощи Sprint Backlog, отмечая статус всех задач в спринте. Scrum Мастер может также помогать заказчику создавать список задач для команды

Product Owner - это человек, отвечающий за разработку продукта. Как правило представитель заказчика для заказной разработки. Владелец продукта - это единая точка принятия окончательных решений для команды в проекте, именно поэтому это всегда один человек, а не группа или комитет

Команда (Team) - в методологии Scrum команда является самоорганизующейся и самоуправляемой. Команда берет на себя обязательства по выполнению объема работ на спринт перед Владельцем продукта. Работа команды оценивается как работа единой группы. В Scrum вклад отдельных членов проектной команды не оценивается, так как это разваливает самоорганизацию команды. Размер команды ограничивается размером группы людей, способных эффективно взаимодействовать лицом к лицу. Типичные размер команды - 7 плюс минус 2. Команда в Scrum кроссфункциональна. В нее входят люди с различными навыками - разработчики, аналитики, тестировщики. Нет заранее определенных и поделенных ролей в команде, ограничивающих область действий членов команды. Команда состоит из инженеров, которые вносят свой вклад в общий успех проекта в соответствии со своими способностями и проектной необходимостью.

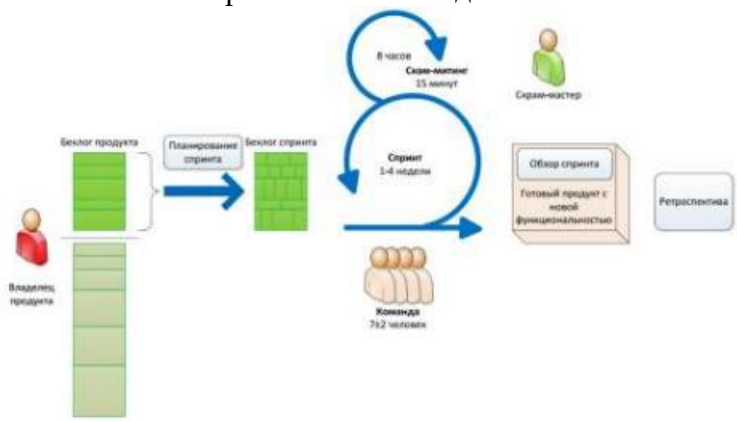

Рисунок 3.5. - Scrum

В основе лежат короткие ежедневные встречи – Scrum и циклические 30-дневные встречи, называемые спринтом. Результатом спринта является готовый продукт, который можно передавать заказчику (по крайней мере, система должна быть готова к показу заказчику).

Короткие спринты обеспечивают быструю обратную связь проектной команды с заказчиком. Заказчик получает возможность гибко управлять системой, оценивая результат спринта и предлагая улучшения к созданной функциональности. Такие улучшения попадают

в список имеющихся на данный момент бизнес-требований и технических требований к системе (Product Backlog), приоритезируются наравне с прочими требованиями и могут быть запланированы на следующий (или на один из следующих) спринтов. Каждый спринт представляет собой маленький «водопад». В течение спринта делаются все работы по сбору требований, дизайну, кодированию и тестированию продукта.

Цель спринта должна быть фиксированным. Это позволяет команде давать обязательства на тот объем работ, который должен быть сделан в спринте. Это означает, что Sprint Backlog не может быть изменен никем, кроме команды.

4. Экстремальное программирование (eXtreme Programming)

Методология XP, разработанная Кентом Беком (Kent Beck), Уордом Каннингемом (Ward Cunningham) и Роном Джеффрисом (Ron Jeffries), является сегодня одной из самых популярных гибких методологий. Она описывается как набор практик: игра в планирование, короткие релизы, метафоры, простой дизайн, переработки кода (refactoring), разработка «тестами вперед», парное программирование, коллективное владение кодом, 40-часовая рабочая неделя, постоянное присутствие заказчика и стандарты кода.

Интерес к XP рос снизу вверх – от разработчиков и тестировщиков, замученных тягостным процессом, документацией, метриками и прочим формализмом. Они не отрицали дисциплину, но не желали бессмысленно соблюдать формальные требования и искали новые быстрые и гибкие подходы к разработке высококачественных программ.

При использовании XP тщательное предварительное проектирование ПО заменяется, с одной стороны, постоянным присутствием в команде заказчика, готового ответить на любой вопрос и оценить любой прототип, а с другой – регулярными переработками кода (так называемый рефакторинг). Основой проектной документации считается тщательно прокомментированный код. Очень большое внимание в методологии уделяется тестированию. Как правило, для каждого нового метода сначала пишется тест, а потом уже разрабатывается собственно код метода до тех пор, пока тест не начнет выполняться успешно. Эти тесты сохраняются в наборах, которые автоматически выполняются после любого изменения кода.

Хотя парное программирование и 40-часовая рабочая неделя и являются, возможно, наиболее известными чертами XP, но все же носят вспомогательный характер и способствуют высокой производительности разработчиков и сокращению количества ошибок при разработке.

5. Crystal Clear

Легковесная гибкая методология, созданная Алистером Коуберном, которая предназначена для небольших команд в 6-8 человек для разработки некритичных бизнес-приложений. Как и все гибкие методологии, Crystal Clear больше опирается на людей, чем на процессы и артефакты. Crystal Clear использует семь методов/практик, три из которых являются обязательными:

– частая поставка продукта;
– улучшения через рефлексию;
– личные коммуникации;
– чувство безопасности;
– фокусировка;
– простой доступ к экспертам;
– качественное техническое окружение.

Методология Crystal Clear уступает XP по производительности, зато максимально проста в использовании. Она требует минимальных усилий для внедрения, поскольку ориентирована на человеческие привычки. Считается, что эта методология описывает тот естественный порядок разработки ПО, который устанавливается в достаточно квалифицированных коллективах, если в них не занимаются целенаправленным внедрением другой методологии.

Основные характеристики Crystal Clear:
– итеративная инкрементная разработка;
– автоматическое регрессионное тестирование;
– пользователи привлекаются к активному участию в проекте;
– состав документации определяется участниками проекта;
– как правило, используются средства контроля версий кода.
В графическом виде практики Crystal Clear можно изобразить таким образом:

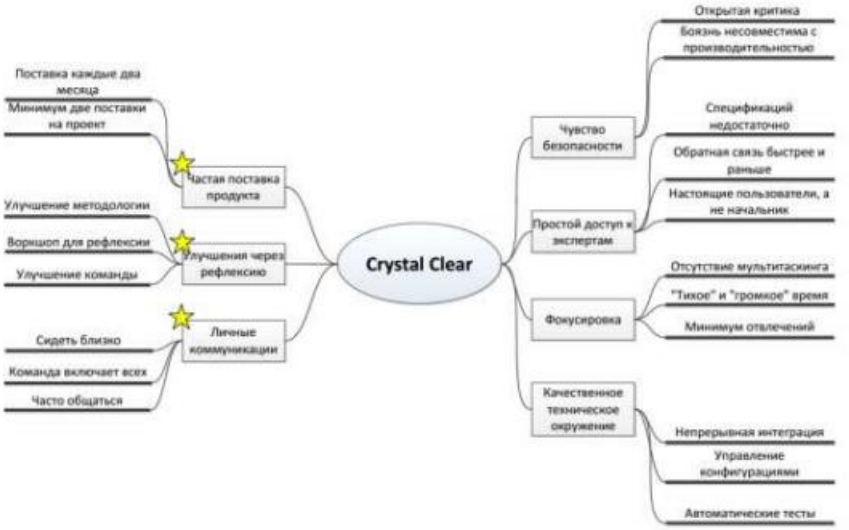

Рисунок 3.6 - Графическое представление Crystal Clear

Тема 4 Управление проектами. определения и концепции

1 Проект - основа инноваций

Классическое управление проектами выделяет два вида организации человеческой деятельности: операционная и проектная.

Операционная деятельность применяется, когда внешние условия хорошо известны и стабильны, когда производственные операции хорошо изучены и неоднократно испытаны, а функции исполнителей определены и постоянны. В этом случае основой эффективности служат узкая специализация и повышение компетенции.

Там, где разрабатывается новый продукт, внешние условия и требования к которому постоянно меняются, где применяемые производственные технологии используются впервые, где постоянно требуются поиск новых возможностей, интеллектуальные усилия и творчество, там требуются проекты.

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

У операционной и проектной деятельности есть ряд общих характеристик: выполняются людьми, ограничены доступностью ресурсов, планируются, исполняются и управляются. Операционная деятельность и проекты различаются, главным образом, тем, что операционная деятельность - это продолжающийся во времени и повторяющийся процесс, в то время как проекты являются временными и уникальными.

Ограничение по срокам означает, что у любого проекта есть четкое начало и четкое завершение. Завершение наступает, когда достигнуты цели проекта; или осознано, что цели проекта не будут или не могут быть достигнуты; или исчезла необходимость в проекте, и он прекращается.

Уникальность так же важное отличие проектной деятельности от операционной. Если

бы результаты проекта не носили уникальный характер, работу по их достижению можно было бы четко регламентировать, установить производственные нормативы и реализовывать в рамках операционной деятельности (конвейер). Задача проекта - достижение конкретной бизнес цели. Задача операционной деятельности - обеспечение нормального течения бизнеса.

Проект - это средство стратегического развития (рисунок 4.1). Цель - описание того, что мы хотим достичь. Стратегия - констатация того, каким образом мы собираемся эти цели достигать. Проекты преобразуют стратегии в действия, а цели в реальность.

Рисунок 4.1 - Проект - средство стратегического развития

Таким образом, каждая работа, которую выполняет конкретный сотрудник, привязывается к достижению стратегических целей организации.

Проекты объединяются в программы. Программа - ряд связанных друг с другом проектов, управление которыми координируется для достижения преимуществ и степени управляемости, недоступных при управлении ими по отдельности.

Проекты и программы объединяются в портфели. Портфель - набор проектов или программ и других работ, объединенных вместе с целью эффективного управления данными работами для достижения стратегических целей.

Проекты и управление ими существовали всегда. В качестве самостоятельной области знаний управление проектами начало формироваться в начале XX века. В этой дисциплине пока нет единых международных стандартов. Наиболее известные центры компетенции:

□ PMI, Project Management Institute, PMBOK - американский национальный стандарт ANSI/PMI 99-001-2004.

□ IPMA, International Project Management Association. В России - СОВНЕТ.

Простая мобилизация средств и усилий уже не может обеспечить прогресс. Согласно Ф. Брукса, «Если проект не укладывается в сроки, то добавление рабочей силы задержит его еще больше». Идею богатства теперь связывают не с деньгами, а с людьми, не с финансовым капиталом, а с «человеческим». Рынок труда превращается в рынок независимых специалистов и его участникам все больше известно о возможных вариантах выбора. Работники интеллектуального труда начинают самостоятельно определять себе цену.

Человечеству известны два вида деятельности. Репродуктивная деятельность (труд) является слепком, копией с деятельности другого человека либо копией своей собственной деятельности, освоенной в предшествующем опыте. Такая деятельность, как, например, труд токаря в любом механическом цеху, или рутинная повседневная деятельность менеджера-

управленца на уровне раз и навсегда усвоенных технологий. Продуктивная деятельность (творчество) - деятельность, направленная на получение объективно нового или субъективно нового (для данного работника) результата.

Репродуктивная деятельность уходит в прошлое. В постиндустриальном обществе интеллект - основная производственная сила. Сегодня от 70 до 80% всего, что сегодня делается людьми, производится при помощи их интеллекта. В любом товаре, сделанном в США, доля зарплаты составляет 70 процентов. Но это в среднем по всем товарам. Что касается разработки ПО, то почти все, что в этой отрасли производится, создается при помощи интеллекта.

Все меньший объем человеческой деятельности может быть организован в виде повторяющихся операций. Традиционный пример операционной деятельности - это работа бухгалтерии. Но жизнь так стремительно изменяется, что сегодня, по утверждению сведущих людей, подготовка и сдача годового финансового отчета каждый раз реализуется как самостоятельный проект.

Проект это основа инноваций. Сделать то, до чего другие компании еще не додумались, сделать это как можно быстрее, иначе это сделают другие. Предложить потребителю более качественный продукт или такой продукт, потребность в котором потребитель даже не может пока осознать.

2 Критерии успешности проекта

Задача проекта - достижение конкретной бизнес-цели, при соблюдении ограничений «железного треугольника» (рисунок 4.2). Это означает, что ни один из углов треугольника не может быть изменен без оказания влияния на другие. Например, чтобы уменьшить время, потребуется увеличить стоимость и/или сократить содержание.

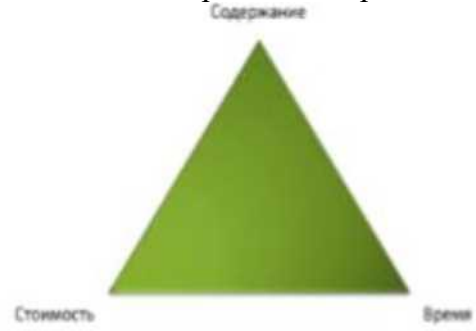

Рисунок 4.2 - «Железный треугольник» ограничений проекта

Согласно текущей редакции стандарта PMBOK, проект считается успешным, если удовлетворены все требования заказчика и участников проекта. Поэтому у проекта разработки ПО сегодня не три, а четыре фактора успеха:

1. Выполнен в соответствие со спецификациями.
2. Выполнен в срок.
3. Выполнен в пределах бюджета.
4. Каждый участник команды уходил с работы в 18:00 с чувством успеха.

Этот четвертый фактор успеха должен стать воспроизводимым, если предприятие хочет быть эффективным. Для успешного проекта характерно постоянное ощущение его участниками чувства удовлетворения и гордости за результаты своей работы, чувства оптимизма. Нет ничего более гибельного для проекта, чем равнодушие или уныние его участников.

Эффективность это отношение полученного результата к произведенным затратам. Нельзя рассматривать эффективность, исходя только из результативности: чем больше ты производишь, чем больше делаешь, тем выше твоя эффективность. С таким подходом можно «зарезать на ужин курицу, несущую золотые яйца». Затраты не следует путать с инвестициями. Оплата аренды, электроэнергии, коммунальные платежи - затраты. Создание

и закрепление эффективной команды - это стратегическое приобретение компании. Обучение участников проекта - инвестиции. Вложение в людей - это увеличение числителя в формуле эффективности. Уход из компании всех профессионалов после проекта, выполненного по принципу «любой ценой», - затраты, причем очень тяжело восполняемые. Нарастающая конкуренция указывает на совершенно четкий тренд в мировой экономике - персонал - это форма инвестиций, активов, которые нужно уметь наращивать, управлять и сохранять. Сегодня люди - это капитал.

Современное предприятие обязано относиться к своим работникам так же, как к своим лучшим клиентам. Главный капитал современной компании - это знания. Большая часть этих знаний неотъемлема от их носителя - человека. Те предприятия, которые этого не поняли, не выживут потому, что не смогут быть эффективными. Сегодня эффективное предприятие - это сервис. Предприятие, с одной стороны, предоставляет услуги и продукты своим клиентам, а с другой, - рабочие места для профессионального персонала. Принципы «Одно предприятие на всю жизнь», «Работай продуктивно, а предприятие о тебе позаботится» - уходят в прошлое. Посмотрите на рынок рабочей силы в ИТ - правила устанавливают профессионалы.

Проект и организационная структура компании

Организационная структура компании отражает ее внутреннее устройство, потоки управляющих воздействий, распределение труда и специфические особенности производства. Функциональная и проектная организации - противоположные полюса, а матричная организация - промежуточные состояния. Нет одной лучшей организационной структуры. Нет смысла противопоставлять функциональные структуры и проектные организации.

Синоним функциональной структуры - иерархическая структура (рисунок 4.3).

Рисунок 4.3 - Функциональная структура

Функциональная структура имеет следующие особенности:

□ Сохраняется принцип единоначалия

□ Понятные и стабильные условия работы

□ Хорошо приспособлены для операционной деятельности.

□ Специализация подразделений позволяет накапливать экспертизу.

□ Затруднено принятие решений и коммуникации между исполнителями. Осуществляются только через руководство.

□ Управление сконцентрировано и держится на компетенции высшего руководства

□ Как правило, неэффективен контроль за ходом проекта (нет целостной картины) Функциональная структура предполагает многоуровневую иерархию. Руководители функциональных подразделений это начальники управлений, начальники подчиненных им

служб, отделов, лабораторий, секторов, групп. А еще у каждого начальника есть заместитель и, порой, не один. Примеры: министерства, ведомства, научные институты и предприятия советского периода.

На другом краю спектра организационных структур находится проектная структура (рисунок 4.4).

Рисунок 4.4 - Проектная структура

В чисто проектных организациях:

- □ Проект организуется как самостоятельное производственное подразделение.
- □ Персонал на проект набирается по временным контрактам.
- □ После завершения проекта персонал увольняется.
- □ Медленный старт.
- □ Опыт не аккумулируется.
- □ Команды не сохраняются.

Проектные организации не самые эффективные, но порой единственно возможные для выполнения проектов, которые физически удалённы от исполняющей организации, например, строительство нового нефтепровода.

В разработке ПО наиболее распространена матричная организация. Различают три вида матричной организационной структуры: слабая, сбалансированная и сильная (рисунок 4.5 - рисунок 2.7). Причем, в компаниях, которые занимаются продуктовой разработкой ПО, функциональные подразделения определяются в соответствие с линейкой продуктов. Например, отдел разработки CRM-систем, отдел разработки финансовых систем, отдел разработки дополнительных продуктов.

В компаниях, которые ориентированы в основном на заказную разработку ПО, функциональные подразделения чаще объединяются в соответствие с используемыми информационными технологиями. Например, отдел разработки баз данных, отдел разработки J2EE-приложений, отдел веб-разработок, отделы тестирования, документирования и т.д.

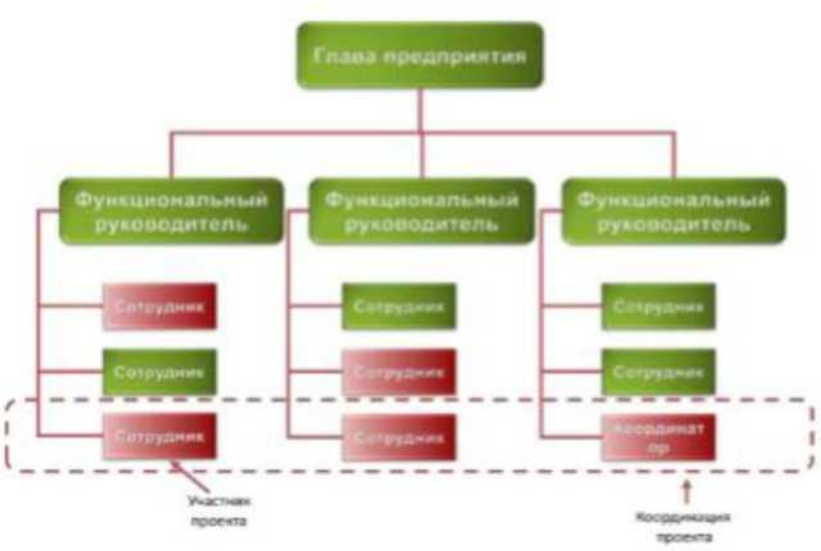

Рисунок 4-5 - Слабая матрица

В слабой матрице роль и полномочия сотрудника, который координирует проект, сильно ограничены. Реальное руководство проектом осуществляет один из функциональных руководителей. Координатор проекта, его еще часто называют «трекер», помогает этому руководителю собирать информацию о статусе выполняемых проектных работ, учитывает затраты, составляет отчеты.

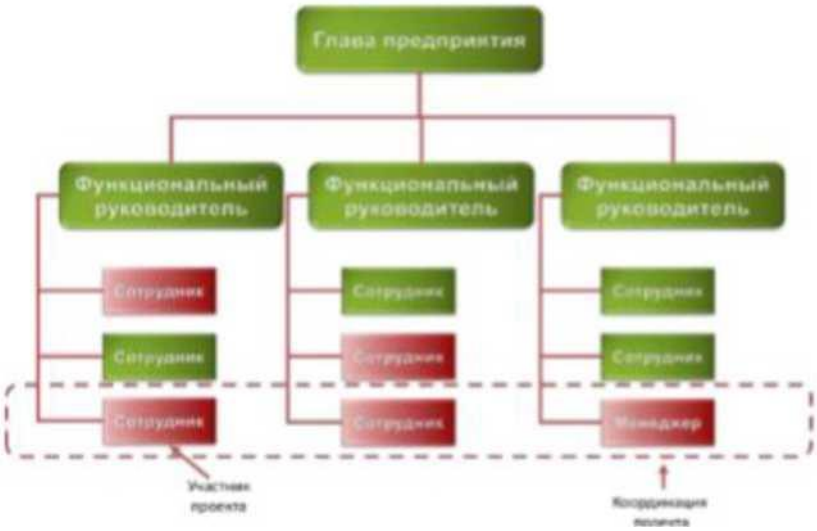

Рисунок 4.6 - Сбалансированная матрица

Сбалансированная матрица характеризуется тем, что появляется менеджер проекта, который реально управляет выделенными на проект ресурсами. Он планирует работы, распределяет задачи среди исполнителей, контролирует сроки и результаты, несет полную ответственность за достижение целей проекта, при соблюдении ограничений.

В сбалансированных матрицах наиболее ярко проявляется проблема двойного подчинения. Руководитель функционального подразделения и менеджер проекта имеют примерно равное влияние на материальный и профессиональный рост разработчиков.

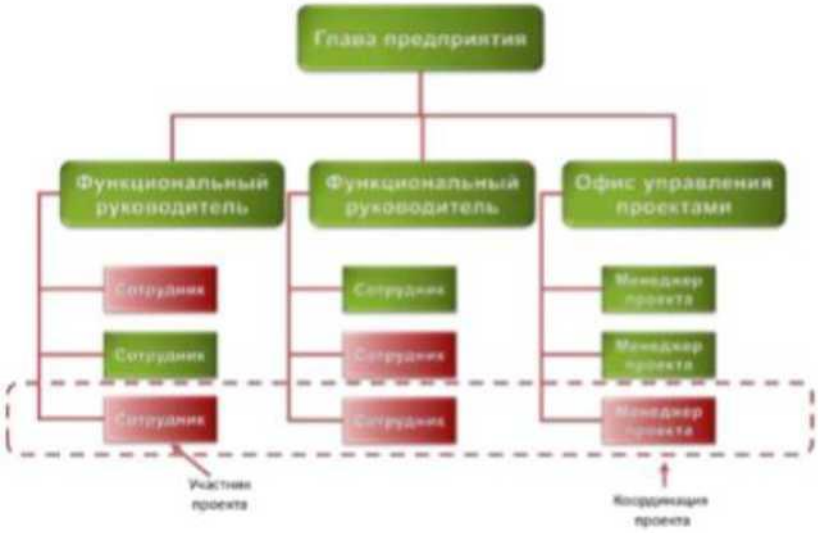

Рисунок 4-7 - Сильная матрица

В сильной матрице признается, что проектное управление является самостоятельной областью компетенции, в которой необходимо накапливать экспертизу и использовать общие ресурсы. Поэтому в сильной матрице менеджеры проектов объединяются в самостоятельное функциональное подразделение - офис управления проектами (ОУП). ОУП разрабатывает корпоративные политики и стандарты в области проектного управления, планирует и осуществляет профессиональное развитие менеджеров.

Одной из особенностей матричных структур является то, что они становятся «плоскими», исчезает многоступенчатая иерархия. Предприятие, как правило, делится на функциональные отделы, в которых работают специалисты разных категорий, напрямую подчиняющиеся начальнику отдела. Начальники лабораторий, секторов, групп упраздняются за ненадобностью. В матричных структурах роль начальника функционального подразделения в производственном процессе заметно снижается, по сравнению с функциональными структурами. В его компетенции остаются вопросы стратегического развития функционального направления, планирование и развитие карьеры сотрудников, вопросы материально-технического обеспечения работ. Следует учитывать, что такое перераспределение полномочий и ответственности от функциональных руководителей к менеджерам проектов часто служит источником конфликтов в компаниях при их переходе от функциональной структуры к матричной.

3 Организация проектной команды

Каждый проект разработки ПО имеет свою организационную структуру, которая определяет распределение ответственности и полномочий среди участников проекта, а также обязанностей и отношений отчетности. Чем меньше проект, тем больше ролей приходится совмещать одному исполнителю.

Роли и ответственности участников типового проекта разработки ПО можно условно разделить на пять групп:

1. Анализ. Извлечение, документирование и сопровождение требований к продукту.
2. Управление. Определение и управление производственными процессами.
3. Производство. Проектирование и разработка ПО.
4. Тестирование. Тестирование ПО.
5. Обеспечение. Производство дополнительных продуктов и услуг.

Группа анализа включает в себя следующие роли:

□ Бизнес-аналитик. Построение модели предметной области (онтологии).

□ Бизнес-архитектор. Разрабатывает бизнес-концепцию системы. Определяет общее видение продукта, его интерфейсы, поведение и ограничения.

□ Системный аналитик. Отвечает за перевод требований к продукту в функциональные требования к ПО.

□ Специалист по требованиям. Документирование и сопровождение требований к продукту.

□ Менеджер продукта (функциональный заказчик). Представляет в проекте интересы пользователей продукта.

Группа управления состоит из следующих ролей:

□ Руководитель проекта. Отвечает за достижение целей проекта при заданных ограничениях (по срокам, бюджету и содержанию), осуществляет операционное управление проектом и выделенными ресурсами.

□ Куратор проекта. Оценка планов и исполнения проекта. Выделение ресурсов.

□ Системный архитектор. Разработка технической концепции системы. Принятие ключевых проектных решений относительно внутреннего устройства программной системы и её технических интерфейсов.

□ Руководитель группы тестирования. Определение целей и стратегии тестирования, управление тестированием.

□ Ответственный за управление изменениями, конфигурациями, за сборку и поставку программного продукта.

В производственную группу входят:

□ Проектировщик. Проектирование компонентов и подсистем в соответствие с общей архитектурой, разработка архитектурно значимых модулей.

Проектировщик базы данных.

□ Проектировщик интерфейса пользователя.

□ Разработчик. Проектирование, реализация и отладка отдельных модулей системы.

В большом проекте может быть несколько производственных групп, ответственных за отдельные подсистемы. Как правило, проектировщик выполняет роль лидера группы и управляет своим подпроектом или пакетом работ. Стоит не забывать, что руководитель проекта делегирует полномочия, но не ответственность.

Группа тестирования в проекте состоит из следующих ролей:

□ Проектировщик тестов. Разработка тестовых сценариев.

□ Разработчик автоматизированных тестов.

□ Тестировщик. Тестирование продукта. Анализ и документирование результатов.

Участники группы обеспечения, как правило, не входят в команду проекта. Они выполняют работы в рамках своей процессной деятельности. К группе обеспечения можно отнести следующие проектные роли:

□ Технический писатель.

□ Переводчик.

□ Дизайнер графического интерфейса.

□ Разработчик учебных курсов, тренер.

□ Участник рецензирования.

□ Продажи и маркетинг.

□ Системный администратор.

□ Технолог.

□ Специалист по инструментальным средствам.

□ Другие.

В зависимости от масштаба проекта одну роль могут исполнять несколько человек. Например, разработчики, тестировщики, технические писатели. Некоторые роли всегда должен исполнять только один человек. Например, Руководитель проекта, Системный архитектор. Один человек может исполнять несколько ролей. Возможны следующие совмещения ролей:

□ Руководитель проекта + системный аналитик (+ системный архитектор)
□ Системный архитектор + разработчик
□ Системный аналитик + проектировщик тестов (+ технический писатель)
Системный аналитик + проектировщик интерфейса пользователя
□ Ответственный за управление конфигурациями + ответственный за сборку и поставку (+ разработчик)

Крайне нежелательно совмещать следующие роли:
□ Разработчик + руководитель проекта
□ Разработчик + системный аналитик.
□ Разработчик + проектировщик интерфейсов пользователя.
□ Разработчик + тестировщик

Не раз приходилось наблюдать, как в критические периоды проекта его менеджер-разработчик с увлечением правит очередные баги, а проектная команда в полном составе стоит у него за спиной и наблюдает за этим процессом. Это плохой пример руководства проектом.

Программисты любят и умеют программировать. Пусть они этим и занимаются. Не стоит загружать программистов несвойственной для них работой. В каждом проекте разработки программного продукта много других работ: бизнес-анализ, проектирование эргономики, графический дизайн, разработка пользовательской документации. Эти работы с программированием не имеют ничего общего. Для них требуются совершенно другая квалификация и другой склад мышления.

При кустарном производстве программ эти задачи, как правило, поручаются программистам, которые это делать не умеют и не любят. Получается обычно плохо, да еще и дорого. В силу своей интроверсии, граничащей с аутизмом, программист просто не в состоянии увидеть свою программу чужими глазами - глазами пользователей. Никто уже не хочет работать с программами с технологической парадигмой навороченного пользовательского интерфейса - кустарным творением программистов - когда для того чтобы работать с системой, надо обязательно знать, как она устроена. Это типичное творение программиста, которому гораздо важнее видеть, как работает его программа, чем разбираться в том, что она делает для пользователя. Поэтому, необходимо привлекать в проектную команду бизнес-аналитиков, эргономистов, художниковдизайнеров, документалистов. Разделение труда и специализация - залог перехода от кустарного производства к более эффективному промышленному производству.

Из профессиональных программистов получаются отличные тестировщики. Лучшая команда тестирования, которую я встречал, была в Luxoft. Это были маститые программисты из одного академического НИИ с опытом 20-30 лет. Они не осваивали новые программистские технологии, но исключительно эффективно ломали то, что было сделано на их основе. Однако, совмещать одновременно роли программиста и тестировщика - плохая практика. Хороший программист убежден, что он пишет программы правильно и ему психологически тяжело допустить, что где-то в его коде может быть ошибка. А ошибки есть всегда!

Организационная структура проекта обязательно должна включать в себя эффективную систему отчетности, оценки хода выполнения проекта и систему принятия решений. Можно рекомендовать еженедельные собрания по статусу проекта, на которых анализируются риски, оцениваются результаты, достигнутые на предыдущей неделе, и уточняются задачи на новый период.

В модели Scrum рекомендуются ежедневные совещания по состоянию работ - «Stand Up Meeting», но это применимо, скорее, для небольших рабочих групп от 3 до 5 разработчиков. Хотя в критические периоды проекта, приходилось проводить и ежедневные совещания.

Важно помнить, что организационная структура проекта - «живой» организм. Она начинает складываться на стадии планирования и может меняться в ходе проекта.

Нестабильность организационной структуры (частые замены исполнителей) - серьезная проблема в управлении сложными программными проектами, поскольку существует время вхождения в контекст проекта, которое может измеряться месяцами.

4 Жизненный цикл проекта. Фазы и продукты

Ранее уже отмечалось, что каждый программный продукт имеет свой жизненный цикл, в который проект разработки очередного релиза входит как одна из фаз. Аналогично, каждый проект разработки ПО имеет свой собственный жизненный цикл, который состоит из четырех фаз (рисунок 2.8).

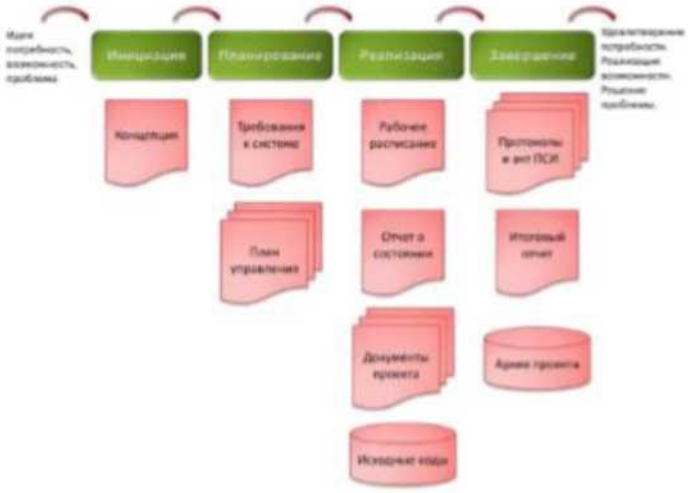

Рисунок 2.8 - Жизненный цикл и основные продукты программного проекта

На фазе инициации проекта необходимо понять, что и зачем мы будем делать - разработать концепцию проекта. Фаза планирования определяет, как мы будем это делать. На фазе реализации происходит материализация наших идей в виде документированного и протестированного программного продукта. И, наконец, на фазе завершения мы должны подтвердить, что мы разработали именно тот продукт, который задумали в концепции проекта, а также провести приемо-сдаточные испытания (ПСИ) продукта на предмет соответствия его свойств, определенным ранее требованиям.

Как правило, редкий проект выполняется в соответствие с первоначальными планами, поэтому важным элементом фазы завершения является «обратная связь»: анализ причин расхождения и усвоение уроков на будущее. Помним, что управляющая система без обратной связи не может быть устойчивой.

Еще одна особенность проекта по сравнению с операционной деятельностью; если в операционной деятельности ресурсы расходуются более-менее равномерно по времени, то в проектном управлении расходование ресурсов в единицу времени имеет явно выраженное колоколообразное распределение (рисунок 2.9).

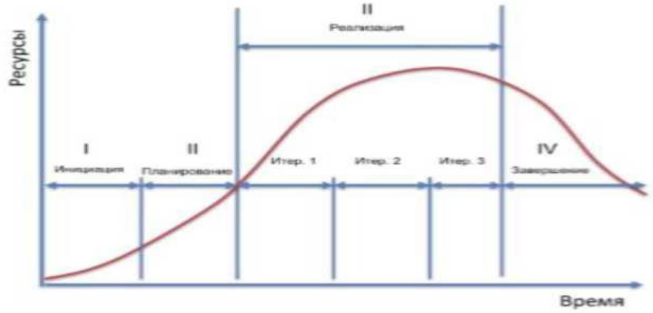

Рисунок 2.9 - Распределение ресурсов по фазам проекта

Проект часто начинается с идеи, которая появляется у одного человека. Постепенно, по мере формулирования, анализа и оценки этой идеи, привлекаются дополнительные

специалисты. Еще больше участников требуется на фазе планирования проекта. Пик потребления ресурсов приходится на фазу реализации.

Выводы

Проект - этот средство стратегического развития. Цель - описание того, что мы хотим достичь. Стратегия - констатация того, каким образом мы собираемся эти цели достигать. Проекты преобразуют стратегии в действия, а цели в реальность.

Участников типового проекта разработки ПО можно условно разделить на пять групп ролей:

1. Анализ. Извлечение, документирование и сопровождение требований к продукту.
2. Управление. Определение и управление производственными процессами.
3. Производство. Проектирование и разработка ПО.
4. Тестирование. Тестирование ПО.
5. Обеспечение. Производство дополнительных продуктов и услуг.

У программного проекта имеется четыре фактора, которые определяют его успешность:

1. Выполнен в соответствие со спецификациями.
2. Выполнен в срок.
3. Выполнен в пределах бюджета.
4. Каждый участник команды уходил с работы в 18:00 с чувством успеха.

Тема 5 Планирование проекта

1 Уточнение содержания и состава работ

«Если не получается проглотить слона целиком, то его надо порезать на отбивные». Человечество пока не придумало ничего более эффективного для решения сложной задачи, чем анализ и ее декомпозиция на боле простые подзадачи, которые, в свою очередь, могут быть разделены на еще боле простые подзадачи и так далее. Получается некоторая иерархическая структура, дерево, в корне которого находится проект, а на листьях элементарные задачи или работы, которые надо выполнить, чтобы завершить проект в условиях заданных ограничений.

Иерархическая структура работ (ИСР) (Work Breakdown Structure, WBS) - ориентированная на результат иерархическая декомпозиция работ, выполняемых командой проекта для достижения целей проекта и необходимых результатов. С ее помощью структурируется и определяется все содержание проекта. Каждый следующий уровень иерархии отражает более детальное определение элементов проекта.

Основой для разработки ИСР служит концепция проекта, которая определяет продукты проекта и их основные характеристики. ИСР обеспечивает выявление всех работ, необходимых для достижения целей проекта. Многие проекты проваливаются не от того, что у них нет плана, а от того что в этом плане забыты важные работы, например, тестирование и исправление ошибок, и продукты проекта, например пользовательская документация. Поэтому, если ИСР составлена корректно, то любая работа, которая в нее не вошла не может считаться работой по проекту.

ИСР делит проект на подпроекты, пакеты работ, подпакеты. Каждый следующий уровень декомпозиции обеспечивает последовательную детализацию содержания проекта, что позволяет производить оценку сроков и объемов работ. ИСР должна включать все промежуточные и конечные продукты.

Выполнять декомпозицию работ проекта можно по-разному. Например, ГОСТ 19.102-77 предусматривает каскадный подход и определяет следующие стадии разработки программной системы:

1. Техническое задание
2. Эскизный проект
3. Технический проект

4. Рабочий проект
5. Внедрение

Если следовать этому стандарту, то на первом уровне ИСР должны находиться именно эти проектные продукты. Если бы пришлось разрабатывать АСУ для управления ядерным реактором или пилотируемым космическим аппаратом, то именно так и следовало поступать. Однако в коммерческой разработке ПО такой подход не эффективен. Как мы уже говорили, современный процесс разработки коммерческого ПО должен быть инкрементальным. Это означает, что на верхнем уровне декомпозиции нашего проекта должны находиться продукты проекта, а на следующем уровне - компоненты, из которых эти продукты состоят. Компоненты далее могут быть декомпозированы на «фичи» - функции, которые они должны реализовывать.

Выделение компонентов, составляющих программный продукт, это элемент высокоуровневого проектирования, которое мы должны выполнить на фазе планирования проекта, не дожидаясь проработки всех функциональных требований к разрабатываемому ПО. Компонентами могут быть как прикладные подсистемы, так и инфраструктурные или ядерные, например, подсистема логирования, безопасности, библиотека визуальных компонентов GUI.

При составлении базового плана работ не стоит стремиться максимально детализировать все работы. ИСР не должна содержать слишком много уровней, достаточно 3-5. Например, ИСР нашего проекта-примера разработки «Автоматизированной системы продажи документации» может выглядеть следующим образом.

Пример: Иерархическая структура работ проекта разработки «Автоматизированной системы продажи документации» (курсивом выделены контрольные точки проекта).

1. Проект разработки «Автоматизированной системы продажи документации»
1.1. Подготовка технического задания на автоматизацию
1.1.1.1. Проведение аналитического обследования
1.1.1.2. Разработка функциональных требований
1.1.1.3. Разработка требований базовому ПО
1.1.1.4. Разработка требований к оборудованию и к операционносистемному ПО
1.1.1.5. Согласование и утверждение ТЗ
1.1.1.6. ТЗ утверждено
1.2. Поставка и монтаж оборудования
1.2.1. Разработка спецификации на оборудование
1.2.2. Закупка и поставка оборудования
1.2.3. Монтаж оборудования
1.2.4. Установка и настройка опреационно-системного ПО
1.2.5. Монтаж оборудования завершен
1.3. Поставка и установка базового ПО
1.3.1. Разработка спецификаций на базовое ПО
1.3.2. Закупка базового ПО
1.3.3. Развертывание и настройка базового ПО
1.3.4. Базовое ПО установлено у заказчика
1.4. Разработка и тестирование прикладного ПО
1.4.1. Разработка спецификаций на прикладное ПО
1.4.2. Установка и конфигурирование рабочей среды
1.4.3. Проектирование и разработка ПО
1.4.3.1. Авторизация и аутентификация пользователей.
1.4.3.2. Разработка подсистемы заказа документации
1.4.3.2.1. Просмотр каталога продуктов.
1.4.3.2.2. Поиск продуктов по каталогу.
1.4.3.2.3. Заказ выбранных продуктов.
1.4.3.2.4. Просмотр информации о статусе заказа.

1.4.3.2.5. Информирование клиента об изменении статуса заказа.

1.4.3.2.6. Подсистема заказа документации передана в тестовую эксплуатацию (на серверах разработчика).

1.4.3.3. Разработка подсистемы обработки заказов

1.4.3.3.1. Просмотр и обработка заказов исполнителями из службы продаж.

1.4.3.3.2. Просмотр статистики поступления и обработки заказов за период.

1.4.3.3.3. Подсистема обработки заказов передана в тестовую эксплуатацию на оборудовании Заказчика

1.4.3.4. Разработка подсистемы сопровождения каталога

1.4.3.4.1. Подготовка и сопровождение каталога продукции.

1.4.3.5. Исправление ошибок

1.4.4. Тестирование ПО

1.4.4.1. Раунд

1 1.4.4.2. Раунд 2

1.4.4.3. Раунд 3

1.4.4.4. Выходное тестирование

1.4.5. Документирование прикладного ПО

1.5. Обучение пользователей

1.5.4. Подготовка учебных курсов

1.5.5. Обучение сотрудников Отдела 123

1.5.6. Обучение руководства ОАО XYZ

1.5.7. Обучение администраторов системы

1.6. Ввод в опытную эксплуатацию

1.6.4. Развертывание и настройка прикладного ПО

1.6.5. Проведение приемо-сдаточных испытаний

1.6.6. Акт передачи системы в опытную эксплуатацию утвержден

1.7. Сопровождение системы в период опытной эксплуатации

1.8. Система передана в промышленную эксплуатацию

Должна быть установлена персональная ответственность за все части проекта (подпроекты и пакеты работ). Для каждого пакета работ должен быть четко определен результат на выходе. Работы и оценки проекта должны быть согласованы с ключевыми участниками команды, руководством компанииисполнителя и, при необходимости, с заказчиком. В результате согласования члены команды принимают на себя обязательства по реализации проекта, а руководство принимает на себя обязательства по обеспечению проекта необходимыми ресурсами.

ИСР является одним из основных инструментов (средств) в механизме управления проектом, с помощью которого измеряется степень достижения результатов проекта. Важнейшая ее функция это обеспечить консистентное представление всех у частников проекта относительно того, как будет делаться проект. В последующем базовый план будет служить ориентиром для сравнения с текущим исполнением проекта и выявления отклонений для целей управления.

2 Планирование управления содержанием

Одна из распространенных «болезней» программных проектов называется «ползучий фичеризм». Это, когда к изначально спроектированной будке для любимой собаки сначала пристраивают сарайчик для хранения садового инвентаря, а потом и домик в несколько этажей для ее хозяина. И все это пытаются построить на одном и том же фундаменте и из тех же самых материалов. Эта болезнь стала причиной летального исхода многих проектов разработки ПО.

Поэтому, сразу, как только удалось стабилизировать и согласовать ИСР, необходимо разработать план управления содержанием проекта. Для этого следует:

□ Определить источники запросов на изменение.

□ Установить порядок анализа, оценки и утверждения/отклонения изменения

содержания.

□ Определить порядок документирования изменений содержания.

□ Определить порядок информирования об изменении содержания.

Первая задача, которую необходимо решить при анализе запроса на изменения - выявить объекты изменений: требования, архитектура, структуры данных, исходные коды, сценарии тестирования, пользовательская документация, проч. Затем требуется спроектировать и детально описать изменения во всех выявленных объектах. И наконец, следует оценить затраты на внесение изменений, тестирование изменений и регрессионное тестирование продукта и их влияние на сроки проекта.

Эта работа, которая потребует затрат рабочего времени и порой значительных разных специалистов: аналитиков, проектировщиков, разработчиков, тестировщиков, наконец, менеджера проекта. Поэтому эта работа должна обязательно быть учтена в плане.

3 Планирование организационной структуры

Организационная структура - это согласованное и утвержденное распределение ролей, обязанностей и целей деятельности ключевых участников проекта. Она в обязательном порядке должна включать в себя систему рабочих взаимоотношений между рабочими группами проекта, систему отчетности, оценки хода выполнения проекта и систему принятия решений. Следует помнить, что организационная структура проекта - «живой» организм. Она начинает складываться на стадии планирования и должна меняться по ходу проекта.

Нестабильность организационной структуры - частая смена исполнителей - может стать серьезной проблемой в управлении проектом, поскольку, существует цена замены, которая определяется временем вхождения нового участника в контекст проекта.

Планирование управления конфигурациям

Конфигурационное управление один из важных процессов производства программного обеспечения. Об этой области знаний написана не одна книга. Мы будем говорить только о том, что эта работа должна быть спланирована.

План проекта должен включать в себя работы по обеспечению единого хранилища всей проектной документации и разрабатываемого программного кода, обеспечению сохранности и восстановление проектной информации после сбоя. Работы по настройке рабочих станций и серверов, используемых участниками проектной команды, тоже должны войти в план. Кроме этого в плане должны содержаться работы, необходимые для организации сборки промежуточных выпусков системы, а также ее конечного варианта.

Эти работы, как правило, выполняет один человек - инженер по конфигурациям. Если проект небольшой, то эта роль может быть дополнительной для одного из программистов. «Размазывать» эту работу на всех участников проекта, во-первых, неэффективно. Установка и конфигурирование среды разработки, например, баз данных и серверов приложений, требует определенных компетенций и знаний особенностей конкретных версий продуктов. Если эти навыки придется осваивать всем разработчикам, то на это уйдет слишком много рабочего времени. Во-вторых, «размазывание» работ по управлению конфигурациями может привести к коллективной безответственности, когда никто не знает, от чего не собирается проект и как откатиться к консистентной версии.

Управление конфигурациями может многократно усложниться, если проектной команде параллельно с разработкой новой функциональности продукта приходится поддерживать несколько релизов этого продукта, которые были установлены ранее у разных клиентов. Все эти работы должны быть учтены в плане проекта.

4 Планирование управления качеством

Обеспечение качества еще одна из базовых областей знаний в программной инженерии. Относительно того, что такое качество ПО и как его эффективно обеспечивать, можно рассуждать очень и очень долго. В нашем курсе мы ограничимся утверждением о том, что обеспечение качества - это важная работа, которая должна быть спланирована заранее и выполняться по ходу всего программного проекта, а не только во время приемосдаточных

испытаний.

При планировании этой работы необходимо понимать, что продукт проекта не должен обладать наивысшим возможным качеством, которое недостижимо за конечное время. Необходимое качество продукта определяется требованиями к нему. И еще. Основная задача обеспечения качества - это не поиск ошибок в готовом продукте (выходной контроль) а их предупреждение в процессе производства. Для примера, гладкость обработки детали на токарном станке только случайно может оказаться соответствующей требуемому качеству в 1 микрон, если шпиндель, в котором крепится деталь, плохо центрован.

План управления качеством должен включать в себя следующие работы:

□ Объективную проверку соответствия программных продуктов и технологических операций применяемым стандартам, процедурам и требованиям.

□ Определение отклонений по качеству, выявление их причин, применение мер по их устранению, а также контроль исполнения принятых мер и их эффективности.

□ Представление высшему руководству независимой информации о несоответствиях, не устраняемых на уровне проекта.

Помимо перечисленных разделов план проекта должен включать еще:

□ План управления рисками

□ Оценку трудоемкости и сроков работ

5 Базовое расписание проекта

После определения трудоемкости работ необходимо определить график их выполнения и общие сроки реализации проекта - составить расписание работ по проекту. Базовое расписание - утвержденный план-график с указанными временными фазами проекта, контрольными точками и элементами иерархической структуры работ.

Базовое расписание может быть наиболее наглядно представлено диаграммой Ганта. В этой диаграмме плановые операции или элементы иерархической структуры работ перечислены с левой стороны, даты отображаются сверху, а длительность операций показана горизонтальными полосками от даты начала до даты завершения.

Базовое расписание это, как правило, элемент контракта с заказчиком. Контрольные точки (вехи) должны служить точками анализа состояния проекта и принятия решения «GO/NOT GO», поэтому они должны зримо демонстрировать статус проекта. Контрольная точка «Проектирование завершено» - плохо. Наиболее эффективный подход - метод последовательных поставок: контрольная точка «Завершено тестирование требований 1, 3, 5, 7»

Если работы не связаны между собой, то любую из них мы можем начинать и завершать, когда нам удобно. Все работы можно делать параллельно и в этом случае минимальная длительность проекта равна длительности самой долгой работы. Однако, на практике между работами существуют зависимости, которые могут быть «жесткими», например, анализ - проектирование - кодирование - тестирование и документирование конкретной функции; или «нежесткими», которые могут пересматриваться или смягчаться. Например, последовательное выполнение задач конкретным исполнителем (можно перепланировать на другого исполнителя) или разработка базового ПО, которая должна предшествовать разработке прикладного ПО. В этом случае можно создавать «заглушки» эмулирующие работу базового ПО. Таким образом, диаграмма Ганта для расписания проекта выглядит как гамак, составленный из множества цепочек взаимосвязанных работ с единой точкой начала и завершения.

Критический путь проекта (Critical path)- самая длинная цепочка работ в проекте. Увеличение длительности любой работы в этой цепочки приводит к увеличению длительности всего проекта.

В проекте всегда существует хотя бы один критический путь, но их может быть несколько. Критический путь может меняться во время исполнения проекта. При исполнении проекта руководитель должен обращать внимание на исполнение задач на критическом пути в первую очередь и следить за появлением других критических путей.

Практическая рекомендация: на критическом пути должны стоять работы с нежесткими связями, которые всегда можно перепланировать, если возникает угроза срыва сроков.

Выводы

На верхнем уровне ИСР должны находиться не процессы, а продукты проекта, на следующем уровне - компоненты из которых эти продукты состоят. Выделение компонентов, составляющих программный продукт, это элемент высокоуровневого проектирования, которое мы должны выполнить на фазе планирования проекта, не дожидаясь проработки всех функциональных требований к разрабатываемому ПО.

Помимо работ, непосредственно направленных на создание программного обеспечения, в плане проекта должны быть предусмотрены необходимые ресурсы для обеспечения работ по следующим процессам:

- □ управление содержанием;
- □ управление конфигурациями,
- □ управление качеством,
- □ управление рисками,
- □ управление проектом.

В проекте всегда существует хотя бы один критический путь, но их может быть несколько. Критический путь может меняться во время исполнения проекта. При исполнении проекта руководитель должен обращать внимание на исполнение задач на критическом пути в первую очередь и следить за появлением других критических путей

Тема 6 Стандарты и стандартизация в области разработки ПО

Список стандартов, использованных в процессе выполнения ВКР, зависит от характера проектируемого объекта. Для проектирования ПС в области защиты информации в общем случае могут использоваться стандарты процессов создания ПО, стандарты, регламентирующие процессы защиты информации, стандарты ЕСТД.

Следует исходить из положений закона РФ № 184-ФЗ от 15.12.2002 “О техническом регулировании”. Рассматриваются организация систем отечественной и международной стандартизации, их структуры. Стандарты ГОСТ и ГОСТ Р. в области информационных технологий. Стандарты ISO в области информационных технологий. Информационные источники по стандартам. Изучается организация работ по стандартизации на международном, государственном уровнях, на уровнях отрасли и предприятия заказчика и разработчика ПО. Системы 19, 34, 35 государственных стандартов. Система стандартов ИСО/МЭК. Нормативнометодическое обеспечение процессов создания ПО.

Стандартизация связана с такими понятиями, как объект стандартизации и область стандартизации. Объектом стандартизации обычно называют продукцию, процесс, услугу, для которых разрабатывают те или иные требования, характеристики, параметры, правила и т.п. Стандартизация может касаться либо объекта в целом, либо его отдельных составляющих (характеристик). Областью стандартизации называют совокупность взаимосвязанных объектов стандартизации.

Стандартизация осуществляется на разных уровнях. Уровень стандартизации зависит от того, участники какого географического, экономического, политического региона мира принимают стандарт. Если участие в стандартизации открыто для соответствующих органов любой страны, то это международная стандартизация. Региональная стандартизация — деятельность, открытая только для соответствующих органов государств одного географического, политического или экономического региона. Региональная и международная стандартизация осуществляется специалистами стран, представленных в соответствующих региональных и международных организациях.

Национальная стандартизация — стандартизация в одном конкретном государстве. При этом национальная стандартизация также может осуществляться на разных уровнях: на государственном, отраслевом, в том или ином секторе экономики (например, на уровне

министерств), на уровне ассоциаций, производственных фирм, предприятий (фабрик, заводов) и учреждений.

Рисунок 6.1 – Уровни стандартизации

Стандартизацию, которая проводится в административно территориальной единице (провинции, крае и т.п.), принято называть административно-территориальной стандартизацией. Стандарт нормативный документ по стандартизации, разработанный, как правило, на основе согласия, характеризующегося отсутствием возражений по существенным вопросам у большинства заинтересованных сторон, принятый (утвержденный) признанным органом (предприятием). (ГОСТ Р 1.0-92. Государственная система стандартизации РФ. Основные положения.)

Стандарт может быть разработан как на материальные предметы (продукцию, эталоны, образцы веществ), так и на процессы, нормы, правила, требования различного характера.

Стандарты бывают международными, региональными, национальными, административно-территориальными. Они принимаются соответственно международными, региональными, национальными, территориальными органами по стандартизации. Все эти категории стандартов предназначены для широкого круга потребителей. По существующим нормам стандартизации стандарты периодически пересматриваются для внесения изменений, чтобы их требования соответствовали уровню научно-технического прогресса, или, согласно терминологии ИСО/МЭК, стандарты должны представлять собой «признанные технические правила». Нормативный документ, в том числе и стандарт, считается признанным техническим правилом, если он разработан в сотрудничестве с заинтересованными сторонами путем консультаций и на основе консенсуса.

Указанные выше категории стандартов называют общедоступными. Другие же категории стандартов, такие, как фирменные или отраслевые, не являясь таковыми, могут, однако, использоваться и в нескольких странах согласно существующим там правовым нормам. Документ “технические условия” (ТУ) устанавливает технические требования к продукции, услуге, процессу. Обычно в документе технических условий должны быть указаны методы или процедуры, которые следует использовать для проверки соблюдения требований данного нормативного документа в таких ситуациях, когда это необходимо.

Свод правил, как и предыдущий нормативный документ, может быть самостоятельным стандартом либо самостоятельным документом, а также частью стандарта. Свод правил обычно разрабатывается для процессов проектирования, монтажа оборудования и конструкций, технического обслуживания или эксплуатации объектов, конструкций, изделий. Технические правила, содержащиеся в документе, носят рекомендательный характер.

Все указанные выше нормативные документы являются рекомендательными. В отличие от них регламент имеет обязательный характер. Регламент — это документ, в котором содержатся обязательные правовые нормы.

Государственные стандарты разрабатывают на продукцию, работы и услуги, потребности в которых носят межотраслевой характер. Отраслевые стандарты разрабатываются применительно к продукции определенной отрасли. Их требования не

должны противоречить обязательным требованиям государственных стандартов, а также правилам и нормам безопасности, установленным для отрасли. Принимают такие стандарты государственные органы управления (например, министерства), которые несут ответственность за соответствие требований отраслевых стандартов обязательным требованиям ГОСТ Р. Диапазон применяемости отраслевых стандартов ограничивается предприятиями, подведомственными государственному органу управления, принявшему данный стандарт. На добровольной основе возможно использование этих стандартов субъектами хозяйственной деятельности иного подчинения. Степень обязательности соблюдения требований стандарта отрасли определяется тем предприятием, которое применяет его, или по договору между изготовителем и потребителем. Контроль за выполнением обязательных требований организует ведомство, принявшее данный стандарт.

Стандарты предприятий разрабатываются и принимаются самими предприятиями. Объектами стандартизации в этом случае обычно являются составляющие подсистем организации и управления производством, совершенствование которых — главная цель стандартизации на данном уровне. Кроме того, стандартизация на предприятии может затрагивать и продукцию, производимую этим предприятием. Тогда объектами стандарта предприятия будут составные части продукции, технологическая оснастка и инструменты, общие технологические нормы процесса производства этой продукции. Стандарты предприятий могут содержать требования к различного рода услугам внутреннего характера. Технические условия разрабатывают предприятия и другие субъекты хозяйственной деятельности в том случае, когда стандарт создавать нецелесообразно. Объектом ТУ может быть продукция разовой поставки, выпускаемая малыми партиями и т.п. Необходимость стандартизации разработки программного обеспечения наиболее удачно описана во введении в стандарт ИСО/МЭК 12207: «Программное обеспечение является неотъемлемой частью информационных технологий и традиционных систем, таких, как транспортные, военные, медицинские и финансовые.

Имеется множество разнообразных стандартов, процедур, методов, инструментальных средств и типов операционной среды для разработки и управления программным обеспечением. Это разнообразие создает трудности при проектировании и управлении программным обеспечением, особенно при объединении программных продуктов и сервисных программ. Стратегия разработки программного обеспечения требует перехода от этого множества к общему порядку, который позволит специалистам, практикующимся в программном обеспечении, «говорить на одном языке» при разработке и управлении программным обеспечением».

Многообразие стандартов, действующих в сфере ИТ можно отразить классификационной схемой, представленной на Рисунок 6.2.

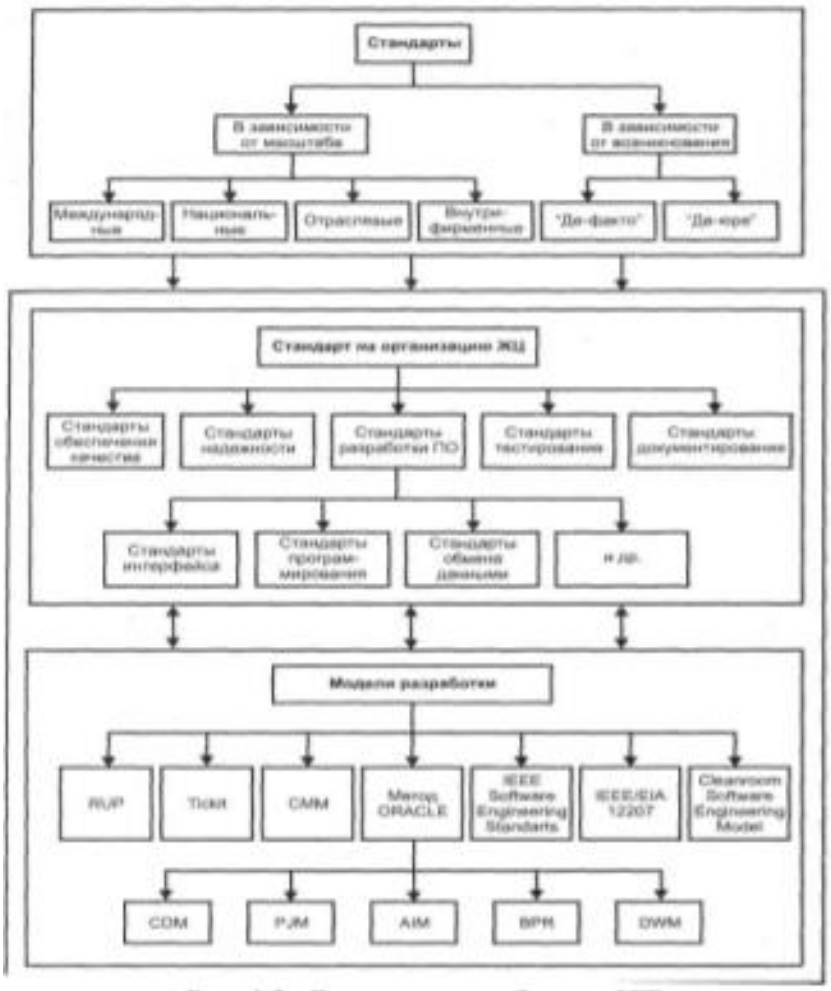

Рисунок 6.2 – Стандарты в области ИТ

Как видно, верхняя часть классификации напоминает указанные выше виды стандартов. В то же время, в практике работы индустрии ПО существует множество стандартов “де-факто”. Стандарт “де-факто” — термин, обозначающий продукт, процесс или услугу какого-либо поставщика, который нашел широкое признание и который другие поставщики стремятся копировать или использовать.

Одна из главных причин значимости современной программы стандартизации — осознание опасности злоупотребления стандартами «дефакто». В 60-е и 70-е годы XX века создание стандартов «де-факто» ставило пользователей в зависимое от производителей положение при использовании основных средств обработки данных и телекоммуникаций. Важный аспект сегодняшней работы по стандартизации — преодоление этой зависимости. Стандарт «де-юре» создается формально признанной стандартизующей организацией. Он разрабатывается при соблюдении правил консенсуса в процессе открытой дискуссии, в которой каждый имеет шанс принять участие. Ни одна группа не может действовать независимо, создавая стандарты для промышленности. Если какая-либо группа поставщиков создаст стандарт, не учитывающий требования пользователей, она потерпит неудачу. То же самое происходит, если пользователи создают стандарт, с которым не могут или не будут соглашаться поставщики, — этот стандарт также не будет успешным. Стандарты «де-юре» не могут быть изменены, не пройдя процесс согласования под контролем организации, разрабатывающей стандарты.

Применение стандарта способствует улучшению качества продукции, повышению уровня унификации и взаимозаменяемости, развитию автоматизации производственных процессов, росту эффективности эксплуатации и ремонта.

Нормативной базой НМО являются международные и отечественные стандарты в области информационных технологий и прежде всего:

• международные стандарты ISO/IEC (ИСО/МЭК) (ISO - International Organization of Standardization

- Международная организация по стандартизации,

IEC — International Electrotechnical Commission — Международная комиссия по электротехнике);

• стандарты ГОСТ Республики Беларусь;

• стандарты организации.

Стандарты в разработке ПС важны по целому ряду причин. Основными из них являются:

1. Стандарты аккумулируют все лучшее из практической деятельности создания ПС и позволяют избежать повторения прошлых ошибок.

2. Стандарты предоставляют необходимую основу для процесса обеспечения качества: достаточно контролировать соблюдение стандартов.

3. Стандарты позволяют упорядочить процесс разработки, что делает разработку прозрачной и снижает затраты на обучение профессиональной деятельности при ротации кадров.

При современном уровне сложности программных систем и в условиях рыночной конкуренции представляется актуальным задача создания технологии коллективной разработки программных средств (ПС), которая будет отражать реалии процесса разработки и обеспечивать рост уровня производства при соответствующем качестве создаваемых программных изделий (ПИ). Из-за разнообразия типов ПС и способов их разработки технология должна обеспечить механизмы собственной адаптации и автоматизации. Актуальными являются и аспекты экономической эффективности и правовой защиты технологии при ее использовании для создания реальных процессов разработки ПС.

Многие современные исследователи в области процессов разработки программных средств рассматривают процедуру создания программ как сущность, включающую три тесно связанный компоненты: процесс (организацию разработки), коллектив разработчиков и программное изделие. Приводимые в статьях и литературе описания процедуры разработки ПИ подробно рассматривают ее организацию (процесс), игнорируя при этом детальное описание других компонент. Те модели компонент технологии разработки ПС, которые можно встретить в современной литературе, носят в основном описательный, а не формальный характер.

Тема 7 Продвижение высокотехнологичной продукции на деловой и потребительские рынки

Маркетинг высокотехнологичной продукции подразумевает использование специфических маркетинговых инструментов, так как применение стандартных инструментов крайне ограничено. Это связано с высоким риском при закупке высокотехнологичной продукции, специфическими особенностями рынка и специфическим назначением товаров.

Основными отличиями продвижения высокотехнологичной продукции являются:

– Важность индивидуального подхода к каждому потенциальному покупателю ввиду их ограниченности;

– Учет уровня инновационного потенциала потребителя;

– Лица, принимающие решение о покупке продукта, часто специалисты в отрасли, что значительно усложняет процесс ведения переговоров и заключения договора; – Высокие финансовые, функциональные, политические и прочие риски, которые усложняют процесс

поставки высокотехнологичной продукции;

– Малоэластичный рынок, на котором преобладает неценовая конкуренция.

Следует отметить, что процесс продвижения продукции на зарубежный рынок имеет незначительные отличия от процесса продвижения продукции на внутреннем рынке. Эти отличия связаны с необходимостью тщательного изучения специфики страны-импортера. Первостепенным шагом является изучение законодательства в области импорта высокотехнологичной продукции (а также в области конкретного сегмента продукции), а также оценка осуществимости вхождения на рынок и оценка издержек, которые компания понесет. Если вхождение на рынок осуществимо, и издержки не превышают потенциальную выручку, далее экспортеру следует задуматься об инструментах продвижения своей продукции на зарубежный рынок. На рисунке 7.1 представлен перечень инструментов, которые могут быть использованы при продвижении высокотехнологичной продукции на зарубежные рынки.

Рисунок 7.1 - Перечень инструментов продвижения высокотехнологичной продукции на зарубежные рынки

Реклама является наиболее популярным и простым способом продвижения продукции. Однако, с точки зрения высокотехнологичных товаров, считается, что реклама не является эффективной: при продажах высокотехнологичной продукции реклама в меньшей степени влияет на узнаваемость товара, по сравнению с личными продажами, стимулированием сбыта или PR-кампаниями. Однако она все же может быть применима при условии выбора специфических рекламных средств [3]. Реклама в специализированных журналах, каталогах, рекламных проспектах. Удачным вариантом для высокотехнологичной продукции потребительского назначения станут, например, журналы для пассажиров рейсов, следующих в страны, где данные продукты реализуются или планируются к реализации. Более того, можно разместить бутик с продукцией в аэропортах, чтобы заинтересованные люди могли сразу ознакомиться с продуктом или приобрести его по прилете. Рекомендуется использовать как имиджевую, так и продуктовую рекламу, или их комбинацию.

Участие в развлекательных массовых мероприятиях (Event marketing). Речь идет о крупных развлекательных мероприятиях, где компании могут стать спонсором, организатором, видео-оператором мероприятия и т.д.

Партнерские программы с потребителями. Здесь подразумеваются бонусные

программы, программы лояльности и прочие программы, которые помогут привлечь и сохранить потребителей. Например, китайская компания по производству беспилотных летательных аппаратов разработала бонусную программу для своих клиентов. Суть программы заключается в том, что потребители оставляют отзывы и рекламируют продукцию производителя в своих социальных сетях, оставляя ссылку на покупку. Программа гарантирует кешбэк с каждой проданной по ссылке единицы товара. Для постоянных пользователей и лояльных клиентов компании это хорошее поощрение за рекомендации и рекламу продукта.

Использование государственной поддержки. Учитывая специфику предложенных методов и высокие издержки на их реализацию, возникает необходимость использования государственной поддержки экспорта высокотехнологичной продукции. Компании-производители, изъявившие желание реализовать политику продвижения, могут воспользоваться услугами государственных структур по продвижению экспортной продукции. Например, в России сформирован Российский экспортный центр - государственный институт поддержки несырьевого экспорта, предоставляющий российским экспортерам широкий спектр финансовых и нефинансовых мер поддержки. В список его услуг входит множество инструментов для поддержки экспорта, в том числе аналитика и исследования, сертификация, патентование, страховани экспортных поставок, помощь при продвижении продукции на внешние рынки (поиск покупателей, помощь при переводе, разработка полиграфических материалов и т.д.).

Сервисное обслуживание. Производители, предоставляющие потребителям сервисное обслуживание после приобретения продукции, чаще выигрывают на рынке. Это дает им неценовое преимущество, а также увеличивает шансы, что сегодняшние потребители перейдут на их продукцию из-за предоставления послепродажного обслуживания.

Кол-центры. Если предоставление сервисного обслуживания за рубежом не является возможным, его могут заменить сервисные кол-центры, операторы которых будут оказывать сервисную поддержку онлайн по телефону или видеосвязи. Это не заменит полноценное сервисное обслуживание, но сэкономит время и деньги клиентов при несерьезных поломках или прочих вопросах, решаемых дистанционно.

Локализация производства. Локализация производства не только поможет эффективней проводить маркетинговую политику в конкретной стране, но и позволит снизить себестоимость продукта. Таким образом, экспортеры могут достичь две цели одновременно: выиграть на рынке за счет снижения себестоимости, и более точно сформировать политику продвижения продукта в зависимости от культурных, политических, экономических, экологических и других особенностей потенциальных покупателей [5].

Организация личных встреч. Для создания имиджа компании и повышения узнаваемости представители организаций могут организовывать выездные мероприятия для знакомства с потенциальными партнерами. Сюда можно включить переговоры с потенциальными клиентами, партнерамипроизводителями, дилерами и прочими потенциально заинтересованными лицами. Отмечается, что продвижение сложной высокотехнологичной продукции тяготеет к использованию личных продаж, поскольку только через личную беседу можно максимально подробно рассказать о высокотехнологичном продукте.

Кооперация с партнерами и конкурентами. Членство с партнерами на международных мероприятиях или приглашение партнеров на мероприятия в своей стране послужат хорошим подходом к продвижению как конкретной продукции, так и бренда в целом.

Trade In. Стандартно это услуга по приему старого автомобиля в счет покупки нового, цена которого уменьшается на стоимость старого, принимаемого в зачет. Такую же схему можно использовать и при продажах высокотехнологичной продукции.

Тест-драйв. Если существует возможность, компании могут предоставлять

потенциальным покупателям услугу тест-драйва, чтобы в течение определенного времени опробовать продукцию и принять решение о покупке. Этот шаг является хорошим инструментом продвижения, учитывая специфику высокотехнологичных товаров и важность их функциональных характеристик для потребителей.

Вышеназванные инструменты могут быть применимы при реализации продукции на зарубежный рынок. Не все из них являются истинно маркетинговыми, однако их применение влияет на сбыт и спрос не меньше, чем классические методы продвижения.

Варианты продвижения программного обеспечения приводятся ниже:

загрузка в магазины приложений;

- создание качественного сайта продукта;
- регистрация на специализированных порталах;
- запуск краудфандинга;
- поисковая оптимизация сайта;
- контекстная реклама для быстрого привлечения трафика;
- ведение экспертного блога;
- продвижение в актуальных соцсетях;
- создание лендингов под разные целевые аудитории;
- работа над юзабилити сайта;
- привлечение внимания на сайте;
- гостевые публикации и пиар;
- запуск рассылок;
- медийная реклама;
- создание и продвижение видео;
- предложение демоверсии;
- запуск партнерской программы;
- прямые продажи для B2B;
- проведение мероприятий онлайн и офлайн;
- ранний доступ для лидеров мнений;
- укрепление позиций бренда;
- локализация программы и маркетинговых материалов.

Тема 8 Прогнозирование

1. Система методов прогнозирования и планирования

Методы прогнозирования и планирования выражаются в способах и приемах разработки прогнозных и плановых документов и показателей применительно к различным их видам и назначениям. Различия в характере прогнозируемых объектов, а также в сроках прогнозирования, степени полноты и достоверности исходных данных предопределяют использование различных методов прогнозирования. Специфика методов отражается на последовательности и содержании работ по составлению прогноза.

Современная прогностика располагает большим арсеналом методов прогнозирования (более 150), но ни один из них не может быть признан универсальным.

На выбор соответствующего метода прогнозирования (планирования) влияют следующие факторы:

– требуемая форма прогноза. При прогнозировании проводится оценка ожидаемых значений показателей на будущее, а также оценка вариации ошибки прогнозирования или промежутка, на котором сохраняется вероятность предсказания реальных будущих значений показателей. Этот промежуток называется предсказуемым интервалом. Однако в некоторых случаях не так важно предсказание конкретных значений прогнозируемой переменной, как предсказание изменений в ее поведении;

– период и горизонт прогнозирования. Период прогнозирования – это основная единица времени, на которую делается прогноз. Горизонт прогнозирования – это число периодов в будущем, которые охватывает

прогноз; – доступность данных;

– требуемая точность;

– поведение прогнозируемого процесса;

– бюджетные ограничения;

– сложность исследуемой социально-экономической системы;

– предпочтения руководителей и др.

Логическая последовательность разработки прогноза предусматривает выполнение следующих этапов:

1. Предпрогнозная ориентация на основе системно-структурного анализа объекта прогнозирования.

2. Постановка задачи для разработки прогноза.

3. Анализ и установление активных факторов прогнозного фона.

4. Формирование информационной базы по объекту прогноза и прогнозному фону.

5. Составление прогнозной модели и выбор методов прогнозирования.

6. Разработка прогноза развития объекта и оценка его достоверности с учетом действия факторов прогнозного фона.

7. Формулирование рекомендаций по принятию плановых решений на основе прогноза.

Основой методики прогнозирования и планирования являются проведение аналитического исследования, подготовка базы данных, изучение и соединение информации в единое целое. Будущее во многом становится предсказуемым, если правильно и полно учитываются сложившаяся ситуация, факторы и тенденции, способствующие ее изменению в перспективе. Поэтому неотъемлемыми составляющими процессов прогнозирования и планирования являются такие общенаучные методы, как анализ и синтез, индукция и дедукция, аналогия и др.

Всю совокупность методов прогнозирования можно распределить по степени их однородности на группы простых и комплексных методов. Группа простых методов объединяет однородные по содержанию и используемому инструментарию (например, морфологический анализ, экстраполяция и т. д.) методы прогнозирования. Комплексные методы отражают совокупность, комбинации методов, чаще всего реализуемые специальными прогностическими системами (например, метод прогнозного графа, система Паттерн и т. д.).

В зависимости от характера информации, на базе которой составляется прогноз, все методы прогнозирования делят на классы: фактограф фактографические, экспертные и комбинированные.

Фактографические методы базируются на фактическом информационном материале о прошлом и настоящем развитии объекта прогнозирования. Чаще всего применяются в поисковом прогнозировании для эволюционных процессов. Фактографические методы прогнозирования привлекательны своей относительной простотой и объективностью. Однако при появлении непредвиденных ограничений, сдерживающих процесс развития, использование этих методов может привести к ошибкам в прогнозах. Следует учитывать, что они применимы, когда:

– вероятность сохранения факторов, обусловивших процесс развития в прошлом, больше, чем вероятность их изменения;

– вероятность совокупного влияния всех этих факторов на развитие в прежнем направлении больше, чем вероятность его изменения.

Надежность и точность фактографических методов может быть увеличена за счет сочетания их с экспертными методами. Экспертные методы основаны на использовании знаний специалистов-экспертов об объекте прогнозирования и обобщении их мнений о

развитии (поведении) объекта в будущем. Такие методы в большей мере соответствуют нормативному прогнозированию динамичных процессов развития.

Комбинированные методы включают в себя методы со смешанной информационной основой, в которых в качестве первичной информации используется, наряду с экспертной, и фактографическая. К ним можно отнести и такие методы макроэкономического планирования, как балансовый, нормативный и программно-целевой.

По степени формализации методы экономического прогнозирования можно подразделить на интуитивные и формализованные.

Интуитивные (эвристические) методы базируются на интуитивно-логическом мышлении. Они используются в тех случаях, когда невозможно учесть влияние многих факторов из-за значительной сложности объекта прогнозирования или объект слишком прост и не требует проведения трудоемких расчетов.

К интуитивным методам относятся методы экспертных оценок, исторических аналогий, прогнозирования по образцу. Формализованные методы основаны на проведении математического анализа тенденций развития экономической системы и выявлении факторов, оказывающих наибольшее влияние на изменение условий хозяйствования. Они базируются на математической теории. Обобщенная классификация методов прогнозирования и планирования представлена на рисунке 8.1.

Рисунок 8.1 - Классификация методов прогнозирования и планирования

2. Индивидуальные методы экспертных оценок

В системе методов прогнозирования значительное место занимают методы экспертных оценок. Метод экспертной оценки базируется на рациональных доводах и интуиции высококвалифицированных специалистов (экспертов), обработке их информации о прогнозируемом объекте.

Основная идея прогнозирования на основе экспертных оценок заключается в построении рациональной процедуры интуитивно-логического мышления человека в сочетании с количественными методами оценки и обработки получаемых результатов.

Экспертиза может быть индивидуальной и коллективной и выражаться в форме докладной записки, сценария, интервью и т. д.

Среди индивидуальных экспертных оценок наиболее широкое распространение

получили метод «интервью», аналитический метод, метод написания сценария. Последний также относится и к коллективным методам экспертных оценок. Метод «интервью» предполагает беседу прогнозиста с экспертом по схеме «вопрос – ответ», в процессе которой прогнозист в соответствии с заранее разработанной программой ставит перед экспертом вопросы относительно перспектив развития прогнозируемого объекта. Эксперт дает заключение экспромтом.

Аналитический метод предусматривает тщательную самостоятельную работу эксперта над анализом тенденций, оценкой состояния и путей развития прогнозируемого объекта. Эксперт может использовать всю необходимую ему информацию об объекте прогноза. Свои выводы эксперт оформляет в виде докладной записки.

Метод написания сценария основан на определении логики процесса или явления во времени при различных условиях. Он предполагает установление последовательности событий, развивающихся при переходе от существующей ситуации к будущему состоянию объекта. Прогнозный сценарий определяет стратегию развития прогнозируемого объекта. Он должен отражать генеральную цель развития объекта, критерии для оценки верхних уровней «дерева целей», приоритеты проблем и ресурсы для достижения основных целей. В сценарии отображаются последовательное решение задачи, возможные препятствия.

Сценарий обычно носит многовариантный характер и рассматривает три линии поведения: оптимистическую – развитие системы в наиболее благоприятной ситуации; пессимистическую – развитие системы в наименее благоприятной ситуации; рабочую – развитие системы с учетом противодействия отрицательным факторам, появление которых наиболее вероятно. В рамках прогнозного сценария целесообразно прорабатывать резервную стратегию на случай непредвиденных ситуаций.

3. Коллективные методы экспертных оценок

Наиболее достоверными являются коллективные экспертные оценки, предполагающие работу двух и более экспертов.

Для организации проведения экспертных оценок создаются рабочие группы, в функции которых входят проведение опроса, обработка материалов и анализ результатов коллективной экспертной оценки. Рабочая группа назначает экспертов, которые дают ответы на поставленные вопросы, касающиеся перспектив развития данного объекта. Количество экспертов, привлекаемых для разработки прогноза, может колебаться от 10 до 150 человек в зависимости от сложности объекта. Определяется цель прогноза, разрабатываются вопросы для экспертов. После опроса осуществляется обработка материалов коллективной экспертной оценки.

Среди методов коллективных экспертных оценок наиболее широкое применение нашли метод коллективной генерации идей, метод «635», метод «Дельфи», метод «комиссий», метод написания сценария.

Сущность метода коллективной генерации идей (мозговой атаки) состоит в использовании творческого потенциала специалистов при «мозговой атаке» проблемной ситуации, реализующей вначале генерацию идей и последующее деструктурирование (разрушение, критику) этих идей с формулированием контридей и выработкой согласованной точки зрения.

Метод коллективной генерации идей предполагает реализацию следующих этапов.

Первый этап связан с формированием группы участников «мозговой атаки» (по численности и составу) по решению определенной проблемы. Оптимальная численность группы участников находится эмпирическим путем.

На втором этапе составляется проблемная записка. Она формируется группой анализа проблемной ситуации и включает описание метода и проблемной ситуации.

Третий этап – этап генерации идей. Каждый из участников имеет право выступать много раз. Критика предыдущих выступлений и скептические замечания не допускаются. Ведущий корректирует процесс, приветствует усовершенствование или комбинацию идей, оказывает поддержку, освобождая участников от скованности.

Четвертый этап связан с систематизацией идей, высказанных на этапе генерации. Формируется перечень идей, выделяются признаки, по которым идеи могут быть объединены, идеи объединяются в группы согласно выделенным признакам.

На пятом этапе осуществляется деструктурирование (разрушение) систематизированных идей. Каждая идея подвергается всесторонней критике со стороны группы высококвалифицированных специалистов.

На шестом этапе дается оценка критических замечаний и составляется список практически реализуемых идей.

Метод «635» – это одна из разновидностей «мозговой атаки». Цифры 6, 3, 5 обозначают следующее: 6 участников, каждый из которых должен записать 3 идеи в течение каждых 5 мин. Лист ходит по кругу в течение получаса. В результате каждый эксперт запишет в свой актив по 18 идей, а в общей сложности будет получено 108 идей.

Сущность метода «Дельфи» заключается в том, что опрос экспертов проводится в несколько туров (с помощью специальных анкет). Каждый тур уточняет предыдущие ответы и постепенно приводит экспертов к согласованному решению. Оценки экспертов основаны как на строго логическом анализе, так и на интуиции и опыте. Этот метод не требует личного общения экспертов и заменяет прямые дебаты тщательно разработанной программой последовательных индивидуальных опросов. В результате происходит сужение диапазона оценок и вырабатывается согласованное суждение.

Метод «комиссий» – это метод экспертных оценок, основанный на работе специальных комиссий. Группы экспертов за «круглым столом» обсуждают ту или иную проблему с целью согласования точек зрения и выработки единого мнения. Недостаток этого метода заключается в том, что группа экспертов в своих суждениях руководствуется в основном логико

Тема 9 Управление рисками

1 Основные понятия

Согласно Том Демарко: «Проект без риска - удел неудачников. Риски и выгода всегда ходят рука об руку». В первой лекции мы уже говорили о том, что, в силу специфики отрасли, программная инженерия остается и, в ближайшем будущем, будет оставаться производством с высоким уровнем рисков. Если задуматься, то все, что мы делаем, управляя проектом разработки ПО, направлено на борьбу с рисками не уложиться в срок, перерасходовать ресурсы, разработать не тот продукт, который требуется. Определение риска было дано в предыдущей лекции.

Риск - это проблема, которая еще не возникла, а проблема - это риск, который материализовался. Риск характеризуется следующими характеристиками (рисунок 5.1).

□ Причина или источник. Явление, обстоятельство обусловливающее наступление риска.

□ Симптомы риска, указание на то, что событие риска произошло или вотвот произойдет. Первопричина нам может быть не наблюдаема, например, заразились гриппом. Мы наблюдаем некоторые симптомы - поднялась температура.

□ Последствия риска. Проблема или возможность, которая может реализоваться в проекте в результате произошедшего риска.

□ Влияние риска. Влияние реализовавшегося риска на возможность достижения целей проекта. Воздействие обычно касается стоимости, графика и технических характеристик разрабатываемого продукта. Многие риски происходят частично и оказывают соразмерное отрицательное или положительное воздействие на проект.

Риск это всегда вероятность и последствия. Например, всегда есть вероятность того, что метеорит упадет на офис центра программных разработок, и это будет иметь катастрофические последствия для проекта. Однако вероятность наступления этого события настолько мала, что мы в большинстве проектов принимаем это риск и не пытаемся им управлять.

Майк Ньюэлл, вице-президент компании PSM Consulting, рассказывал, как он объясняет аудитории на своих лекциях, что такое риск. Он предлагает сыграть в кости на таких условиях, если на кубике выпадает шестерка, то выигрывает он. Если - любое другое число, то выигрывает слушатель. Ставка по 1 доллару. Обычно, большая часть аудитории соглашается сыграть на таких условиях. Майк поднимает ставки: $10, $100, $1000. Постепенно количество желающих поиграть становится все меньше и меньше. При ставке $1000, как правило, желающих рисковать не остается.

Принято выделять две категории рисков:

□ «Известные неизвестные». Это те риски, которые можно идентифицировать и подвергнуть анализу. В отношении таких рисков можно спланировать ответные действия.

□ «Неизвестные неизвестные». Риски, которые невозможно идентифицировать и, следовательно, спланировать ответные действия.

Рисунок 5.1 - Пример характеристик риска

Неизвестные риски это непредвиденные обстоятельства. Единственное, что мы можем в этом случае предпринять, это создать управленческий резерв бюджета проекта на случай незапланированных, но потенциально возможных изменений. На расходование этого резерва менеджер проекта, как правило, обязан получать одобрение вышестоящего руководства. Управленческие резервы на непредвиденные обстоятельства не входят в базовый план по стоимости проекта, но включаются в бюджет проекта. Они не распределяются по проекту, как бюджет, и поэтому не учитываются при расчете освоенного объема.

Девиз разработчиков ПО из Microsoft:«Mbi не боремся с рисками — мы ими управляем». Цели управления рисками проекта - снижение вероятности возникновения и/или значимости воздействия неблагоприятных для проекта событий. Адекватное управление рисками в компании - признак зрелости производственных процессов. Том Демарко пишет: «Рассматривать только благоприятные сценарии и встраивать их в план проекта - настоящее ребячество. И все же мы постоянно так поступаем. ...Если тех, кто говорит о возможных проблемах до открытия проекта, называют troublemakers, а тех, кто сдает проект спустя 2 месяца после обещанного срока, работая при этом по 6080 часов в неделю, - героями, то у вас плохая команда».

Отказываться от управления проектными рисками это все равно, что в кинотеатре не иметь огнетушителей и плана эвакуации на случай пожара.

2 Планирование управления рисками

Управление рисками это определенная деятельность, которая выполняется в проекте

от его начала до завершения. Как и любая другая работа в проекте управление рисками требует времени и затрат ресурсов. Поэтому эта работа обязательно должна планироваться. Планирование управления рисками - это процесс определения подходов и планирования операций по управлению рисками проекта. Тщательное и подробное планирование управления рисками позволяет:

□ выделить достаточное количество времени и ресурсов для выполнения операций по управлению рисками,

□ определить общие основания для оценки рисков,

□ повысить вероятность успешного достижения результатов проекта.

Планирование управления рисками должен быть завершено на ранней стадии планирования проекта, поскольку оно крайне важно для успешного выполнения других процессов.

Исходными данными для планирования управления рисками служат:

□ Отношение к риску и толерантность к риску организаций и лиц, участвующих в проекте, оказывает влияние на план управления проектом. Оно должно быть зафиксировано в изложении основных принципов и подходов к управлению рисками.

□ Стандарты организации. Организации могут иметь заранее разработанные подходы к управлению рисками, например категории рисков, общие определение понятий и терминов, стандартные шаблоны, схемы распределения ролей и ответственности, а также определенные уровни полномочий для принятия решений.

□ Описание содержания проекта подробно описывает результаты поставки проекта и работы, необходимые для создания этих результатов поставки.

□ План управления проектом, формальный документ, в котором указано, как будет исполняться проект и как будет происходить мониторинг и управление проектом.

План управления рисками обычно включает в себя следующие элементы:

□ Определение подходов, инструментов и источников данных, которые могут использоваться для управления рисками в данном проекте.

□ Распределение ролей и ответственности. Список позиций выполнения, поддержки и управления рисками для каждого вида операций, включенных в план управления рисками, назначение сотрудников на эти позиции и разъяснение их ответственности.

□ Выделение ресурсов и оценка стоимости мероприятий, необходимых для управления рисками. Эти данные включаются в базовый план по стоимости проекта.

□ Определение сроков и частоты выполнения процесса управления рисками на протяжении всего жизненного цикла проекта, а также определение операций по управлению рисками, которые необходимо включить в расписание проекта.

□ Категории рисков. Структура, на основании которой производится систематическая и всесторонняя идентификация рисков с нужной степенью детализации. Такую структуру можно разработать с помощью составления иерархической структуры рисков (рисунок 5.2).

□ Общие подходы для определения уровней вероятности, шкалы воздействия и близости рисков на проект.

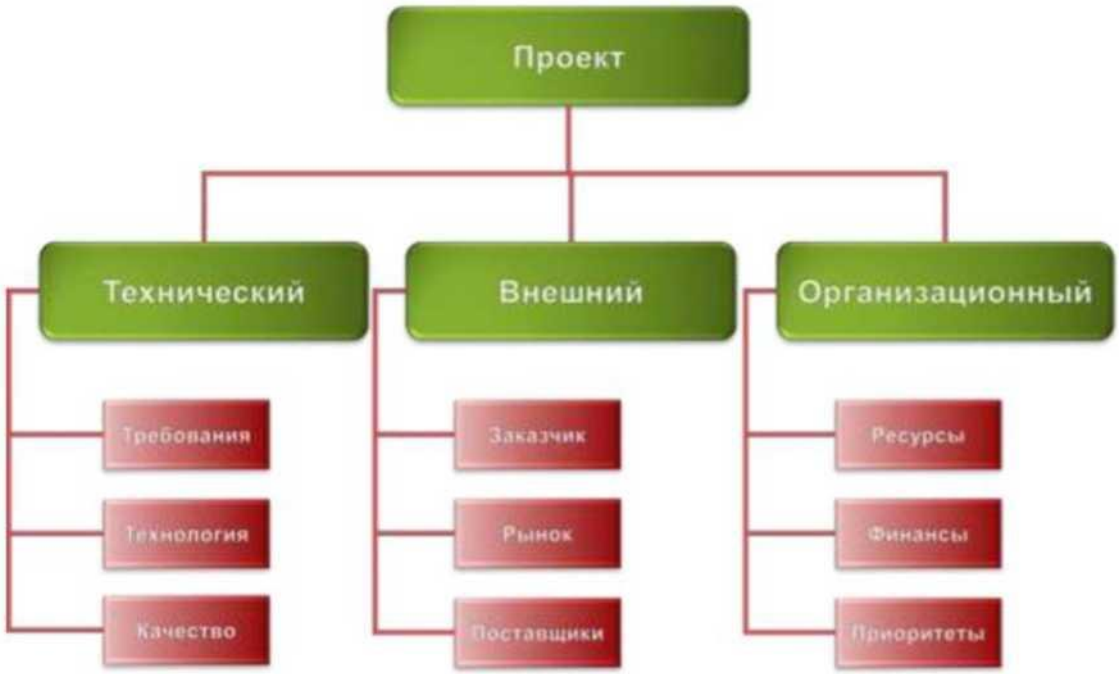

Рисунок 5.2 - Пример иерархической структуры рисков проекта

Шкала оценки воздействия отражает значимость риска (таблица 5.1) в случае его возникновения. Шкала оценки воздействия может различаться в зависимости от потенциально затронутой риском цели, типа и размера проекта, принятыми в организации стратегиями и его финансовым состоянием, а также от чувствительности организации к конкретному виду воздействий.

Таблица 5.1 - Пример шкалы оценки воздействия рисков

Вес	**Значение**	**Критерий**
3	Катастрофические	Потери более $100K
2	Критичные	Потери от $10K до $100K
1	Умеренные	Потери менее $10K

Хотя риск может воздействовать и на сроки проекта, и на качество получаемого продукта, но все эти отклонения могут быть оценены в денежном эквиваленте. Например, последствия задержка по срокам для заказной разработки может быть выражена в сумме денежных санкций, определенных в контракте.

Похожая шкала может быть применена для оценки вероятности наступления риска (таблица 5.2).

Таблица 5.2 - Пример шкалы оценки вероятности осуществления риска

Вес	**Значение**	**Критерий**
3	Очень вероятно	Шансы наступления весьма велики
2	Возможно	Шансы равны
1	Мало вероятно	Наступление события весьма сомнительно

Еще одной важной характеристикой риска является близость его наступления. Естественно, что при прочих равных условиях рискам, которые могут осуществиться уже завтра, следует сегодня уделять больше внимания, чем тем, которые могут произойти не ранее, чем через полгода. Для шкалы оценки близости риска может быть применена, например, следующая градация: очень скоро, не очень скоро, очень нескоро.

3 Идентификация рисков

Идентификация рисков - это выявление рисков, способных повлиять на проект, и

документальное оформление их характеристик. Это итеративный процесс, который периодически повторяется на всем протяжении проекта, поскольку в рамках его жизненного цикла могут обнаруживаться новые риски.

Исходные данные для выявления и описания характеристик рисков могут браться из разных источников.

В первую очередь это база знаний организации. Информация о выполнении прежних проектов может быть доступна в архивах предыдущих проектов. Следует помнить, что проблемы завершенных и выполняемых проектов, это, как правило, риски в новых проектах.

Другим источником данных о рисках проекта может служить разнообразная информация из открытых источников, научных работ, маркетинговая аналитика и другие исследовательские работы в данной области. Наконец, многие форумы по программированию могут дать бесценную информацию о возникших ранее проблемах в похожих проектах.

Каждый проект задумывается и разрабатывается на основании ряда гипотез, сценариев и допущений. Как правило, в описании содержания проекта перечисляются принятые допущения - факторы, которые для целей планирования считаются верными, реальными или определенными без привлечения доказательств. Неопределенность в допущениях проекта следует также обязательно рассматривать в качестве потенциального источника возникновения рисков проекта. Анализ допущения позволяет идентифицировать риски проекта, происходящие от неточности, несовместимости или неполноты допущений.

Для сбора информации о рисках могут применяться различные подходы. Среди этих подходов наиболее распространены:

- □ Опрос экспертов
- □ Мозговой штурм
- □ Метод Дельфи
- □ Карточки Кроуфорда

Цель опроса экспертов - идентифицировать и оценить риски путем интервью подходящих квалифицированных специалистов. Специалисты высказывают своё мнение о рисках и дают им оценку, исходя из своих знаний, опыта и имеющейся информации. Этот метод может помочь избежать повторного наступления на одни и те же грабли.

Перед опросом эксперт должен получить всю необходимую вводную информацию. Деятельность экспертов необходимо направлять, задавая вопросы. Во время опроса вся информация, выдаваемая экспертом, должна записываться и сохраняться. При работе с несколькими экспертами выходная информация обобщается и доводится до сведения всех задействованных экспертов.

К участию в мозговом штурме привлекаются квалифицированные специалисты, которым дают «домашнее задание» - подготовить свои суждения по определенной категории рисков. Затем проводится общее собрание, на котором специалисты по очереди высказывают свои мнения о рисках. Важно: споры и замечания не допускаются. Все риски записываются, группируются по типам и характеристикам, каждому риску дается определение. Цель - составить первичный перечень возможных рисков для последующего отбора и анализа.

Метод Дельфи во многом похож на метод мозгового штурма. Однако есть важные отличия. Во-первых, при применении этого метода эксперты участвуют в опросе анонимно. Поэтому результат характеризуется меньшей субъективностью, меньшей предвзятостью и меньшим влиянием отдельных экспертов. Во-вторых, опрос экспертов проводится в несколько этапов. На каждом этапе модератор рассылает анкеты, собирает и обрабатывает ответы. Результаты опроса рассылаются экспертам снова для уточнения их мнений и оценок. Такой подход позволяет достичь некоего общего мнения специалистов о рисках.

Для быстрого выявления рисков можно воспользоваться еще одной из методик социометрии является известной как "Карточки Кроуфорда" .

Суть этой методики в следующем. Собирается группа экспертов 7-10 человек. Каждому участнику мини-исследования раздается по десять карточек (для этого вполне

подойдет обычная бумага для записок). Ведущий задает вопрос: "Какой риск является наиболее важным в этом проекте?" Все респонденты должны записать наиболее, по их мнению, важный риск в данном проекте. При этом никакого обмена мнениями не должно быть. Ведущий делает небольшую паузу, после чего вопрос повторяется. Участник не может повторять в ответе один и тот же риск.

После того как вопрос прозвучит десять раз, в распоряжении ведущего появятся от 70 до 100 карточек с ответами. Если группа подобрана хорошо (в том смысле, что в нее входят люди с различными точками зрения), вероятность того, что участники эксперимента укажут большинство значимых для проекта рисков, весьма высока. Остается составить список названных рисков и раздать его участникам для внесения изменений и дополнений.

В качестве источника информации при выявлении рисков могут служить различные доступные контрольные списки рисков проектов разработки ПО, которые следует проанализировать на применимость к данному конкретному проекту.

Например, Барии Боэм приводит список 10 наиболее распространенных рисков программного проекта:

1. Дефицит специалистов.
2. Нереалистичные сроки и бюджет.
3. Реализация несоответствующей функциональности.
4. Разработка неправильного пользовательского интерфейса.
5. “Золотая сервировка”, перфекционизм, ненужная оптимизация и оттачивание деталей.
6. Непрекращающийся поток изменений.
7. Нехватка информации о внешних компонентах, определяющих окружение системы или вовлеченных в интеграцию.
8. Недостатки в работах, выполняемых внешними (по отношению к проекту) ресурсами.
9. Недостаточная производительность получаемой системы.
10. “Разрыв” в квалификации специалистов разных областей знаний.

Демарко и Листер приводят свой список из пяти наиболее важных источников рисков любого проекта разработки ПО:

1. Изъяны календарного планирования
2. Текучесть кадров
3. Раздувание требований
4. Нарушение спецификаций
5. Низкая производительность

Не существует исчерпывающих контрольных списков рисков программного проекта, поэтому необходимо внимательно анализировать особенности каждого конкретного проекта.

Результатом идентификации рисков должен стать список рисков с описанием их основных характеристик: причины, условия, последствий и ущерба.

Если вернуться к примеру проекта создания «Автоматизированной системы продажи документации», который мы рассматривали в предыдущих лекциях, то список главных выявленных рисков может выглядеть следующим образом

Таблица 5.3 - Список рисков проекта создания «Автоматизированной системы продажи документации»

Причина	**Условия**	**Последствия**	**Ущерб**
Требования не ясны.	Отсутствие описания сценариев использования системы.	Задержка начала разработки прикладного ПО. Большой объем переработок.	Задержки в сроках сдачи готового продукта и дополнительные трудозатраты.

Недостаток квалифицированных кадров.	Архитектура и код низкого качества	Большое число ошибок. Большие затраты на их исправление	Задержки в сроках сдачи готового продукта и дополнительные трудозатраты
Текучесть кадров	Частая смена участников команды	Низкая производительность при вводе новых участников в проект	Задержки в сроках сдачи готового продукта и дополнительные трудозатраты

За процессом идентификации рисков следует процесс их качественного анализа.

4 Качественный анализ рисков

Качественный анализ рисков включает в себя расстановку рангов для идентифицированных рисков. При анализе вероятности и влияния предполагается, что никаких мер по предупреждению рисков не производится.

Качественный анализ рисков включает:

- □ Определение вероятности реализации рисков.
- □ Определение тяжести последствий реализации рисков.
- □ Определения ранга риска по матрице «вероятность - последствия».
- □ Определение близости наступления риска.
- □ Оценка качества использованной информации.

Для качественной оценки вероятности реализации риска и определения тяжести последствий его реализации применяется, как правило, общепринятые в организации шкалы, примеры которых мы приводили ранее.

Для определения ранга риска используется матрица вероятностей и последствий (рисунок 5.2). Ранг риска определяется произведением веса вероятности и значимости последствий.

Рисунок 5.2 - Ранг риска и матрица вероятностей и последствий

Могут, конечно, существовать и более сложные шкал для оценок вероятностей, значимости последствий и ранга рисков. Встречались шкалы, которые содержали до 10 градаций. Но, на мой взгляд, наиболее прагматичный подход - это использовать трехуровневое ранжирование.

Продолжая рассмотрение примера проекта создания «Автоматизированной системы продажи документации», матрица рангов главных выявленных рисков может выглядеть следующим образом (таблица 5.4).

Таблица 5.4 - Матрица рангов главных выявленных рисков проекта создания «Автоматизированной системы продажи документации»

Причина	Вероятность	Воздействие	Ранг
Требования не ясны	Очень вероятно	Катастрофические	9
Недостаток квалифицированных кадров	Очень вероятно	Критичные	6
Текучесть кадров	Возможно	Критичные	4

Для оценки рисков необходима точная и адекватная информация. Использование неточной информации ведет к ошибкам в оценке. Неверная оценка риска также является риском.

Критерии оценки качества используемой при анализе информации выглядят следующим образом:

- Степень понимания риска.
- Доступность и полнота информации о риске.
- Надежность, целостность и достоверность источников данных.

Результатом качественного анализа рисков является их подробное описание (таблица 5.5).

Таблица 5.5 - Пример карточки с описанием риска

Номер : R-101	**Категория: Технологический**
Причина: Недостаток квалифицированных кадров.	Симптомы: Разработчики будут использовать новую платформу J2EE
Последствия: Низкая производительность разработки	Воздействие: Увеличение сроков и трудоемкости разработки.
Вероятность: Очень вероятно	Степень воздействия: Критичная.
Близость : Очень скоро.	Ранг: 6.
Исходные данные: «Содержание проекта», «План обеспечения ресурсами», Протоколы совещаний №21 от 01.06.2008, №27 от 25.06.2008.	

Результаты качественного анализа используются в ходе последующего количественного анализа рисков и планирования реагирования на риски.

5 Количественный анализ рисков

Количественный анализ производится в отношении тех рисков, которые в процессе качественного анализа были квалифицированы как имеющие высокий и средний ранг.

Для количественного анализа рисков могут быть использованы следующие методы:

- Анализ чувствительности.
- Анализ дерева решений.
- Моделирование и имитация.

Анализ чувствительности помогает определить, какие риски обладают наибольшим потенциальным влиянием на проект. В процессе анализа устанавливается, в какой степени неопределенность каждого элемента проекта отражается на исследуемой цели проекта, если остальные неопределенные элементы принимают базовые значения. Результаты представляются, как правило, в виде диаграммы «торнадо». На рисунке 5.3 представлен пример диаграммы, которая отражает влияние на проектные трудозатраты различных факторов профессионализма разработчиков ПО.

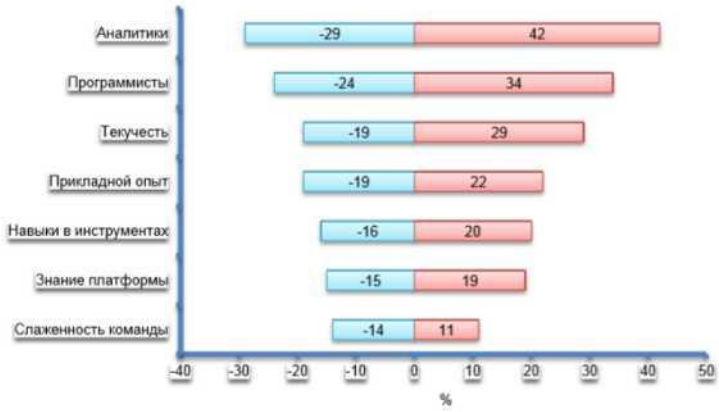

Рисунок 5.3 - Влияние факторов профессионализма разработчиков ПО на трудозатраты по проекту.

Анализ последствий возможных решений проводится на основе изучения диаграммы дерева решений, которая описывает рассматриваемую ситуацию с учетом каждой из имеющихся возможностей выбора и возможного сценария. Рисунок 5.4 представляет пример диаграммы дерева решений на дугах которой проставлены вероятности и затраты при развитии событий по тому или иному сценарию. Критерием для принятия решения служит математическое ожидание потерь от его принятия.

Рисунок 5.4 - Пример анализ дерева решений при выборе покупать или производить необходимую для проекта библиотеку визуальных компонентов (VCL).

При моделировании рисков проекта используется модель для определения последствий от воздействия подробно описанных неопределенностей на результаты проекта в целом. Моделирование обычно проводится с помощью метода Монте-Карло.

Пример подобной модели - система Riskology от Демарко и Листера, который иллюстрирует применение метода Монте-Карло для получения информации о том, какой запас времени будет необходим для того, чтобы преодолеть влияние всех неуправляемых рисков проекта. Модель позволяет учесть пять основных (рисунок 5.5) и пять дополнительных рисков проекта.

Таблица 5.6 - Пять основных факторов риска программного проекта, учитываемые в модели Riskology

НАЗВАНИЕ РИСКА	*ОПИСАНИЕ*	*СТАТУС*
КАЛЕНД. ПЛАН	Изъяны календарного планирования	ВКЛ
ТЕКУЧКА	Текучесть кадров	ВКЛ
РАЗДУВАНИЕ	Раздувание требований	ВКЛ
СПЕЦИФИКАЦИИ	Нарушение спецификаций	ВКЛ
	Низкая производительность	ВКЛ

Характеристики предопределенных в системе Riskology рисков пользователь может изменить, задав значения минимальной, максимальной и наиболее вероятной задержки сроков сдачи проекта из-за влияния данного риска. Можно включить в модель дополнительные собственные риски. Результат моделирования по методу Монте-Карло будет представлен в виде гистограммы распределения срока завершения оцениваемого проекта (рисунок 5.5).

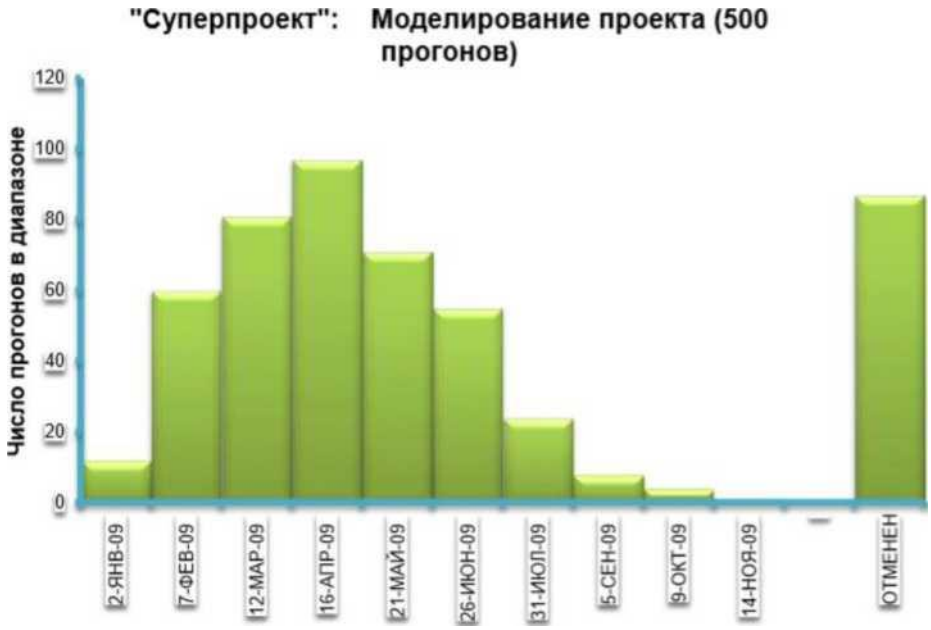

Рисунок 5.5 - Гистограмма распределения возможного срока завершения проекта, рассчитанная по результатам моделирования методом Монте-Карло

На диаграмме также приведено количество случаев, примерно 80 из 500 прогонов, в которых проект, согласно результатам моделирования, был отменен до своего завершения.

6 Планирование реагирования на риски

Планирование реагирования на риски - это процесс разработки путей и определения действий по увеличению возможностей и снижению угроз для целей проекта. Данный процесс начинается после проведения качественного и количественного анализа рисков.

Запланированные операции по реагированию на риски должны соответствовать серьезности риска, быть экономически эффективными в решении проблемы, своевременными, реалистичными в контексте проекта и согласованными со всеми участниками.

Возможны четыре метода реагирования на риски:

- □ Уклонение от риска (risk avoidance).
- □ Передача риска (risk transference).
- □ Снижение рисков (risk mitigation).
- □ Принятие риска (risk acceptance).

Уклонение от риска предполагает изменение плана управления проектом таким образом, чтобы исключить угрозу, вызванную негативным риском, оградить цели проекта от последствий риска или ослабить цели, находящиеся под угрозой (например, уменьшить содержание проекта). Некоторые риски, возникающие на ранних стадиях проекта, можно избежать при помощи уточнения требований, получения дополнительной информации или проведения экспертизы. Например, уклониться от риска можно, если отказаться от реализации рискованного функционального требования или самостоятельно разработать необходимый программный компонент, вместо ожидания поставок продукта от субподрядчика.

Передача риска подразумевает переложение негативных последствий угрозы с ответственностью за реагирование на риск на третью сторону. Передача риска просто переносит ответственность за его управление другой стороне, но риск при этом никуда не девается. Передача риска практически всегда предполагает выплату премии за риск стороне, принимающей на себя риск. Например, заказ на стороне разработки рискованного компонента по фиксированной цене. В IT часто приходится формулировать риски в виде допущений, тем самым передавая его заказчику. Например, оценивая проект внедрения, мы

можем записать допущение о том, что производитель не изменит стоимость лицензий на базовое ПО.

Снижение рисков предполагает понижение вероятности и/или последствий негативного рискованного события до приемлемых пределов. Принятие предупредительных мер по снижению вероятности наступления риска или его последствий часто оказываются более эффективными, нежели усилия по устранению негативных последствий, предпринимаемые после наступления события риска. Например, раннее разрешение архитектурных рисков снижает потери при досрочном закрытии проекта. Или регулярная ревизия поставок заказчиком может снизить вероятность риска его неудовлетворенности конечным результатом. Если в проектной команде высока вероятность увольнения сотрудников, то введение на начальной стадии в проект дополнительных (избыточных) людских ресурсов снижает потери при увольнении членов команды, поскольку не будет затрат на «въезд» в проектный контекст новых участников.

И, наконец, принятие риска означает, что команда проекта осознанно приняла решение не изменять план управления проектом в связи с риском или не нашла подходящей стратегии реагирования. Мы вынуждены принимать все «неизвестные риски».

Принятие - это то, что всегда происходит, когда мы вообще не управляем рисками. Если же мы управляем рисками, то мы можем страховать риски, закладывая резерв в оценки срока завершения и/или трудозатрат. Проактивное отношение к принятым рискам может состоять в разработке план реагирования на риски. Этот план может быть введен в действие только при заранее определенных условиях, если есть уверенность и достаточное количество признаков того, что данный план будет успешно выполнен.

Важно помнить о вторичных рисках (Secondary Risks), возникающих в результате применения реагирования на риски, которые тоже должны быть идентифицированы, проанализированы и при необходимости включены в список управляемых рисков.

7 Главные риски программных проектов и способы реагирования

Список из пяти главных причин провала программных проектов - следующий:

- □ Требования заказчика отсутствуют / не полны / подвержены частым изменениям.
- □ Отсутствие необходимых ресурсов и опыта.
- □ Отсутствие рабочего взаимодействия с заказчиком.
- □ Неполнота планирования. «Забытые работы».
- □ Ошибки в оценках трудоемкостей и сроков работ.

Это звучит банально, но сколько бы раз об этом не твердили ранее, попрежнему, приходится сталкиваться с программными проектами, в которых отсутствуют какие-либо определенные цели и требования. Цитата из жизни: «Была бы разработана хорошая программа, а какой процесс автоматизировать с ее помощью, мы найдем». К этому можно добавить только одно: «Когда человек не знает, к какой пристани он держит путь, для него никакой ветер не будет попутным» (Сенека Луций Аней, философ, 65-3 до н.э.)

К часто упускаемым требованиям можно отнести:

- □ Функциональные
- o Программы установки, настройки, конфигурации.
- o Миграция данных.
- o Интерфейсы с внешними системами.
- o Справочная система.
- □ Общесистемные
- o Производительность.
- o Надежность.
- o Открытость.
- o Масштабируемость.
- o Безопасность.
- o Кросплатформенность.
- o Эргономичность.

Как правило, эти требования «всплывают» при подготовке и проведении приемо-сдаточных испытаний и могут сильно задержать проект по времени и увеличить трудозатраты на его реализацию. Чтобы этого не происходило, следует достигать соглашения с заказчиком по всем перечисленным пунктам предпочтительнее еще на стадии инициации проекта. Например, если требования портируемости продукта на разные аппаратно-программные платформы нет, то это целесообразно включить в раздел концепции с допущениями проекта.

Если вероятность изменений требований проекта высока, то возможны следующие подходы для реагирования на данный риск:

□ Переоценка проекта каждый раз, когда требования добавляются / изменяются (уклонение).

□ Итерационная разработка. Контракт с компенсацией затрат на основе «Time & Materials» (передача риска Заказчику).

□ Учет в оценках трудоемкости и сроков возможности роста требований, например, на 50% (резервирование риска).

И еще, при сборе требований следует соблюдать принцип минимализма Вольтера: «Рассказ закончен не тогда, когда в него нечего добавить, а тогда, когда из него нечего больше выкинуть». Для большинства программных продуктов применим принцип Парето: 80% ценности продукта заключены лишь в 20% требований к нему.

Если у нас в проекте недостаточно квалифицированных специалистов, то мы можем снизить последствия этого риска, применив следующие действия:

□ Привлечь экспертов-консультантов на начальных этапах.

□ Учитывать в оценках трудоемкости издержки на обучение сотрудников.

□ Уменьшать потери от текучести кадров, привлекая на начальном этапе избыточное число участников.

□ Учесть в оценках «время разгона» для новых сотрудников.

Для установления открытых и доверительных отношений с заказчиком, необходимо предпринимать следующие шаги:

□ Постоянное взаимодействие.

□ Согласование пользовательских интерфейсов и разработка прототипа продукта.

□ Периодические поставки тестовых версий конечным пользователям для их оценки.

При планировании работ по проекту часто «забывают»:

□ Обучение.

□ Координация работ.

□ Уточнение требований.

□ Управление конфигурациями.

□ Разработка и поддержка скриптов автосборки.

□ Разработка автотестов.

□ Создание тестовых данных.

□ Обработка запросов на изменения.

И еще. Не стоит надеяться, что участники проекта будут каждую неделю по 40 часов работать именно над вашим проектом. Есть множество причин, по которым они не смогут работать по проекту 100% своего времени. К списку наиболее распространенных причин этого относятся:

□ Сопровождение действующих систем.

□ Повышение квалификации.

□ Участие в подготовке технико-коммерческих предложений.

□ Участие в презентациях.

□ Административная работа.

□ Отпуска, праздники, больничные.

Рекомендация, планировать, что разработчики, которые назначены в ваш проект на 100% будут реально работать над вашими задачами в среднем от 24 до 32 часов в неделю.

8 Управление проектом, направленное на снижение рисков

На стадии инициации проекта оценка его трудоемкости имеет погрешность от 50% до +100%. Это, если оценка хорошая! А если плохая, то неопределенность, а, следовательно, и риски сорвать сроки и превысить плановую трудоемкость, могут быть в разы больше. Если не прилагать специальных усилий этот «дамоклов меч» неопределенности будет висеть над проектом на всем его протяжении (рисунок 5.6).

Рисунок 5.6 - Неопределенность не уменьшается, если управление не направлено на раннее разрешение рисков

Проектом следует управлять так, чтобы риски несвоевременной сдачи и перерасхода ресурсов постоянно снижались.

Ранее мы уже говорили о том, что 80% ценности разработки обусловлена лишь 20% требований к продукту, без реализации которых продукт для заказчика становится просто ненужным. Остальные требования, как правило, так называемые «украшательства», от части которых заказчик, как правило, может отказаться, чтобы получить проект в срок. Поэтому следует в первую очередь реализовывать ключевые функциональные требования.

Но есть и еще архитектурные риски. Известно, что закон Парето применим и к потреблению вычислительных ресурсов: 80% потребления ресурсов (время и память) приходится на 20% компонентов. Поэтому, необходимо реализовывать архитектурно-значимые требования так же в первую очередь, создавая «представительный» прототип будущей системы, который «простреливает» весь стек, применяемых технологий. Прототип позволит измерить и оценить общесистемные свойства будущего продукта: доступность, быстродействие, надежность, масштабируемость и проч. (рисунок 5.7)

Ошибка - реализовывать сначала легкие требования, чтобы продемонстрировать быстрый прогресс проекта.

Рисунок 5.7 - Определение приоритетов требований на первые итерации проекта

Управление, нацеленное на снижение рисков, позволяет существенно снизить

неопределенность на ранних стадиях проекта (рисунок 5.8).

Рисунок 5.8 - Управление, нацеленное на снижение рисков, позволяет уменьшать неопределенность

Проработка ключевых функциональных требований и детальное планирование их реализации позволяет уменьшить разброс начальных оценок, примерно, в 2 раза: от -30% до +50%. Детальное проектирование и разработка прототипа будущей системы позволит получить еще более точные оценки общей трудоемкости: от -10% до +15%.

Может оказаться так, что по результатам прототипирования, уточненные оценки суммарной трудоемкости окажутся неприемлемыми. В этом случае проект придется закрыть досрочно, но потери при этом, будут значительно меньше, чем в случае, если то же самое произойдет, когда проект уже в 2 раза превысит первоначальную оценку трудоемкости.

Если с заказчиком не удается найти взаимоприемлемое решение при первоначальной оценке проекта, то разумно попытаться договориться о выполнении проекта в 2 этапа с самостоятельным финансированием:

1. Исследование. Бизнес-анализ, уточнение требований, проектирование и прототипироание решения, уточнение суммарных оценок трудозатрат. Эта работа, как правило, требует 10 % общих трудозатрат и 20% времени всего проекта.
2. Непосредственно реализация. Если уточненные оценки трудозатрат окажутся приемлемыми для заказчика.

С вменяемыми заказчиками это часто удается.

8 Мониторинг и контроль рисков

Управление рисками должно осуществляться на протяжении всего проекта. Не вести мониторинг рисков в ходе проекта - все равно, что не следить за уровнем топлива при поездке на автомобиле.

Мониторинг и управление рисками - это процесс идентификации, анализа и планирования реагирования на новые риски, отслеживания ранее идентифицированных рисков, а также проверки и исполнения операций реагирования на риски и оценка эффективности этих операций.

В процессе мониторинга и управления рисками используются различные методики, например, анализ трендов и отклонений, для выполнения которых необходимы количественные данные об исполнении, собранные в процессе выполнения проекта.

Мониторинг и управление рисками включает в себя следующие задачи:

- □ Пересмотр рисков.
- □ Аудит рисков.
- □ Анализ отклонений и трендов.

Пересмотр рисков должен проводиться регулярно, согласно расписанию. Управление рисками проекта должно быть одним из пунктов повестки дня всех совещаний команды проекта. Неплохо начинать каждый статус митинг с вопроса: «Ну и какие еще неприятности нас ожидают?» Идентификация новых рисков, и пересмотр известных рисков происходит с использованием процессов, описанных ранее.

Аудит рисков предполагает изучение и предоставление в документальном виде результатов оценки эффективности мероприятий по реагированию на риски, относящихся к идентифицированным рискам, изучение основных причин их возникновения, а также оценку эффективности процесса управления рисками.

Тренды в процессе выполнения проекта подлежат проверке с использованием данных о выполнении. Для мониторинга выполнения всего проекта могут использоваться анализ освоенного объема и другие методы анализа отклонений проекта и трендов (см. Лекция 8. Реализация проекта). На основании выходов этих анализов можно прогнозировать потенциальные отклонения проекта на момент его завершения по показателям стоимости и расписания. Отклонения от базового плана могут указывать на последствия, вызванные как угрозами, так и благоприятными возможностями.

Выводы

Отказываться от управления проектными рисками это все равно, что в кинотеатре не иметь огнетушителей и плана эвакуации на случай пожара.

Все, что мы делаем, управляя проектом разработки ПО, должно быть направлено на борьбу с рисками не уложиться в срок, перерасходовать ресурсы, разработать не тот продукт, который требуется.

Цели управления рисками проекта - снижение вероятности возникновения и/или значимости воздействия неблагоприятных для проекта событий.

Главные причины провала программных проектов:

- □ Требования заказчика отсутствуют / не полны / подвержены частым изменениям.
- □ Отсутствие необходимых ресурсов и опыта.
- □ Отсутствие рабочего взаимодействия с заказчиком
- □ Неполнота планирования. «Забытые работы».
- □ Ошибки в оценках трудоемкостей и сроков работ.

3 ПРАКТИЧЕСКИЙ РАЗДЕЛ

Практическое занятие 1 Организация группового взаимодействия

Задание 1. Дать собственное определение понятиям проект и управление проектом на основе обобщения существующих. Например:

Проект – это уникальное и временное начинание, с определенным началом и концом с целью создать или модифицировать определенный продукт или услугу.

Управление проектом – это область менеджмента, охватывающая те сферы, в которых создание продукта реализуется как уникальный комплекс работ при определенных требованиях к срокам, бюджету и характеристикам результата.

Задание 2. Определить какая деятельность является проектом, а какая – нет.

Организация вечеринки; внедрение новой процедуры подбора персонала компании; уборка квартиры; замена информационной системы по учету труда и заработной платы компании; покраска крупного моста; возведение монумента на площади; повторяющиеся (рутинные) операции предприятия; организация олимпиады в Токио в 2020 г., постройка офисного здания; апгрейд планшета производителем; разработка и вывод на рынок инновационного продукта; замена аппаратного (ПК) и программного обеспечения учебной аудитории ВУЗа; составление ежегодных финансовых отчетов предприятия; осуществление изменений в оргструктуре и кадровом составе организации, строительство Титаника.

Задание 3. Задание по кейсу: продумайте Ваш предпринимательский проект в условиях города? Зафиксируйте данную проектную инициативу в следующем документе:

МОДЕЛЬ ПРОЕКТА «_____»

Отразите:

1. Сущность проекта
2. Какую проблему решает проект?
3. Основные цели, результаты (продукты проекта) и требования к ним
4. Состав работ проекта (описать конкретные действия в ходе реализации проекта)
5. Риски проекта
6. Оценить доход от проекта

Концепция проекта должна отражать, что Вы хотите сделать в проекте, зачем и как Вы это сделаете. Каждая группа должна представить концепцию своего проекта в презентации Power Point.

***Задание* 4** Выявите заинтересованные стороны (шаблон - рисунок 1) и составить матрицу ответственности RACI (шаблон - таблица 1) по проекту

Рисунок 1 – Заинтересованные стороны проекта

Таблица 1 - Матрица ответственности RACI

Пакеты или этапы работ по проекту	Роли заинтересованных сторон (ФИО или должность)			
	Спонсор/куратор Менеджер проекта проекта (начальник (стажёр FPT подразделения ГГ в FPT) Marge)		Заказчик	
…	R		I	
Разработка бюджета проекта	А, Б, С			

R – Responsible (Исполнитель), ответственный за выполнение, должен быть минимум один по каждому пакету или этапу работ;

A – Accountable (Утверждает), перед ним производится отчет в полученном результате (принимает работу у исполнителя), должен быть один на пакет/этап работ;

C – Consult before doing (Согласует) - с ним необходимо предварительное согласование/консультация перед началом выполнения;

I – Inform after doing (Информируемый) оповещается после исполнения.

ПРИМЕРНЫЙ СПИСОК РАБОТ ПО СОЗДАНИЮ ВЕБСАЙТА И ВИДЕОУРОКОВ

При составлении списка работ матрицы RACI необходимо учесть, что работу по созданию продукта (услуги) можно разделить на следующие этапы:

— Подготовительный (создание Технического Задания на разработку);

— Разработка дизайн-макета;

— Верстка;

— Программирование;

— Наполнение контентом (информацией);
— Расположение сайта в сети Интернет;
— Тестирование сайта
— Раскрутка сайта
— Администрирование (поддержка) сайта

Задание 5 Исходя из того, что проектная команда уже сформирована и не нуждается в развитии, описать каким образом будет осуществляться управление её членами, в т.ч. составить план по контролю и мотивации

Практические занятия 2-5 Менеджмент и методологии разработки программного обеспечения. Управление программными проектами. Планирование. Стандарты и стандартизация в области разработки ПО

По выбранному проекту предоставить (в презентации Power Point) следующую информацию:

Задание 1. Общая информация о проекте (аннотация):

- Наименование проекта
- Менеджер проекта
- Даты начала и окончания, длительность проекта
- Причины инициации проекта (обоснование)
- Цели
- Продукты/результаты проекта и требования к ним
- Оценка бюджета проекта
- Список заинтересованных сторон

Задание 2. Описать требования проекта.

Задание 3. Состав работ проекта

Представить иерархическое разбиение всей работы, которую необходимо выполнить для достижения целей проекта, как показано на рисунке 1. В упрощённом варианте пакетов работ может не быть, если Вы используете только три уровня иерархии.

Задание 3. Расписание проекта

Используя составленную иерархическую структуру работ по проекту, составить упрощённое расписание проекта в таблице MS Excel, как показано в таблице 1.

Таблица 1 – Структура работ

ID	Наименование	Предшественник	Начало (дата)	Конец (дата)	Длительность (дней)
	Проект				
1	Создание результата 1.				
1.1	Пакет работ 1.1.- ...				
1.1.1					
1.1.2					
1.2					
2.	Создание результата 2.				
2.1					
2.2					
2.2.1					

Задание 4. Исполнение и контроль исполнения проекта

Выполнить запланированные работы и осуществить контроль исполнения проекта, ответив на вопросы:

- Все ли работы выполнены?
- Получены ли запланированные продукты/результаты проекта?
- Соответствуют ли продукты/результаты проекта требованиям к ним?
- Соблюдено ли расписание и бюджет проекта?

Практическое занятие 6 Продвижение высокотехнологичной продукции на деловой и потребительские рынки

Задание 1 Опишите потенциальную целевую аудиторию (регион проживания, пол, возраст, сфера деятельности, образ жизни и т. п).

2 Опишите потребности, которые будут удовлетворяться новым программным продуктом/услугой.

3 Укажите перечень существующих на рынке анатогов-конкуреитов (с указанием их функций, стоимости, достоинств и недостатков).

4 Опишите предполагаемых конкурентных преимуществ продукта /услуги (чем конкретно он выделяется на фоне описанных в п. 4 конкурентов).

Практическое занятие 7-8 Управление рисками проекта

Задание 1 Оцените основные риски проекта, составив матрицу рисков.

Таблица 1 - Матрица оценки риска

Событие	Вероятность	Степень серъезности	Трудность обнаружения	Время

Задание 2 Составить план управления рисками проекта:

1. Иерархическая структура рисков - перечисляет источники рисков либо категории и подкатегории, в рамках которых могут возникать риски

2. Методом мозгового штурма получить список идентифицированных рисков и предполагаемые даты их наступления

3. Ранжирование рисков по значимости с использованием Матрицы вероятностей и влияний (последствий) рисков. В её ячейках - ранги рисков, определяющие их значимость для дальнейшего планирования реагирования

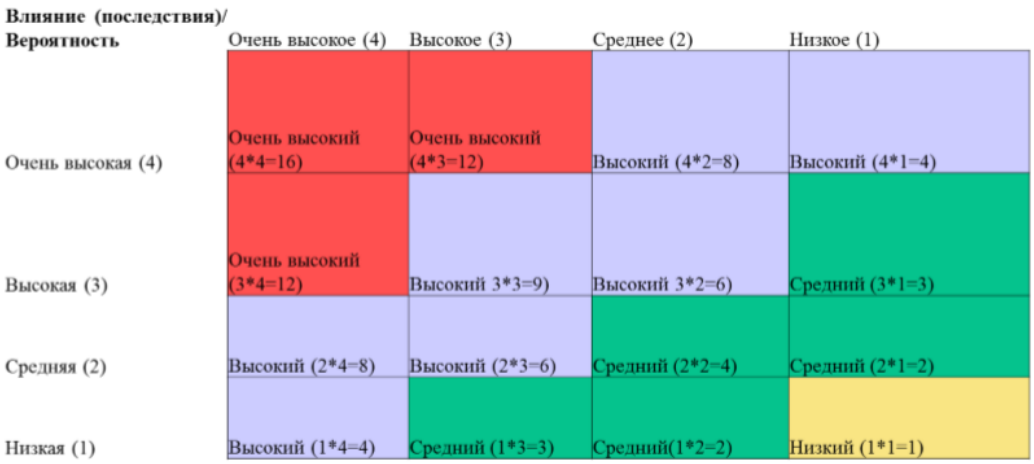

Рисунок 1 – Матрица вероятностей и влияний (последствий)

4. Количественный анализ наиболее значимых рисков

5. Упреждающие мероприятия (реакции) по снижению вероятностей наступления наиболее значимых рисков и их последствий/влияний в случае наступления

6. Мероприятия (реакции) по уменьшению последствий на случай возникновения наиболее значимых рисков

7. Реестр рисков.

Таблица 2 - – Шаблон реестра рисков проекта

Риск	Категория (Источник)	Вероятность наступления (% и по шкале от 1 до 4)	Степень влияния последствия но шкале от 1 до 4	Рейтинг	Упреждающие мероприятия по снижению вероятности наступления
Уход из проекта ключевою члена проектной команды	Организационный риск: потеря ресурса	60%, 3-высокая	3 - высокое	9(=3*3)-высокий	Увеличить его оплату труда
Несвоевременное финансирование	Организационный: финансирование	20%, 1 - низкая	3 - высокое	3(-1*3)-среднип	отсутствуют

4 РАЗДЕЛ КОНТРОЛЯ ЗНАНИЙ

4.1 Вопросы к зачету по учебной дисциплине «Менеджмент программного обеспечения»

1. Отличия программной инженерии от других отраслей.
2. Эволюция подходов к управлению программными проектами.
3. Модели процесса разработки ПО.
4. Критерии успешности проекта.
5. Проект и организационная структура компании.
6. Организация проектной команды.
7. Командные роли.
8. Жизненный цикл проекта.
9. Фазы и продукты.
10. Управление приоритетами проектов.
11. Концепция проекта.
12. Цели и результаты проекта.
13. Допущения и ограничения.
14. Ключевые участники и заинтересованные стороны.
15. Обоснование полезности проекта.
16. Уточнение содержания и состава работ.
17. Планирование управления содержанием.
18. Планирование организационной структуры.
19. Планирование управления конфигурациям.
20. Планирование управления качеством
21. Базовое расписание проекта.
22. Методологии проектной деятельности.
23. Каскадная методология.
24. Итерационная методология.
25. V – методология.
26. Scrum.
27. Маркетинг ИТ продуктов и услуг.
28. Продвижение ИТ продуктов.
29. Прогнозирование.
30. Планирование управления рисками.
31. Идентификация рисков.
32. Качественный анализ рисков.
33. Количественный анализ рисков.
34. Планирование реагирования на риски.
35. Главные риски программных проектов и способы реагирования.
36. Управление проектом, направленное на снижение рисков.
37. Мониторинг и контроль рисков.
38. Оценка - вероятностное утверждение.

4.2 Критерии оценок результатов учебной деятельности студентов по учебной дисциплине «Менеджмент программного обеспечения» (на основании письма Министерства образования Республики Беларусь от 28.05.2013 г. № 09- 10/53-ПО)

Десятибалльная шкала в зависимости от величины балла и отметки включает следующие критерии:

10 (десять) баллов, зачтено выставляется за систематизированные, глубокие и полные знания по всем разделам учебной программы учреждения высшего образования по учебной дисциплине, а также по основным вопросам, выходящим за её пределы; за точное использование научной терминологии (в том числе на иностранном языке), грамотное, логически правильное изложение ответа на вопросы; за безупречное владение инструментарием учебной дисциплины, умение его эффективно использовать в постановке и решении научных и профессиональных задач; за выраженную способность самостоятельно и творчески решать сложные проблемы в нестандартной ситуации; за полное и глубокое усвоение основной, дополнительной литературы, по изучаемой учебной дисциплине; за умение свободно ориентироваться в теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им аналитическую оценку, использовать научные достижения других дисциплин; за творческую самостоятельную работу на практических, лабораторных занятиях, активное творческое участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

9 (девять) баллов, зачтено выставляется за систематизированные, глубокие и полные знания по всем разделам учебной программы учреждения высшего образования по учебной дисциплине; за точное использование научной терминологии (в том числе на иностранном языке), грамотное, логически правильное изложение ответа на вопросы; за владение инструментарием учебной дисциплины, умение его эффективно использовать в постановке и решении научных и профессиональных задач; за способность самостоятельно и творчески решать сложные проблемы в нестандартной ситуации в рамках учебной программы учреждения высшего образования по учебной дисциплине; за полное усвоение основной и дополнительной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за умение ориентироваться в теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им аналитическую оценку; за систематическую, активную самостоятельную работу на практических, лабораторных занятиях, творческое участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

8 (восемь) баллов, зачтено выставляется за систематизированные, глубокие и полные знания по всем разделам учебной программы учреждения высшего образования по учебной дисциплине в объеме учебной программы учреждения высшего образования по учебной дисциплине; за использование научной терминологии (в том числе на иностранном языке), грамотное,

логически правильное изложение ответа на вопросы, умение делать обоснованные выводы и обобщения; за владение инструментарием учебной дисциплины (методами комплексного анализа, техникой информационных технологий), умение его использовать в постановке и решении научных и профессиональных задач; за способность самостоятельно решать сложные проблемы в рамках учебной программы учреждения высшего образования по учебной дисциплине; за усвоение основной и дополнительной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за умение ориентироваться в теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им аналитическую оценку; за активную самостоятельную работу на практических, лабораторных занятиях, систематическое участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

7 (семь) баллов, зачтено выставляется за систематизированные, глубокие и полные знания по всем разделам учебной программы учреждения высшего образования по учебной дисциплине; за использование научной терминологии (в том числе на иностранном языке), грамотное, логически правильное изложение ответа на вопросы, умение делать обоснованные выводы и обобщения; за владение инструментарием учебной дисциплины, умение его использовать в постановке и решении научных и профессиональных задач; за свободное владение типовыми решениями в рамках учебной программы учреждения высшего образования по учебной дисциплине; за усвоение основной и дополнительной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за умение ориентироваться в основных теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им аналитическую оценку; за самостоятельную работу на практических, лабораторных занятиях, участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

6 (шесть) баллов, зачтено выставляется за достаточно полные и систематизированные знания в объеме учебной программы учреждения высшего образования по учебной дисциплине; за использование необходимой научной терминологии, грамотное, логически правильное изложение ответа на вопросы, умение делать обобщения и обоснованные выводы; за владение инструментарием учебной дисциплины, умение его использовать в решении учебных и профессиональных задач; за способность самостоятельно применять типовые решения в рамках, учебной программы учреждения высшего образования по учебной дисциплине; за усвоение основной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за умение ориентироваться в базовых теориях, концепциях и направлениях по изучаемой дисциплине и давать им сравнительную оценку; за активную самостоятельную работу на практических, лабораторных занятиях, периодическое участие в групповых обсуждениях, высокий уровень культуры исполнения заданий.

5 (пять) баллов, зачтено выставляется за достаточные знания в объеме

учебной программы учреждения высшего образования по учебной дисциплине; за использование научной терминологии, грамотное, логически правильное изложение ответа на вопросы, умение делать выводы; за владение инструментарием учебной дисциплины, умение его использовать в решении учебных и профессиональных задач; за способность самостоятельно применять типовые решения в рамках учебной программы учреждения высшего образования по учебной дисциплине; за усвоение основной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за умение ориентироваться в базовых теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им сравнительную оценку; за самостоятельную работу на практических, лабораторных занятиях, фрагментарное участие в групповых обсуждениях, достаточный уровень культуры исполнения заданий.

4 (четыре) балла, зачтено выставляется за достаточный объем знаний в рамках образовательного стандарта высшего образования; за усвоение основной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за использование научной терминологии, логическое изложение ответа на вопросы, умение делать выводы без существенных ошибок; за владение инструментарием учебной дисциплины, умение его использовать в решении стандартных (типовых) задач; за умение под руководством преподавателя решать стандартные (типовые) задачи; за умение ориентироваться в основных теориях, концепциях и направлениях по изучаемой учебной дисциплине и давать им оценку; за работу под руководством преподавателя на практических, лабораторных занятиях, допустимый уровень культуры исполнения заданий.

3 (три) балла, не зачтено выставляется за недостаточно полный объем знаний в рамках образовательного стандарта высшего образования; за знание части основной литературы, рекомендованной учебной программой учреждения высшего образования по учебной дисциплине; за использование научной терминологии, изложение ответа на вопросы с существенными, логическими ошибками; за слабое владение инструментарием учебной дисциплины, некомпетентность в решении стандартных (типовых) задач; за неумение ориентироваться в основных теориях, концепциях и направлениях изучаемой учебной дисциплины; за пассивность на практических и лабораторных занятиях, низкий уровень культуры исполнения заданий.

2 (два) балла, не зачтено выставляется за фрагментарные знания в рамках образовательного стандарта высшего образования; за знания отдельных литературных источников, рекомендованных учебной программой учреждения высшего образования по учебной дисциплине; за неумение использовать научную терминологию учебной дисциплины, наличие в ответе грубых, логических ошибок; за пассивность на практических и лабораторных занятиях, низкий уровень культуры исполнения заданий.

1 (один) балл, не зачтено выставляется за отсутствие знаний и (компетенций) в рамках образовательного, стандарта высшего образования,

отказ от ответа, неявка на аттестацию без уважительной причины.

Экзаменуемые имеют право пользоваться во время проведения зачета учебной программой курса. Результаты зачета объявляются, как правило, в день его проведения и заносятся в зачетную книжку экзаменуемого и экзаменационную ведомость.

Студенты, не выполнившие в полном объеме требования по изучению данной дисциплины, не допускаются кафедрой к сдаче экзамена.

4.3 Самостоятельная работа по дисциплине «Менеджмент программного обеспечения»

В ходе освоения студентам рекомендуется использовать в качестве информационно-методической поддержки электронные презентации лекционных занятий, задания для самостоятельной работы, контрольные вопросы. Наиболее эффективными формами и методами организации самостоятельной работы студентов являются: выполнение заданий по проекту.

4.4 Образец тестовых заданий по учебной дисциплине «Менеджмент программного обеспечения»

::1::Два вида организации человеческой деятельности в управлении проектами:
=операционная и проектная;
~проектная и программная;
~операционная и процессорная;
~процессорная и программная;
~портфельная и программная
::2::В операционной деятельности отсутствует
~известные внешние условия;
~многократно изученные производственные операции;
= необходимость новых возможностей и творчества;
~известные функции исполнителей;
~постоянные функции исполнителей
::3::В каком случае применяется проектная деятельность?
~известны внешние условия;
~многократно изучены производственные операции;
~известны функции исполнителей;
~постоянны функции исполнителей;
=необходимость новых возможностей и творчества
::4::Главное различие операционной деятельности от проектной состоит в ~уникальности;
=повторяемости процесса;
~отсутствии ограничений по времени;
~отсутствие правил;
~отсутствие нормативов
::5::Задача проектной деятельности состоит в
~установлении нормативов;
~ограничении ресурсов;
~обеспечении развития бизнеса;
~поиске новых возможностей;
=достижении конкретной бизнес-цели

::6::Задача операционной деятельности состоит в
~установлении производственных нормативов;
~ограничении доступности ресурсов;
=обеспечении нормального течения бизнеса;
~поиске новых возможностей;
~достижении конкретной бизнес-цели
::7::Проект - это
~средство оперативного развития;
=средство стратегического развития;
~средство тактического развития;
~средство бизнес-развития;
~средство производственного развития
::8::Цель проекта есть
~определение способа достижения прибыли;
~преобразование проекта в организацию;
~разработка нормативов для работы в операционной деятельности;
=описание того, что необходимо достичь;
~обеспечение нормальной работы
::9::Правильная последовательность проекта
=Цель ^ Стратегия ^ Портфель ^ Программа ^ Проект ^ Работа;
~Стратегия ^ Цель ^ Программа ^ Портфель ^ Проект ^ Работа;
~Цель ^ Стратегия ^ Портфель ^ Программа ^ Проект;
~Стратегия ^ Цель ^ Программа ^ Проект ^ Работа;
~Цель ^ Стратегия ^ Программа ^ Проект ^ Портфель ^ Работа
::10::Проект способствует достижению следующих целей
~тактических;
=стратегических;
~оперативных;
~аналитических;
~временных
::11::Какие потери можно отнести к материальным?
~ущерб репутации;
~ущерб здоровью;
~невыполнение сроков сдачи объекта;
~потери рабочего времени;
=потери сырья
::12::На основании чего формируются новые риски в процессе реализации проекта?
=отклонения фактических от базовых значений уже реализованных работ проекта;
~интервью со специалистами по предметной области;
~новых требований к проекту;
~интервью с заинтересованными сторонами;
~интервью с членами проектной команды
::13::Используются ли количественные показатели при выборе перечня рисков?
~в зависимости от типа проекта;
=да, используются для более точной оценки рисков;
~нет, не используются;
~в зависимости от типа организации;
~в зависимости от длительности проекта
::14::Используется ли матрица вероятности и влияния на этапе формирования плана мероприятий по управлению рисками?
~в зависимости от типа организации;
~в зависимости от длительности проекта;

~в зависимости от типа проекта;
=да, используется для выбора наиболее влиятельных рисков;
~нет, не используется
::15::Кто ответственный за реализацию антирисковых мероприятий?
~координатор проекта;
~главный инженер проекта;
~руководитель проекта;
~менеджер по рискам;
=любой участник команды, ответственный за риск
::16::Управление риском проекта - это
=системное применение политики, процедур и методов управления к *задачам определения ситуации,* идентификации, анализа, оценки, обработки, мониторинга риска и обмена информацией, для обеспечения снижения потерь и увеличения рентабельности;
~системное применение политики, процедур и методов управления *целями проекта*, анализа, оценки, обработки, мониторинга информацией, для обеспечения снижения потерь и увеличения рентабельности;
~системное применение политики, процедур и методов *управления командой* проекта и обмена информацией, для обеспечения снижения потерь и увеличения рентабельности;
~системное применение политики, процедур и методов управления к задачам определения ситуации, мониторинга риска и обмена информацией, для обеспечения снижения потерь;
~системное применение политики, процедур и методов управления к задачам определения ситуации, мониторинга риска и обмена информацией, для обеспечения снижения потерь;
~системное применение политики, процедур и методов управления к выбору спонсоров
::17::К способам снижения проектного риска относится
~мотивирование;
~планирование;
=диверсификация;
~контроль;
~отсутствие инициативы

4.5 Критерии оценки результатов ККР (компьютерное тестирование)

10 - балльная шкала

10 баллов = 100% (правильных ответов)
98% < 9 баллов < 99%
96% < 8 баллов < 97%
93% < 7 баллов < 95%
82% < 6 баллов < 92%
71% < 5 баллов < 81%
60% < **4 балла** < 70%
50% < 3 балла < 59%
40% < 2 балла < 49%
1 балл < 40%
0 баллов тест не закончен

Для получения положительной отметки (**4 балла**) студенту необходимо ответить правильно на 30 - 34 вопроса из 50 вопросов, случайно выбранных компьютерной программой по учебной дисциплине.

5 ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ

5.1 Учебная программа по учебной дисциплине «Менеджмент программного обеспечения»

Учреждение образования
«Гомельский государственный университет имени Франциска Скорины»

УТВЕРЖДАЮ
Проректор по учебной работе
ГГУ имени Ф. Скорины
_____ И. В. Семченко
05.05.2021
(дата утверждения)
Регистрационный № УД-2021-68 /уч.

МЕНЕДЖМЕНТ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Учебная программа учреждения высшего образования
по учебной дисциплине для направления специальности
1-31 03 07-01 Прикладная информатика
(программное обеспечение компьютерных систем)

2021

Учебная программа составлена на основе типовой программы специальности 1-31 03 07 Прикладная информатика (по направлениям), утвержденная 20.04.2020 (регистрационный № ТД-G.641.тип.) и учебного плана специальности 1-31 03 07 Прикладная информатика (по направлениям), направление специальности 1-31 03 07-01 Прикладная информатика (программное обеспечение компьютерных систем), утвержденного 15.04.2020 (регистрационный номер № G 31-01-20/УП).

СОСТАВИТЕЛЬ**:**
Марченко Л. Н. – заведующий кафедрой фундаментальной и прикладной математики Гомельского государственного университета имени Франциска Скорины», кандидат технических наук, доцент

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой фундаментальной и прикладной математики УО «Гомельский государственный университет имени Франциска Скорины»,
(протокол № 9 от 29.04.2021)

Научно-методическим советом УО «Гомельский государственный университет имени Франциска Скорины»
(протокол № 6 от 05.05.2021)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Учебная программа по учебной дисциплине «Менеджмент программного обеспечения» составлена с учетом требований следующих нормативных и методических документов:

1. Образовательный стандарт Республики Беларусь «Высшее образование. Первая ступень. Специальность 1-31 03 07 Прикладная информатика (по направлениям)», утвержден постановление Министерства образования Республики Беларусь.
2. Учебный план специальности 1-31 03 07 Прикладная информатика (по направлениям), направление специальности 1-31 03 07 01 Прикладная информатика (программное обеспечение компьютерных систем), утвержденный 15.04.2020 (регистрационный номер № G 31-01-20/УП).
3. Порядок разработки и утверждения учебных программ и программ практики для реализации содержания образовательных программ высшего образования, утвержденный Министром образования Республики Беларусь от 27.05.2019.
4. Типовая программа специальности 1-31 03 07 Прикладная информатика (по направлениям), утвержденная 20.04.2020 (регистрационный № ТД-G.641.тип.)

Настоящая учебная программа определяет цели и задачи изучения дисциплины «Менеджмент программного обеспечения» студентами специальности 1-31 03 07 «Прикладная информатика» (по направлениям), направление специальности 1-31 03 07 01 «Прикладная информатика (программное обеспечение компьютерных систем)», ее содержание и структуру, требования к результатам учебной деятельности и форму аттестации.

Учебная дисциплина «Менеджмент программного обеспечения» относится к государственному компоненту цикла специальных дисциплин.

Специалисты, занятые в сфере информационно-коммуникационных технологий, имеют дело с управлением, чаще всего – руководством разработкой программного (аппаратного) обеспечения вычислительных машин и систем, а многим специалистам также предстоит заниматься продвижением программных и аппаратных средств на рынок.

Цели учебной дисциплины:

– развитие профессиональных компетенций студентов в области управления разработкой программного обеспечения и ИТ-проектами;

– обучение навыкам управления коллективом специалистов, продвижения программной и аппаратной продукции на рынок и разработки соответствующей технической, деловой и потребительской документации.

Основные задачи решаемые при изучении дисциплины «Менеджмент программного обеспечения»:

– изучение проблематики менеджмента в разработке программного обеспечения;

– изучение методологии программного обеспечения; – изучение структуры

и менеджмента проекта; – изучение основ управления рисками и качеством;

– ознакомление со стандартами в области менеджмента программного обеспечения.

Дисциплина «Менеджмент программного обеспечения» является важнейшей составной частью в подготовке специалистов в области прикладной математики и информатики и связана с учебными дисциплинами «Проектирование программных систем» и «Технологии программирования».

В результате изучения учебной дисциплины студент должен: знать:

- особенности менеджмента в разработке программного обеспечения;
- методологические стратегии, структуру и менеджмент проекта;
- производственные функции и технологические аспекты развития программных систем в моделях жизненного цикла;

уметь:

- разрабатывать структуру и менеджмент проекта;
- выполнять планирование проекта;
- распределять функциональные роли в коллективе разработчиков;
- моделировать цикл управления проектом;
- анализировать качество и управлять рисками;

владеть:

- информацией о проблематике менеджмента в разработке программного обеспечения;
- информацией о методологиях разработки программного обеспечения;
- принципами и приемами оперирования требованиями;
- методами разработки программных проектов.

В результате изучения учебной дисциплины «Менеджмент программного обеспечения» формируются следующие компетенции:

АК-1. Уметь применять базовые научно-теоретические знания для теоретических и практических задач.

АК-2. Владеть системным и сравнительным анализом.

АК-3. Владел» исследовательскими навыками.

АК-4. Уметь работать самостоятельно.

АК-5. Быть способным порождать новые идеи (обладать креативностью).

АК-6. Владеть междисциплинарным подходом при решении проблем.

ПК-18. Оказывать консультации по вопросам работы программного обеспечения, в том числе разработанного сторонними организациями.

ПК-21. Анализировать результаты работы установленного программною обеспечения и вырабатывать предложения по улучшению качества его работы.

ПК-23. 11роводить обучение специалистов, занимающихся эксплуатацией программного обеспечения.

ПК-25. Проводить обучение специалистов, занимающихся эксплуатацией насыщенных Интернет приложений.

ПК-29. Взаимодействовать со специалистами смежных профилей.

ПК-30. Вести переговоры с другими заинтересованными участниками.

ПК-31. Готовить доклады, материалы к презентациям.

ПК-32. Работать с юридической литературой и трудовым законодательством

Форма получения высшего образования: дневная.

Дисциплина изучается студентами 2-го курса направления специальности 1-31 03 07-01 Прикладная информатика (программное обеспечение компьютерных систем) в 4-м семестре дневной формы получения высшего образования (первая ступень).

Общее количество часов – 54 (2 зачетных единиц); аудиторное количество часов – 34, из них лекций – 18 (из них управляемая самостоятельная работа студентов – 6 часов), лабораторные занятия – 16. Форма отчётности – зачет (4 семестр).

ПРИМЕРНЫЙ ТЕМАТИЧЕСКИЙ ПЛАН

		Всего	**В том числе**	
	Название раздела, темы		**Лекции**	**Практические занятия**
1.	Основы менеджмента программного обеспечения	4	2	-
2.	Организация группового взаимодействия	4	2	2
3.	Менеджмент и методологии разработки программного обеспечения		2	2
4.	Управление программными проектами	4	2	2
5.	Планирование	4	2	2
6.	Стандарты и стандартизация в области разработки ПО			
7.	Продвижение высокотехнологичной продукции на деловой и потребительские рынки	4	2	2
8.	Прогнозирование	4	2	2
9.	Управление рисками	4	2	2
	Всего	**34**	**18**	**16**

СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Тема 1. Основы менеджмента программного обеспечения.

Определение менеджмента. Национальные особенности менеджмента. Типовые организационные структуры IT компаний. Классические и современные задачи менеджмента. Концепция эффективного менеджмента. Цели и задачи МПО. Уровни МПО. Программный продукт и дополнительные задачи МПО. Менеджмент в системе деятельностей проекта. Элементы современного подхода к МПО. Модели процесса разработки ПО. Разница между Project, Program Project, Program Portfolio. Project vs Process.

Тема 2. Организация группового взаимодействия

Разработка плана управления человеческими ресурсами. Набор команды проекта. Основной принципы совмещения ролей. Развитие команды проекта. Управление командой. Эффективные коммуникации проекта. Разрешение конфликтов. Кодекс этики и профессионального поведения PMI. Элементы технологий управления взаимоотношения с клиентами. Основы переговоров.

Теории мотивации. Мотивация программистов и их менеджеров.

Тема 3. Менеджмент и методологии разработки программного обеспечения.

Последовательный и итерационный проект. Жесткие, гибкие, адаптивные стратегии в методологиях разработки Г10. Принципы совмещения методологий.

Принципы работы с требованиями к программному обеспечению.

Тема 4. Управление программными проектами.

Жизненный цикл проекта. Законы Брукса в жизненном цикле разработки программного обеспечения. Функции управления разработкой ПО. Классические ошибки в управлении программными проектами: люди, процесс, технология, продукт.

Тема 5. Планирование.

Понятие о планировании. Классификационные, иерархические признаки планирования. Основные принципы планирования. Стратегическое планирование. SWOT-анализ. Тактическое планирование. Оформление тактического планирования. Оперативное планирование. Процессы планирования по PMBOK.

Технологии и системы планирования: ERP, CSRP. Планирование и контроль менеджмента. Быстрое предприятие и быстрое планирование. Консервативные, технические, адаптивные методики планирования. Технология личного планирования. Системы начального уровня: календарное планирование и контроль.

Тема 6 Стандарты и стандартизация в области разработки ПО

Понятие о стандартизации в области менеджмента. Стандарт, свод правил,

свод знаний в области ПО. Элементы классификации стандартов в области ПО. Стандарты ISO серии 900х. Применение стандартов ISO/TC 176 в процессном подходе. Стандарты ISO серии 9126. Стандарт TickIT. Стандарты SEI SW-CMM. Стандартизация по Project Management. PMBOK.

Тема 7. Продвижение высокотехнологичной продукции на деловой и потребительские рынки.

Системы маркетинговых исследований и маркетинговой информации. Особенности рынка программного обеспечения B2B и его отличия от рынка B2C. Понятие о прямой коммуникации. Продвижение программного обеспечения: стратегия коммуникации и стимулирования спроса. Интернет-маркетинг.

Тема 8. Прогнозирование.

Основные методы прогнозирования. Элементы технического анализа. Экспертные и эвристические методы прогнозирования. Метод Дельфи. Методы сценариев. Ситуационный анализ и моделирование. Оценка качества прогнозов.

Тема 9. Управление рисками.

Понятия о рисках в области разработки ПО. План управления рисками. Идентификация рисков. Выставление приоритетов рискам. Работа с рисками: исключение, уменьшение, переключение, предупреждение ущерба. Методический аппарат анализа риска. Математические методы оценки рискам принятия решений в рисковых ситуациях.

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ
Менеджмент программного обеспечения

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов			Материальное обеспечение занятия	Литература	Формы контроля знаний
		Лекции	Практические занятия	Количество часов УСР			
1	2	3	4	5	6	7	8
1.	**Основы менеджмента программного обеспечения.** 1 Определение менеджмента. 2 Типовые организационные структуры IT компаний. 3 Концепция эффективного менеджмента. 4 Разница между Project, Program Project, Program Portfolio. Project vs Process.	2	–	–	Тексты лекций	[1-8]	Устный опрос
2.	**Тема 2. Организация группового взаимодействия** 1 Разработка плана управления человеческими ресурсами. 2 Основной принципы совмещения ролей. 3 Эффективные коммуникации проекта. 4 Основы переговоров. 5 Теории мотивации.	2	2	–	Тексты лекций	[1-8]	Устный опрос
3.	**Менеджмент и методологии разработки программного обеспечения.** 1 Последовательный и итерационный проект. 2 Жесткие, гибкие, адаптивные стратегии в методологиях разработки Г10. 3 Принципы совмещения методологий. 4 Принципы работы с требованиями к программному обеспечению.	2	2	–	Тексты лекций		Устный опрос
4.	**Управление программными проектами.** 1 Жизненный цикл проекта. 2 Законы Брукса в жизненном цикле разработки программного обеспечения. 3 Функции управления разработкой ПО. 4 Классические ошибки в управлении программными проектами: люди, процесс, технология, продукт.	2	2	–	Тексты лекций		Устный опрос

5.	**Планирование.** 1 Классификационные, иерархические признаки планирования. Основные принципы планирования. 2 Стратегическое планирование. 3 SWOT-анализ. 4 Тактическое и оперативное планирование. Оформление тактического планирования. 5 Технологии и системы планирования: ERP, CSRP.	2	2	–	Тексты лекций		Устный опрос
6.	**Стандарты и стандартизация в области разработки ПО** 1 Понятие о стандартизации в области менеджмента. 2 Стандарт, свод правил, свод знаний в области ПО. 3 Элементы классификации стандартов в области ПО. 4 Стандарты SEI SW-CMM. 5 Стандартизация по Project Management. PMBOK.	-	2	2	Тексты лекций		Групповая консультация
7.	**Продвижение высокотехнологичной продукции на деловой и потребительские рынки**. 1 Системы маркетинговых исследований и маркетинговой информации. 2 Особенности рынка программного обеспечения B2B и его отличия от рынка B2C. 3 Понятие о прямой коммуникации. 4 Продвижение программного обеспечения: стратегия коммуникации и стимулирования спроса. 5 Интернет-маркетинг.	2	2	–	Тексты лекций		Устный опрос
8.	**Прогнозирование.** 1 Основные методы прогнозирования. 2 Элементы технического анализа. 3 Метод Дельфи. 4 Методы сценариев.	2	2	–	Тексты лекций		Устный опрос
9.	**Управление рисками.** 1 Понятия о рисках в области разработки ПО. 2 План управления рисками. 3 Идентификация рисков. 4 Выставление приоритетов рискам. 5 Работа с рисками: исключение, уменьшение, переключение, предупреждение ущерба.	2	02	–	Тексты лекций		Устный опрос
	ИТОГО	12	16	6			

.

Заведующий кафедрой фундаментальной и прикладной математики,
кандидат технических наук, доцент Л.Н.Марченко

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Перечень практических занятий

1. Организация группового взаимодействия
2. Менеджмент и методологии разработки программного обеспечения
3. Управление программными проектами
4. Планирование
5. Стандарты и стандартизация в области разработки ПО
6. Продвижение высокотехнологичной продукции на деловой и потребительские рынки
7. Прогнозирование.
8. Управление рисками

Рекомендуемые формы контроля знаний

1 Устный опрос.
2 Групповая консультация.
3 Тестовые задания.

Методические рекомендации по организации и выполнению УСР по дисциплине государственного компонента

«Менеджмент программного обеспечения»

Для самостоятельного изучения выделяются следующие темы.
1 Основные понятия и определения теории множеств.

Самостоятельное изучение данных тем преследует следующие цели:

– овладеть основными понятиями, определениями,

– самостоятельно анализировать полученные результаты, делать соответствующие выводы.

Тема 6 Стандарты и стандартизация в области разработки ПО

Понятие о стандартизации в области менеджмента. Стандарт, свод правил, свод знаний в области ПО. Элементы классификации стандартов в области ПО. Стандарты ISO серии 900х. Применение стандартов ISO/TC 176 в процессном подходе. Стандарты ISO серии 9126. Стандарт TickIT. Стандарты SEI SW-CMM. Стандартизация по Project Management. PMBOK

Учебная программа УСР

*1 Тема «**Стандарты и стандартизация в области разработки ПО**» – 2 часа.*

Цели: 1) овладеть основными понятиями и определениями по теме;

2) сформировать компетенцию понимании основных понятий: стандарт,

свод правил, свод знаний в области ПО

.

Виды заданий УСР с учетом модулей сложности
по теме «Стандарты и стандартизация в области разработки ПО»

А) *Задания, формирующие знания по учебному материалу на уровне узнавания:*

1 Составление глоссария основных определений.
2 Подготовка краткого конспекта по теме.
3 Структурирование материала в виде таблиц или схем.

Форма выполнения заданий – индивидуальная.

Форма контроля выполнения заданий – устное сообщение.

Б) *Задания, формирующие компетенции на уровне воспроизведения*:

1 Формулировка основных понятий и определений.
2 Знание основных законов стандартов.

Форма выполнения заданий – индивидуальная и групповая.

Форма контроля выполнения заданий – устный опрос, групповая консультация.

В) *Задания, формирующие компетенции на уровне применение полученных знаний:*

1 Описание стандартов для конкретного программного обеспечения.

Форма выполнения заданий – индивидуальная.

Форма контроля выполнения заданий – защита отчета проведенного исследования.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

Основная

1. Алиев, В. С. Бизнес-планирование с использованием программы Project Expert (полный курс): учебное пособие / В. С. Алиев, Д.В. Чистов. – Москва: ИНФРА-М, 2014. – 352 с.
2. Беляцкий, Н.П. Бизнес-лидерство : учебное пособие : [для студентов и магистрантов экономических специальностей вузов] / Н.П.Беляцкий . – Минск : Амалфея : Мисанта, 2016 . – 247 с. : ил., табл. – 1 экз.
3. Верзух, Э. Управление проектами : ускоренный курс по программе MBA / Э. Верзух . – Спб.: ООО «Диалектика», 2020 . – 480 с. - 1 экз.
4. Гейзлер, П.С. Управление проектами : учебное пособие / П.С. Гейзлер. – Минск: Книжный Дом «Мисанта, 2005. – 288 с.
5. Горбунов, В.Л. Бизнес-планирование с оценкой рисков и эффективности проектов : научно-практическое пособие / В.Л. Горбунов . – 2-е изд . – Москва : РИОР : ИНФРА-М, 2021 . – 288 с.
6. Инвестиционный менеджмент : учебник для студентов вузов, обучающихся по направлению подготовки "Экономика" (квалификация (степень) "бакалавр") / С.Е. Метелев [и др.] . – Москва : ФОРУМ, 2020 . – 287 с.
7. Петухова, С.В. Бизнес-планирование: как обосновать и реализовать бизнес-проект: практическое руководство / С.В.Петухова. – Москва: Омега-Л, 2014.– 352 с.
8. Полковников, А.В. Управление проектами. Полный курс MBA / А.В.Полковников, М.Ф.Дубовик. – Москва: Олимп-Бизнес, 2019. – 552 с.
9. Смольский, А.П. Деловой менеджмент : учебно-практическое пособие / А.П. Смольский. – Москва: Современная школа, 2011. – 230 с.
10. Тихомирова, О.Г. Управление проектом: комплексный подход и системный анализ : монография / О.Г. Тихомирова . – Москва : ИНФРА-М, 2018 . – 300 с.
11. Трофимова, Л. А. Методы принятия управленческих решений: учебник / Л.А. Трофимова. – Москва: ЮРАЙТ, 2013. – 335.
12. Управление проектами : учебник и практикум для студентов вузов, обучающихся по экономическим направлениям и специальностям / под общ. ред. Е.М. Роговой [и др.] . – Москва : ЮРАЙТ, 2020 . – 383 с.
13. Устинович, И. В. Бизнес-планирование. Практикум : учебное пособие для студентов вузов по специальности "Бизнес-администрирование" / И.В.Устинович, С.В. Шевченко, А.Л. Ивашутин, Министерство образования Республики Беларусь . – Минск : РИВШ, 2020 . – 162 с.
14. Фролов, Ю. В. Стратегический менеджмент. Формирование стратегии и проектирование бизнес-процессов : учебное пособие для студентов вузов / Ю.В. Фролов, Р.В. Серышев ; ред. Ю. В. Фролов . – 2-е изд., испр. и доп . – Москва : ЮРАЙТ, 2021 . – 154 с.

15. Чараева, М.В. Инвестиционное бизнес-планирование : учебное пособие для студентов вузов, обучающихся по направлению подготовки "Экономика (квалификация (степень) "бакалавр") / М.В.Чараева, Г. М.Лапицкая, Н. В.Крашенникова . – Москва : Альфа-М : ИНФРА-М, 2022 . – 176 с.

16. Черняк, В. З. Управление инвестиционными проектами: учебник / В.З. Черняк. – Москва: Юнити-ДАНА, 2004. – 365 с.

Дополнительная

1. Архипенков, С. Лекции по управлению программными проектами: учебное пособие / С. Архипенков. – Москва, 2009. – 128 с. https://ita.sibsutis.ru/sites/csc.sibsutis.ru/files/courses/trpo/sw_project_management.pdf

2. Беляцкий, Н.П. Менеджмент. Основы лидерства: учебное пособие / Н.П. Беляцкий. – Минск: Новое знание, 2002. – 250 с.

3. Брукс, Ф.П. Как проектируются и создаются программные комплексы : мифич.человеко-месяц : очерки по систем.программир. / Ф.П. Брукс. – Москва: Наука, 1979. – 304 с.

4. Вольфсон Б. Гибкие методологии разработки. [Электронный ресурс] // URL: http://agilerussia.ru/methodologies/borisvolfson_ebook/

5. Гибкая методология разработки программного обеспечения : курс лекций [электронный ресурс] Режим доступа: http://www.intuit.ru/studies/courses/583/439/info

6. Ильин В., Руководство качеством проектов. Практический опыт / В. Ильин. – СПб.: Вершина, 2006.

7. Инвестиционный менеджмент: практикум : учебное пособие. / В.И. Маколов [и др.] – Москва : КНОРУС, 2012. — 176 с.

8. Управление проектами: Учебное пособие / М.В. Романова. - М.: ИД ФОРУМ: НИЦ ИнфраМ, 2013 http://znanium.com/bookread.php?book=391146

9. Грекул, В.И. Проектное управление в сфере информационных технологий / Грекул В.И., Коровина Н.Л., Куприянов Ю.В. – Из-во «Бином». – 2020. – С.339. https://znanium.com/read?id=358779

10. Управление проектами. Учебник и практикум для академического бакалавриата / Балашов А.И., Рогова Е.М., Тихонова М.В., Ткаченко Е.А. – М.:Юрайт. – 2020. — 383 с. https://urait.ru/book/upravlenie-proektami-431784

11. Фатхутдинов, Р. А. Инновационный менеджмент: учебник для вузов / Р.А. Фатхутдинов. – Москва: ЗАО Бизнес-школа Интел-Синтез, 1998. – 272 с.

12. Филлипс, Д. Управление проектами в области информационных технологий / Д.Филлипс. – Из-во «Лори». – 2018. – С. 400. https://www.ozon.ru/product/upravlenie-proektami-v-oblasti-informatsionnyh-tehnologiy-146074787/?sh=_37R5Vlz

ПРОТОКОЛ СОГЛАСОВАНИЯ УЧЕБНОЙ ПРОГРАММЫ ПО ИЗУЧАЕМОЙ УЧЕБНОЙ ДИСЦИПЛИНЫ С ДРУГИМИ ДИСЦИПЛИНАМИ СПЕЦИАЛЬНОСТИ

Название учебной дисциплины с которой требуется согласование	Название кафедры	Предложения об изменениях в содержании учебной программы учреждения высшего образования по учебной дисциплине	Решение, принятое кафедрой, разработавшей учебную программу (с указанием даты и номера протокола)
Проектирование программных систем	Кафедра математических проблем управления и информатике	При изучении тем дисциплины «Проектирование программных систем», опираться на знания, полученные на занятиях по дисциплине «Менеджмент программного обеспечения»	Протокол № 9 от 23.04.2021

Заведующий кафедрой
к.т.н., доцент _____ Л.Н.Марченко