Интернет Образование
Орловский Региональный Центр

Ужаринский А.Ю.

Управление программными проектами: лабораторный практикум

г. Орел

ББК 32.97

У33

Ужаринский А.Ю. Управление программными проектами: лабораторный практикум / А.Ю. Ужаринский / Учебное электронное издание. — Орел: АНО "Центр интернет-образования", 2015. — 74 с.

Методические указания содержат описание выполнения лабораторных работ по применению теоретических знаний студентов, обучающихся по укрупненной группе направлений "Информатика и вычислительная техника", по управлению реальными программными проектами.

Содержание

Введение

Данные методические указания содержат лабораторные работы по применению теоретических знаний студентов по управлению реальными программными проектами. В качестве процесса разработки программного обеспечения, которым будет производиться управление, необходимо использовать разработку и внедрение программного продукта, разрабатываемого в рамках выпускной квалификационной работы .

Лабораторная работа 1

Тема: Разработка модели VISION видения программного проекта по методике RUP.

Теоретический материал

Одной из наиболее распространенных является унифицированный процесс управления проектами фирмы IBM Rational Unified Process (RUP). Он был разработан Филиппом Крачтеном (Philippe Kruchten), Иваром Якобсоном (Ivar Jacobson) и другими сотрудниками компании «Rational Software» в качестве дополнения к языку моделирования UML. RUP опирается на проверенные практикой методы анализа, проектирования и разработки программного обеспечения, методы управления проектами. Процесс обеспечивает прозрачность и управляемость процесса и позволяет создавать продукты в соответствии с требованиями заказчика на момент ее сдачи, а также в соответствии с возможностями инструментальных средств поддержки разработки.

В тоже время, 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.

В основе RUP лежат следующие принципы:

1) ранняя идентификация и непрерывное (до окончания проекта) устранение основных рисков;
2) концентрация на выполнении требований заказчиков к исполняемой программе (анализ и построение модели прецедентов (вариантов использования));

3) ожидание изменений в требованиях, проектных решениях и реализации в процессе разработки;
4) компонентная архитектура, реализуемая и тестируемая на ранних стадиях проекта;
5) постоянное обеспечение качества на всех этапах разработки проекта (продукта);
6) работа над проектом в сплочённой команде, ключевая роль в которой принадлежит архитекторам.

Для определения целей и границ проекта в подходе RUP используется документ «Видение проекта» (Vision) — это краткое описание сути будущего продукта. В нем кратко описывается, что это за продукт, каковы цели и задачи его создания, кто его пользователи и каковы основные возможности будущей системы. Видение позволяет с помощью нескольких абзацев ознакомить с сутью проекта любое заинтересованное лицо, собрать основные требования и дать общее представление о целях, которые поставлены перед продуктом.

Видение проекта разрабатывается на основе анализа потребностей бизнеса. Главная функция документа – подтверждение и согласование единого видения целей, задач и результатов всеми участниками проекта. Видение определяет что и зачем делается в проекте. Видение проекта это ключевой документ, который используется для принятия решений в ходе всего проекта, а также на фазе приемки — для подтверждения результата[9].

Видение программного продукта (Vision) предполагает ответ на пять основных вопросов:

- кто будет приобретать и использовать продукт (целевая аудитория);
- какие нужды пользователей продукт удовлетворяет;
- каковы критические характеристики продукта, чтобы удовлетворить эти нужды и сделать продукт успешным;

- чем продукт похож на существующие аналоги и чем он от них отличается;
- какой предполагается срок и бюджет для создания продукта.

Форма документа "Vision" по методике Rational Unified Process (RUP) содержит следующие основные разделы:

- Введение — описываются цель документа, его контекст, основные определения, акронимы и сокращения, даются ссылки на другие документы и характеризуется содержание документа.
- Раздел "позиционирование" содержит определение решаемой проблемы, указывается целевой заказчик продукта и исследуются преимущества изделия перед аналогами на рынке.
- Описание совладельцев и пользователей продукта, демография рынка и репутация разработчика.
- Краткий обзор изделия, описание его перспектив и ключевых возможностей в среде применения, предположения и зависимости, оценка стоимости разработки (калькуляция), вопросы лицензирования и инсталляции. Основные возможности продукта в развёрнутом изложении.
- Ограничения технические, технологические и прочие обстоятельства, которые необходимо учитывать при создании продукта.
- Показатели качества, включая показатели эффективности, надежности, отказоустойчивости и проч.
- Приоритеты реализации требований по степени важности, очередности реализации и т.п.
- Прочие требования к продукту - применяемые стандарты, системные требования, эксплуатационные требования, требования к окружающей среде.

– Требования к документации - содержание руководства пользователя, интерактивной справки, руководства по установке и конфигурированию и прочих документов, сопровождающих продукт. Приложения могут содержать расчёты и подтверждения характеристик проекта: статус, выгода, объем работ, риск, стабильность, назначение, причина создания продукта и проч.

Задание на лабораторную работу

Для программного продукта, представляемого в ВКР, составить развёрнутое описание видения продукта. Общий объём документа не должен превышать 4-6 листов.

Ход работы:

1. Проанализировать потребности бизнеса, рассматриваемого в впускной квалификационной работе.

2. Создать документ "Vision" по методике Rational Unified Process (RUP).

Контрольные работы:

Лабораторная работа 2

Тема: Разработка концепции проекта.

Теоретический материал

Концепция проекта разрабатывается на основе анализа потребностей бизнеса. Главная функция документа - подтверждение и согласование единого видения целей, задач и результатов всеми участниками проекта. Концепция определяет что и зачем делается в проекте.

Концепция проекта это ключевой документ, который используется для принятия решений в ходе всего проекта, а также на фазе приемки - для подтверждения результата. Она содержит, как правило, следующие разделы:

1. Название проекта
2. Цели проекта
3. Результаты проекта
4. Допущения и ограничения
5. Ключевые участники и заинтересованные стороны
6. Ресурсы проекта
7. Сроки
8. Риски
9. Критерии приемки
10. Обоснование полезности проекта

Цели и результаты проекта

Цели проекта должны отвечать на вопрос, зачем данный проект нужен. Цели проекта должны описывать бизнес-потребности и задачи, которые решаются в результате исполнения проекта. Целями проекта могут быть:

– Изменения в Компании. Например, автоматизация ряда бизнеспроцессов для повышения эффективности основной производственной деятельности.

- Реализация стратегических планов. Например, завоевание значительной доли растущего рынка за счет вывода на него нового продукта.
- Выполнение контрактов. Например, разработка программного обеспечения по заказу.
- Разрешение специфических проблем. Например, доработка программного продукта в целях приведения его в соответствие с изменениями в законодательстве.

Цели должны быть значимыми (направленными на достижение стратегических целей Компании), конкретными (специфичными для данного проекта), измеримыми (т.е иметь проверяемые количественные оценки), реальными (достижимыми). Четкое определение бизнес-целей важно, поскольку существенно влияет на все процессы и решения в проекте. Проект должен быть закрыт, если признается, что достижение цели невозможно или стало нецелесообразным. Например, если реальные затраты на проект будут превосходить будущие доходы от его реализации.

Результаты проекта отвечают на вопрос, что должно быть получено после его завершения. Результаты проекта должны определять:

- Какие именно бизнес-выгоды получит заказчик в результате проекта.
- Какой продукт или услуга. Что конкретно будет произведено по окончании проекта.
- Высокоуровневые требования. Краткое описание и при необходимости ключевые свойства и/или характеристики продукта/услуги.

Следует помнить, что результаты проекта должны быть измеримыми. Это означает, что при оценке результатов проекта должна иметься возможность сделать заключение достигнуты оговоренные в концепции результаты или нет.

Допущения и ограничения

Данный раздел описывает исходные допущения и ограничения. Допущения, как правило, тесно связаны с управлением рисками, о котором мы будем говорить далее. В разработке ПО часто приходится формулировать риски в виде допущений, тем самым передавая его заказчику. Например, оценивая проект разработки и внедрения по схеме с фиксированной ценой, мы должны записать в допущения предположение о том, что стоимость лицензий на стороннее ПО не изменится, до завершения проекта.

Ограничения, как правило, сокращают возможности проектной команды в выборе решений. В частности они могут содержать:

- Специфические нормативные требования. Например, обязательная сертификация продукта, услуги на соответствие определенным стандартам.
- Специфические технические требования. Например, разработка под заданную программно-аппаратную платформу.
- Специфические требования к защите информации.

В этом разделе также уместно сформулировать те требования к системе, которые могут ожидаться заказчиком по умолчанию, но не включаются в рамки данного проекта. Например, в данный раздел может быть включен пункт о том, что разработка программного интерфейса (API) для будущей интеграции с другими системами заказчика не входит в задачи данного проекта.

Ключевые участники и заинтересованные стороны

Одна из задач фазы инициации проекта это выявить и описать всех его участников. К участникам проекта относятся все заинтересованные стороны (stakeholders), лица и организации, например заказчики, спонсоры, исполняющая организация, которые активно участвуют в проекте или чьи интересы могут быть затронуты при исполнении или завершении проекта. Участники также могут влиять на проект и его результаты поставки.

К ключевым участникам программного проекта, как правило, относятся:

- Спонсор проекта — лицо или группа лиц, предоставляющая финансовые ресурсы для проекта в любом виде.
- Заказчик проекта — лицо или организация, которые будут использовать продукт, услугу или результат проекта. Следует учитывать, что заказчик и спонсор проекта не всегда совпадают.
- Пользователи результатов проекта.
- Куратор проекта — представитель исполнителя, уполномоченный принимать решение о выделении ресурсов и изменениях в проекте.
- Руководитель проекта — представитель исполнителя, ответственный за реализацию проекта в срок, в пределах бюджета и с заданным качеством.
- Соисполнители проекта. Субподрядчики и поставщики.

Ресурсы

Для того чтобы понять, сколько будет стоить реализация программного проекта, требуется определить и оценить ресурсы необходимые для его выполнения:

- Людские ресурсы и требования к квалификации персонала.
- Оборудование, услуги, расходные материалы, лицензии на ПО, критические компьютерные ресурсы.
- Бюджет проекта. План расходов и, при необходимости, предполагаемых доходов проекта с разбивкой по статьям и фазам/этапам проекта.

Специфика программного проекта заключается в том, что людские ресурсы вносят основной вклад в его стоимость. Все остальные затраты, как правило, незначительны, по сравнению с этим расходами. О том, как следует подходить к оценкам трудозатрат на реализацию проекта разработки ПО, мы будем подробно говорить в следующих лекциях. На фазе инициации хорошей считается оценка трудозатрат с точностью от -50% до +100%.

Необходимо помнить, что помимо непосредственно программирования в проекте разработки ПО есть много других процессов, которые требуют ресурсы соответствующей квалификации, а само программирование составляет лишь четверть всех затрат. Поэтому, если по вашей оценки для реализации требуемой функциональности в проекте необходимо написать 10 KSLOC (тысяч строк исходного программного кода), а ваши программисты пишут в среднем по 100 SLOC в день, то общие трудозатраты на проект будут не 100 чел.*дней, а не менее чем 400 чел.*дней. Остальные ресурсы потребуются на анализ и уточнение требований, проектирование, документирование, тестирование и другие проектные работы.

Сроки

Ф. Брукс писал: «Чтобы родить ребенка требуется девять месяцев независимо от того, сколько женщин привлечено к решению данной задачи. Многие задачи программирования относятся к этому типу, поскольку отладка по своей сути носит последовательный характер».

Там же Брукс приводит исключительно полезную, но почему-то редко применяемую, эмпирическую формулу оценки срока проекта по его трудоемкости. Формула была выведена Барии Боэмом (Barry Boehm) на основе анализа результатов 63 проектов разработки ПО, в основном в аэрокосмической области. Согласно этой формуле, для проекта, общая трудоемкость которого составляет N ч.*м. (человеко-месяцев), пожно утверждать что:

- Существует оптимальное, с точки зрения затрат, время выполнения графика для первой поставки: T = 2,5 (N ч.*м.)^1/3. То есть оптимальное время в месяцах пропорционально кубическому корню предполагаемого объема работ в человеко-месяцах. Следствием является кривая, дающая оптимальную численность проектной команды.

– Кривая стоимости медленно растет, если запланированный график длиннее оптимального. Работа занимает все отведенное для нее время.
– Кривая стоимости резко растет, если запланированный график короче оптимального. Практически ни один проект невозможно завершить быстрее, чем за ¾ расчетного оптимального графика вне зависимости от количества занятых в нем! (Рисунок 1)

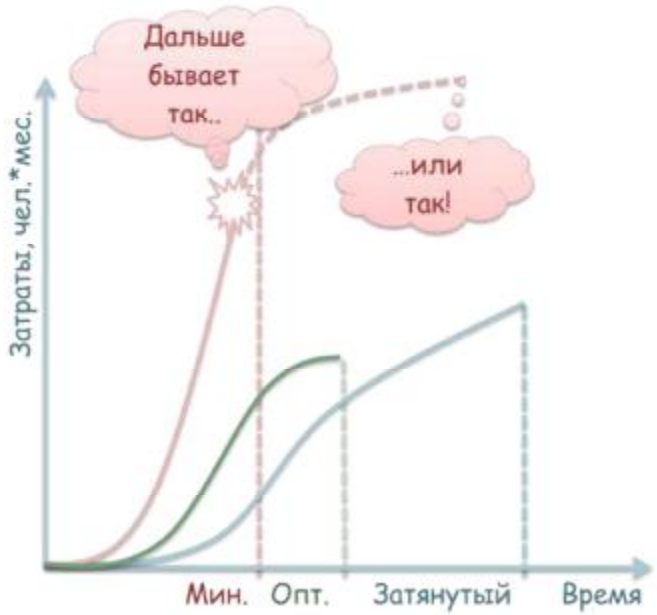

Рисунок 1 – Следствие закона Б. Боэма

Этот примечательный результат дает менеджеру программного проекта солидное подкрепление, когда высшее руководство требует принятия невозможного графика.

Для сколь-нибудь серьезного программного проекта недостаточно определить только срок его завершения. Необходимо еще определить его этапы — контрольные точки, в которых будет происходить переоценка проекта на основе реально достигнутых показателей.

Контрольная точка — важный момент или событие в расписании проекта, отмечающее достижение заданного результата и/или начало /

завершение определенного объема работы. Каждая контрольная точка характеризуется датой и объективными критериями ее достижения.

Как мы говорили ранее, современный проект разработки ПО должен реализовываться с применением инкрементального процесса. В этом случае контрольные точки должны соответствовать выпуску каждой промежуточной версии ПО, в которой будет реализована и протестирована определенная часть конечной функциональности программного продукта. В зависимости от сложности и масштаба проекта продолжительность одной итерации может составлять от 2 до 8 недель.

Риски

Риск – неопределенное событие или условие, наступление которого отрицательно или положительно сказывается на целях проекта.

Как правило, в случае возникновения негативного риска, почти всегда стоимость проекта увеличивается и происходит задержка в выполнении мероприятий, предусмотренных расписанием проекта. Управлению рисками проекта будет посвящена отдельная лекция.

На этапе инициации, когда нет необходимых данных для проведения детального анализа, часто приходится ограничиваться качественной оценкой общего уровня рисков: низкий, средний, высокий.

Критерии приемки

Критерии приемки должны определять числовые значения характеристик системы, которые должны быть продемонстрированы по результатам приемосдаточных испытаний или опытной эксплуатации и однозначно свидетельствовать о достижении целей проекта.

Обоснование полезности проекта

Этот раздел концепции должен содержать краткое технико-экономическое обоснование проекта:

- Для кого предназначены результаты проекта.
- Описание текущей ситуации «As Is». Какие у потенциального заказчика существуют проблемы.

- Каким образом результаты проекта решают эти проблемы («To Be»).
- Насколько значимо для клиента решение данных проблем (оценка экономического эффекта).
- Какие преимущества в итоге из этого может извлечь компания исполнитель проекта.

Пример концепции

В качестве примера, который позволит иллюстрировать теоретическое изложение основ управления проектами, возьмем реальный проект разработки ПО для автоматизации одного из подразделений крупной производственной компании. Назовем его «Автоматизированная система продажи документации».

Краткая легенда проекта. Заказчик ОАО «XYZ» является одним из ведущих производителей сложных технических изделий. Отдел «123», входящий в ОАО «XYZ», отвечает за продажу дополнительной сопроводительной документации для клиентов ОАО.

Дополнительная документация не входит в стандартную поставку, поскольку владелец этого технического изделия не всегда сам его эксплуатирует, а передает в эксплуатацию другой компании, которая становится клиентом «XYZ», и закупает у нее эксплуатационную документацию. Ремонт и техобслуживание конкретного изделия может выполнять третья компания, которой уже потребуется детальная техническая документация по ремонту и обслуживанию. Она также становится клиентом «XYZ» и закупает у нее требуемую продукцию.

Основная функция отдела «123» - получение и обработка заказов на дополнительную документацию, согласно ежегодно рассылаемому каталогу. В связи с переездом отдела «123» в новое здание, была поставлена задача на разработку и поставку системы, автоматизирующей основную деятельность отдела «123».

1. Цели и результаты проекта

1.1. Целью проекта является повышение эффективности основной производственной деятельности отдела «123».

1.2. Дополнительными целями проекта являются:

1.2.1. Установление долгосрочных отношений с важным заказчиком ОАО «XYZ».

1.2.2. Выход на новый перспективный рынок современных B2C систем.

2. Результаты проекта должны обеспечить:

2.1. Снижение затрат на обработку заявок.

2.2. Снижение сроков обработки заявок.

2.3. Повышение оперативности доступа к информации о наличии продукции.

2.4. Повышение оперативности доступа к информации о прохождении заявок.

2.5. Повышение надежности и полноты хранения информации о поступивших заявках и результатах их обработки.

3. Продуктами проекта являются:

3.1. Прикладное ПО и документация пользователей.

3.2. Базовое ПО.

3.3. Оборудование ЛВС, рабочие станции, сервера и операционно-системное ПО.

3.4. Проведение пуско-наладочных работ и ввод в опытную эксплуатацию.

3.5. Обучение пользователей и администраторов системы.

3.6. Сопровождение системы на этапе опытной эксплуатации.

3.7. Передача системы в промышленную эксплуатацию.

4. Система должна автоматизировать следующие функции:

4.1. Авторизация и аутентификация пользователей.

4.2. Просмотр каталога продуктов.

4.3. Поиск продуктов по каталогу.

4.4. Заказ выбранных продуктов.

4.5. Просмотр информации о статусе заказа.

4.6. Информирование клиента об изменении статуса заказа.

4.7. Просмотр и обработка заказов исполнителями из службы продаж.

4.8. Просмотр статистики поступления и обработки заказов за период.

4.9. Подготовка и сопровождение каталога продукции.

5. Допущения и ограничения

5.1. Проектирование прикладного ПО выполняется с использованием UML1.

5.2. Средством разработки ПО является Symantec Visual Cafe for Java2.

5.3. В качестве промежуточного ПО сопровождения и поддержки каталога используется ОО БД «Poet»3.

5.4. Нагрузка на систему не должна быть более 100 одновременно работающих пользователей.

5.5. В рамки проекта не входят:

5.5.1. Защита системы от преднамеренного взлома.

5.5.2. Разработка B2B API и интеграция с другими системами.

6. Ключевые участники и заинтересованные стороны

6.1. Спонсор проекта - директор Департамента информатизации ОАО «XYZ» В.Васильев.

6.2. Заказчик – начальник Отдела «123» Ф.Федотов

6.3. Пользователи автоматизированной системы:

6.4. Клиенты ОАО «XYZ» (поиск и заказ документации).

6.5. Руководство ОАО «XYZ» (анализ деятельности Отдела «123»).

6.6. Сотрудники производственных департаментов ОАО «XYZ» (сопровождение каталога).

6.7. Сотрудники Отдела «123» (обработка заявок и поставка документации).

6.8. Сотрудники департамента информатизации ОАО «XYZ» (администрирование системы).

6.9. Куратор проекта - начальник отдела заказных разработок И.Иванов.

6.10. Руководитель проекта - ведущий специалист отдела заказных разработок МП П.Петров.

7. Соисполнители:

7.1. Поставщик оборудования и операционно-системного ПО - ООО «Альфа».

7.2. Поставщик базового ПО - ООО «Бета».

8. Ресурсы проекта

8.1. Требования к персоналу

8.1.1. 1 - руководитель проекта,

8.1.2. 1 - технический лидер (архитектура, проектирование),

8.1.3. 1 - системный аналитик (требования, тест-дизайн, документирование),

8.1.4. 4 - программисты (с учетом работ по конфигурационному управлению),

8.1.5. 3 - тестировщика.

8.2. Материальные и другие ресурсы

8.2.1. Сервер управления конфигурациями и поддержки системы контроля версий

8.2.2. 2 серверных комплекса (для разработки и тестирования):

8.2.3. Сервер приложений с установленным BEA Weblogic AS

8.2.4. Сервер оперативной БД с установленной Oracle RDBMS

8.2.5. Сервер каталога с установленной OODB “Poet”

8.3. Лицензии на средства разработки и тестирования:

8.3.1. Oracle Designer – 1 лицензия

8.3.2. Symantec Visual Cafe for Java - 5 лицензий.

8.3.3. IBM Rational Test Robot (1 лицензия разработчика + неограниченная лицензия на клиент).

8.4. Расходная часть бюджета проекта4

8.4.1. Разработка и сопровождение прикладного ПО:

8.4.1.1. 9000 чел.*час. * $40 = $360 000

8.4.2. Поставка оборудования и операционно-системного ПО:

8.4.2.1. 3 сервера * $10 000 = $30 000

8.4.3. Поставка базового ПО:

8.4.3.1. BEA Weblogic AS $20 000

8.4.3.2. Oracle RDBMS $20 000

Итого: $430 000

9. Сроки проекта

9.1. 03.03 старт

9.2. 28.11 завершение

9.3. Контрольные точки:

9.3.1. 15.04 ТЗ утверждено

9.3.2. 30.04 1-я итерация завершена. Подсистема заказа документации передана в тестовую эксплуатацию (на серверах разработчика).

9.3.3. 15.05 Монтаж оборудования у заказчика завершен .

9.3.4. 30.05 Базовое ПО установлено у заказчика.

9.3.5. 15.06 2-я итерация завершена. Подсистема обработки заказов передана в тестовую эксплуатацию на оборудовании Заказчика

9.3.6. 02.09 3-я итерация завершена. Акт передачи системы в опытную эксплуатацию утвержден

9.3.7. 28.11 Система передана в промышленную эксплуатацию.

10. Риски проекта

10.1. Задачи системы поняты недостаточно полно. Понимание масштаба и рамок проекта недостаточно. Системы создаются на новой технологической платформе, сомнения в рыночной стабильности платформы. Суммарный уровень рисков следует оценить выше среднего.

11. Критерии приемки. По итогам опытной эксплуатации система должна продемонстрировать следующие показатели:

11.1. Средние затраты сотрудников Отдела «123» на регламентную обработку одного

заказа не превышают 4 чел.*час.

11.2. Срок регламентной обработки 1-го заказа не более 2 недель.

11.3. Время поиска и предоставления информации о наличии дополнительной документации не более 1 мин.

11.4. Время предоставления информации о сделанных заказах и истории их обработки не более 1 мин.

11.5. Система хранит всю информацию о сделанных заказах и истории их обработки.

11.6. Показатель доступности системы 98%.

12. Обоснование полезности проекта

12.1. Для Заказчика:

12.1.1. Повышение производительности обработки заказов в 2 раза.

12.1.1.1. “As Is”: 2500 заказов/год по 8 чел.*час.

12.1.1.2. “To Be”: 2500 заказов/год по 4 чел.*час.

12.1.1.3. Экономия: 2500 * 4 * $50 = $500 000 в год.

12.1.2. Повышение оперативности контроля

12.1.2.1. “As Is”: Ежемесячная отчетность.

12.1.2.2. “To Be”: Отчетность on-line.

12.1.3. Повышение удовлетворенности клиентов:

12.1.3.1. Сокращение срока обработки заказа в 2 раза.

12.1.3.2. Сокращение времени на поиск необходимой документации в 10 раз

12.1.3.3. Повышение оперативности обновления каталога 10 раз.

12.2. Для компании-исполнителя:

12.2.1. Высокая стратегическая ценность. Дает устойчивое увеличение рынка и завоевание нового рынка.

12.2.2. Финансовая ценность выше среднего. Ожидаемые доходы от проекта не менее чем в 1.3 раза превышают расходы.

Задание на лабораторную работу

В соответствии со сформированной моделью видения продукта, разработать концепцию проекта разработки программы для ВКР.

Ход работы:

Контрольные вопросы:

Лабораторная работа 3

Тема: Оценка трудоёмкости и экономической эффективности проекта

Теоретический материал

Оценку экономической эффективности проекта рекомендуется производить с учетом следующих показателей:

- чистый дисконтированный доход (ЧДД) и интегральный эффект;
- индекс доходности (ИД);
- внутренняя норма доходности (внд);
- срок окупаемости (СО);
- другие показатели, характеризующие интересы участников или специфики проекта.

Чистый дисконтированный доход (ЧДД)

Этот доход определяется как сумма текущих эффектов за весь расчетный период, приведенная к начальному шагу, как превышение интегральных результатов над интегральными затратами.

$$Э = ЧДД = \sum_{t=0}^{T} (R_t - З_t) \frac{1}{(1+d)^t}$$

где R_t – результаты достигаемые на t-ом шаге расчета;

$З_t$ – затраты, осуществляемые на том же шаге;

T – горизонт расчета (продолжительность расчетного периода) – равен номеру шага расчета на котором производится закрытие (ликвидация) проекта;

$Э = (R_t - З_t)$ – эффект достигаемый на t-ом шаге расчета;

D – постоянная норма дисконта, равна приемлемой для инвестора норме дохода на капитал.

Если **ЧДД**:

- **положителен** (ЧДД>0), проект является **эффективным**. Чем больше ЧДД, тем проект эффективнее;
- **ноль** (ЧДД=0) или **отрицателен** (ЧДД<0),то проект **неэффективен**.

Если проект будет принят, то инвестор понесет убытки.

Для определения ЧДД можно воспользоваться модифицированной методикой:

$$ЧДД = \sum_{t=0}^{T} \frac{Rt - 3t^{-}}{(1+d)^{t}} - K$$

где К – сумма дисконтированных капвложений.

Индекс доходности (ИД)

Индекс доходности (ИД) представляет собой отношение суммы приведенных эффектов к величине капиталовложений.

$$ИД = \frac{1}{K} \sum_{t=0}^{T} \frac{Rt - 3t^{-}}{(1+d)^{t}}$$

Индекс доходности тесно связан с ЧДД: если ЧДД положителен то ИД>1 и наоборот. Таким образом, если ИД>1, то проект эффективен, а если же ИД<1, то проект неэффективен.

В отличие от чистого дисконтированного дохода (ЧДД) индекс доходности (ИД) является относительным показателем. Благодаря этому он очень удобен при выборе одного проекта из ряда альтернативных, имеющих примерно одинаковые значения ЧДД, либо при комплектовании портфеля инвестиций с максимальным суммарным значением ЧДД.

Внутренняя норма доходности (ВНД).

Внутренняя норма доходности представляет собой норму дисконта (d), при которой величина приведенных эффектов равна приведенным капиталовложениям и определяется решением уравнения:

Если определение ЧДД инвестиционного проекта дает ответ на вопрос, является ли он эффективным или нет при некоторой заданной норме дисконта (d), то ВНД проекта определяется в процессе расчета и затем сравнивается с требуемой инвестором нормой дохода на вкладываемый капитал.

$$\sum_{t=0}^{T} \frac{Rt - 3t^{-}}{(1+d)^{t}} = \sum_{t=0}^{1} \frac{Kt}{(1+d)^{t}}$$

$$ВНД = d_1 + \frac{f(d_1)}{f(d_1) - f(d_2)} \times (d_2 - d_1),$$

где d_1 – значение дисконта, при котором ЧДД>0;

d_2 – значение дисконта, при котором ЧДД <0;

$f(d_1)$ – значение ЧДД при d1;

$f(d_2)$ – значение ЧДД при d2.

Срок окупаемости.

Срок окупаемости – это минимальный временной интервал (от начала осуществления проекта), за пределами которого интегральный эффект становится положительным и, в дальнейшем, остается таковым.

СО = К / Д

где К – единовременные капитальные затраты,

Д – ежегодный доход от капитала.

Показатель эффективности инвестиций.

Метод расчета коэффициента эффективности инвестиций, это, в принципе, то же самое, что и рентабельность инвестиций.

КЭИ = ЧП /(К – ЛС)

где ЧП – чистая прибыль (балансовая прибыль за вычетом отчислений в бюджет);

ЛС – ликвидационная стоимость проекта.

Лучше все–таки КЭИ определять с дисконтированием денежных потоков:

$$КЭИ = \frac{\sum_{t=1}^{T} ЧПt \times \frac{1}{(1+d)^{t}}}{\sum_{t=1}^{T} Kt \frac{1}{(1+d)^{t}}}$$

Для оценки трудоёмкости и сроков реализации проекта рекомендуется использовать одну из следующих методик:

1. Методика PERT.

2. Метод функциональных точек.
3. Методика COCOMO II.

Методика PERT

Инженерный метод оценки трудоемкости проекта PERT (Program Project Evaluation and Review Technique) был разработан в 1958 году в ходе проекта по созданию баллистических ракет морского базирования «Поларис». Входом для данного метода оценки служит список элементарных пакетов работ. Для инженерного подхода нет необходимости точно знать закон распределения нашей оценки трудоемкости каждого такого элементарного пакета. Диапазон неопределенности достаточно охарактеризовать тремя оценками:

M_i – наиболее вероятная оценка трудозатрат.

O_i – минимально возможные трудозатраты на реализацию пакета работ. Ни один риск не реализовался. Быстрее точно не сделаем. Вероятность такого, что мы уложимся в эти затраты, равна 0.

P_i – пессимистическая оценка трудозатрат. Все риски реализовались.

Оценку средней трудоемкости по каждому элементарному пакету можно определит по формуле:

$E_i = (P_i + 4\ M_i + O_i)/6$

Для расчета среднеквадратичного отклонения используется формула:

$СКО_i = (P_i - O_i)/6$

Если наши оценки трудоемкости элементарных пакетов работ статистически независимы, а не испорчены, например, необоснованным оптимизмом то, согласно центральной предельной теореме теории вероятностей суммарная трудоемкость проекта может быть рассчитана по формуле:

$$E = \sum E_i$$

А среднеквадратичное отклонение для оценки суммарной трудоемкости будет составлять:

$$\text{СКО} = \sqrt{\sum \text{СКО}_i^2}$$

Тогда для оценки суммарной трудоемкости проекта, которую мы не превысим с вероятностью 95%, можно применить формулу:

$E_{95\%} = E + 2 * СКО$

Это значит, что вероятность того, что проект превысит данную оценку трудоемкости составляет всего 5%. А это уже вполне приемлемая оценка, под которой может расписаться профессиональный менеджер.

Список элементарных пакетов работ, который используется при оценке трудоемкости, как правило, берется из нижнего уровня ИСР проекта. Но может быть использован и накопленный опыт аналогичных разработок.

Рассмотрим применение данного подхода на примере реального проекта. В Ассоциации CBOSS задачей проекта была разработка на основе стандартов J2EE общесистемного ПО для перевода рабочих мест CBOSS на новую трехзвенную архитектуру. Был разработан набор стандартных компонентов и сервисов, из которых как из конструктора можно эффективно и качественно собирать прикладные подсистемы. Высокоуровневая архитектура реализовывала стандартный паттерн MVC (Рисунок 2), каждый из компонентов которого имел «точки расширения» для прикладной разработки, которые на рисунке выделены красным светом.

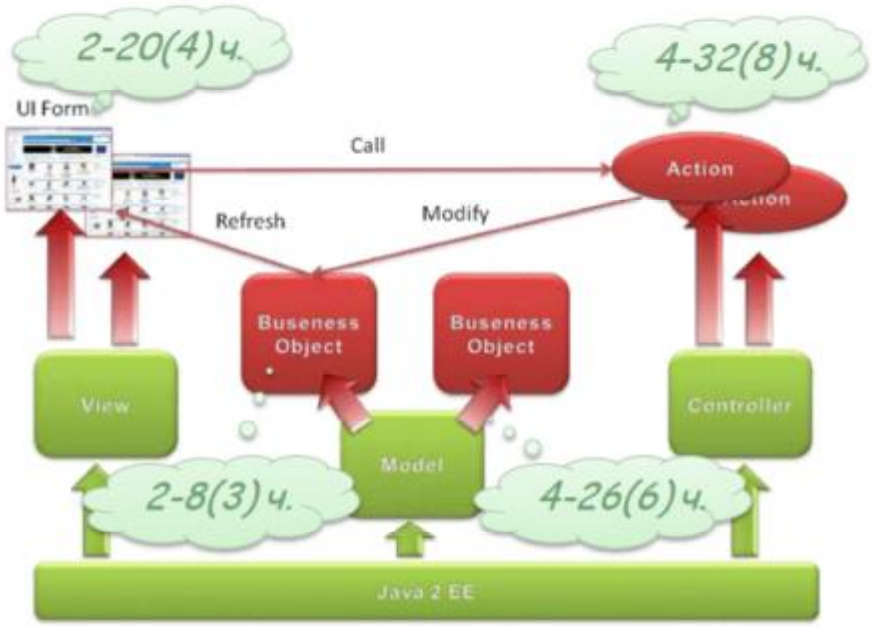

Рисунок 2 – Высокоуровневая архитектура фреймворка J2EE для разработки приложений

Такими точками расширения являлись:

- пользовательский экран (UI Form), который собирался из готовых визуальных компонентов;
- обработчики (Action), которые обрабатывали на сервере приложений события от активных визуальных компонентов, входящих в состав экрана;
- объекты (Business Obj), которые моделировали прикладную область, и к которым обращались обработчики событий.

Так вот, хотя все разрабатываемые рабочие места различались по функциональности и сложности, накопленная статистика фактических трудозатрат на разработку прикладных систем позволяла нам оценивать проекты разработки нового приложения достаточно быстро и с высокой достоверностью.

Согласно этой статистике, разработка и отладка требовала у программиста:

- для одного экрана - от 2 до 20 часов (наиболее вероятно – 4 часа);

- для одного обработчика событий - от 4 до 32 часов (наиболее вероятно – 8 часов);
- для нового бизнес-объекта - от 2 до 8 часов (наиболее вероятно – 3 часа);
- для добавление нового бизнес-метода - от 2 до 26 часов (наиболее вероятно – 6 часов).

Весь проект прикладной разработки измерялся в «попугаях»:

- KUI – количество пользовательских экранов.
- КАct – количество обработчиков событий.
- КВО – количество новых бизнес-объектов.
- КВМ – количество новых или модифицируемых бизнес-методов.

Если новое разрабатываемое приложение содержит 20 пользовательских экранов, 60 обработчиков событий, 16 новых бизнес-объекта и 40 новых бизнесметодов, которые необходимо добавить, как в новые, так и в уже существующие бизнес-объекты, тогда, согласно нашей статистике,

E_{UI} = (2 + 4*4 + 20) / 6 = 6.7 чел.*час., $СКО_{UI}$ = (20 - 2) / 6 = 3 чел.*час

E_{Act} = (4 + 4*8 + 32) / 6 = 11.3 чел.*час., $СКО_{Act}$ = (32 - 4) / 6 = 4.7 чел.*час

E_{BO} = (2 +4*3 + 8) / 6 = 3.7 чел.*час., $СКО_{BO}$ = (8 - 2) / 6 = 1 чел.*час

E_{BM} = (2 + 4*6 + 26) / 6 = 8.7 чел.*час., $СКО_{BM}$ = (26 - 2) / 6 = 4 чел.*час

Для средней трудоемкости работ по кодированию в проекте может быть получена следующая оценка:

E = 20 * 6.7 + 60 * 11.3 + 16 * 3.7 + 40 * 8.7 ≈ 1220 чел.*час.

$$\text{СКО} = \sqrt{20 * 3^2 + 60 * 4{,}7^2 + 16 * 1^2 + 40 * 4^2} = \sqrt{180 + 1325 + 16 + 640} \approx 46 \text{ чел} * \text{час}$$

Тогда для оценки суммарной трудоемкости проекта, которую мы не превысим с вероятностью 95%, получим

$E_{95\%}$ = 1220 + 2 * 46 ≈ 1300 чел.*час.

Хотя относительная погрешность в оценке трудоемкости каждой такой элементарной работы составляла десятки процентов, для нашего проекта, в котором было таких «попугаев» было 136, относительная погрешность оценки суммарной трудоемкости, сделанной по методу PERT, составила, приблизительно, лишь 4%.

Даже если у нас очень размытые оценки трудоемкости каждой из элементарных работ, но они независимы, то ошибки мы делаем как в меньшую, так и большую стороны. Поэтому при фактической реализации проекта эти ошибки будут компенсироваться, что позволяет нам оценить общие трудозатраты по проекту существенно точнее, чем трудозатраты на каждую элементарную работу. Но это утверждение будет справедливо только в том случае, если наша ИСР содержит все необходимые работы, которые должны быть выполнены для получения всех продуктов проекта.

Полученную оценку трудоемкости кодирования необходимо умножить на четыре, поскольку помним, что кодирование составляет только 25% общих трудозатрат проекта. Поэтому суммарная трудоемкость нашего проекта составит, приблизительно, 5200 чел.*час.

Как мы уже говорили ранее, если сотрудник на 100% назначен на проект, это, как правило, не означает, что он все 40 часов в неделю будет тратить на проектные работы. Тратить он будет 60 – 80% своего рабочего времени. Поэтому, в месяц сотрудник будет работать по проекту, примерно, 165 * 0.8 = 132 чел.*час/мес. Следовательно, трудоемкость проекта в человеко-месяцах составит, приблизительно 5200 / 132 ≈ 40.

Тогда согласно формуле Б. Боэма оптимальная продолжительность проекта составит:

T = 2.5 * (40) ^ 1/3 = 8.5 месяцев,

а средняя численность команды – 5 человек.

Метод функциональных точек

Анализ функциональных точек - стандартный метод измерения размера программного продукта с точки зрения пользователей системы. Метод

разработан Аланом Альбрехтом (Alan Albrecht) в середине 70-х. Метод был впервые опубликован в 1979 году. В 1986 году была сформирована Международная Ассоциация Пользователей Функциональных Точек (International Function Point User Group – IFPUG), которая опубликовала несколько ревизий метода [2].

Метод предназначен для оценки на основе логической модели объема программного продукта количеством функционала, востребованного заказчиком и поставляемого разработчиком. Несомненным достоинством метода является то, что измерения не зависят от технологической платформы, на которой будет разрабатываться продукт, и он обеспечивает единообразный подход к оценке всех проектов в компании.

При анализе методом функциональных точек надо выполнить следующую последовательность шагов:

1. Определение типа оценки.
2. Определение области оценки и границ продукта.
3. Подсчет функциональных точек, связанных с данными.
4. Подсчет функциональных точек, связанных с транзакциями.
5. Определение суммарного количества не выровненных функциональных точек (*UFP*).
6. Определение значения фактора выравнивания (*FAV*).
7. Расчет количества выровненных функциональных точек (*AFP*).

Определение типа оценки

Первое, что необходимо сделать, это определить тип выполняемой оценки. Метод предусматривает оценки трех типов:

1. Проект разработки. Оценивается количество функциональности поставляемой пользователям в первом релизе продукта.
2. Проект развития. Оценивается в функциональных точках проект доработки: добавление, изменение и удаление функционала.
3. Продукт. Оценивается объем уже существующего и установленного продукта.

Определение области оценки и границ продукта

Второй шаг – это определение области оценки и границ продукта. В зависимости от типа область оценки может включать:

- Все разрабатываемые функции (для проекта разработки)
- Все добавляемые, изменяемые и удаляемые функции (для проектов поддержки)
- Только функции, реально используемые, или все функции (при оценке продукта и/или продуктов).

Третий шаг. Границы продукта (Рисунок 3) определяют:

- Что является «внешним» по отношению к оцениваемому продукту.
- Где располагается «граница системы», через которую проходят транзакции передаваемые или принимаемые продуктом, с точки зрения пользователя.
- Какие данные поддерживаются приложением, а какие – внешние.

Рисунок 3 – Границы продукта в методе функциональных точек

К логическим данным системы относятся:

- Внутренние логические файлы (ILFs) – выделяемые пользователем логически связанные группы данных или блоки управляющей информации, которые поддерживаются внутри продукта.
- Внешние интерфейсные файлы (EIFs) - выделяемые пользователем логически связанные группы данных или блоки

управляющей информации, на которые ссылается продукт, но которые поддерживаются вне продукта.

Примером логических данных (информационных объектов) могут служить: клиент, счет, тарифный план, услуга.

Подсчет функциональных точек, связанных с данными

Третий шаг - подсчет функциональных точек, связанных с данными. Сначала определяется сложность данных по следующим показателям:

- DET (data element type) – неповторяемое уникальное поле данных, например, Имя Клиента – 1 DET; Адрес Клиента (индекс, страна, область, район, город, улица, дом, корпус, квартира) – 9 DET’s
- RET (record element type) – логическая группа данных, например, адрес, паспорт, телефонный номер.

Оценка количества не выровненных функциональных точек, зависит от сложности данных, которая определяется на основании матрицы сложности (Таблица 1).

Таблица 1 – Матрица сложности данных

	1 – 19 DET	20 – 50 DET	50+ DET
1 RET	Низкая	Низкая	Средняя
2 – 5 RET	Низкая	Средняя	Высокая
6+ RET	Средняя	Высокая	Высокая

Оценка данных в не выровненных функциональных точках (*UFP*) подсчитывается по-разному для внутренних логических файлов (*ILFs*) и для внешних интерфейсных файлов (*EIFs*) (Таблица 2) в зависимости от их сложности.

Таблица 2 – Оценка данных в не выровненных функциональных точках (*UFP*) для внутренних логических файлов (*ILFs*) и внешних интерфейсных файлов (*EIFs*).

Сложность данных	Количество UFP (ILF)	Количество UFP (EIF)
Низкая	7	5
Средняя	10	7
Высокая	15	10

Для иллюстрации рассмотрим пример оценки в не выровненных функциональных точках объекта данных «Клиент»

Объект «Клиент» содержит четыре логических группы данных, которые в совокупности состоят из 15 неповторяемых уникальное полей данных. Согласно матрице (Таблица 1), нам следует оценить сложность этого объекта данных, как «Low». Теперь, если оцениваемый объект относится к внутренним логическим файлам, то согласно Таблица 8 его сложность будет 7 не выровненных функциональных точек (*UPF*). Если же объект является внешним интерфейсным файлом, то его сложность составит 5 *UPF*.

Подсчет функциональных точек, связанных с транзакциями

Подсчет функциональных точек, связанных с транзакциями – это четвертый шаг анализа по методу функциональных точек.

Транзакция – это элементарный неделимый замкнутый процесс, представляющий значение для пользователя и переводящий продукт из одного консистентного состояния в другое.

В методе различаются следующие типы транзакций (Таблица 3):

- *EI* (external inputs) – внешние входные транзакции, элементарная операция по обработке данных или управляющей информации, поступающих в систему из вне.
- *EO* (external outputs) – внешние выходные транзакции, элементарная операция по генерации данных или

управляющей информации, которые выходят за пределы системы. Предполагает определенную логику обработки или вычислений информации из одного или более ILF.

– *EQ* (external inquiries) – внешние запросы, элементарная операция, которая в ответ на внешний запрос извлекает данные или управляющую информацию из *ILF* или *EIF*.

Таблица 3. Основные отличия между типами транзакций. Легенда: О – основная; Д – дополнительная; NA – не применима.

Функция	Тип транзакции		
	EI	EO	EQ
Изменяет поведение системы	О	Д	NA
Поддержка одного или более ILF	О	Д	NA
Представление информации пользователю	Д	О	О

Оценка сложности транзакции основывается на следующих ее характеристиках:

– *FTR* (file type referenced) – позволяет подсчитать количество различных файлов (информационных объектов) типа ILF и/или EIF модифицируемых или считываемых в транзакции.

– *DET* (data element type) – неповторяемое уникальное поле данных. Примеры. *EI*: поле ввода, кнопка. *EO*: поле данных отчета, сообщение об ошибке. *EQ*: поле ввода для поиска, поле вывода результата поиска.

Для оценки сложности транзакций служат матрицы, которые представлены в Таблице 4 и Таблице 5.

Таблица 4 – Матрица сложности внешних входных транзакций (EI)

EI	1 – 4 DET	5 – 15 DET	16+ DET
0 – 1 FTR	Низкий	Низкий	Средний
2 FTR	Низкий	Средний	Высокий
3+ FTR	Средний	Высокий	Низкий

Таблица 5 – Матрица сложности внешних выходных транзакций и внешних запросов (EO & EQ)

EO & EQ	1 – 5 DET	6 – 19 DET	20+ DET
0 – 1 FTR	Низкий	Низкий	Средний
2 – 3 FTR	Низкий	Средний	Высокий
4+ FTR	Средний	Высокий	Низкий

Оценка транзакций в не выровненных функциональных точках (*UFP*) может быть получена из матрицы (Таблица 6)

Таблица 6 – Сложность транзакций в не выровненных функциональных точках (*UFP*)

Сложность транзакций	Количество UFP (EO & EQ)	Количество UFP (EI)
Низкая	3	4
Средняя	4	5
Высокая	6	7

В качестве примера, рассмотрим оценку управляющей транзакции (*EI*) для диалогового окна, задающего параметры проверки орфографии в MS Office Outlook (Рисунок 4).

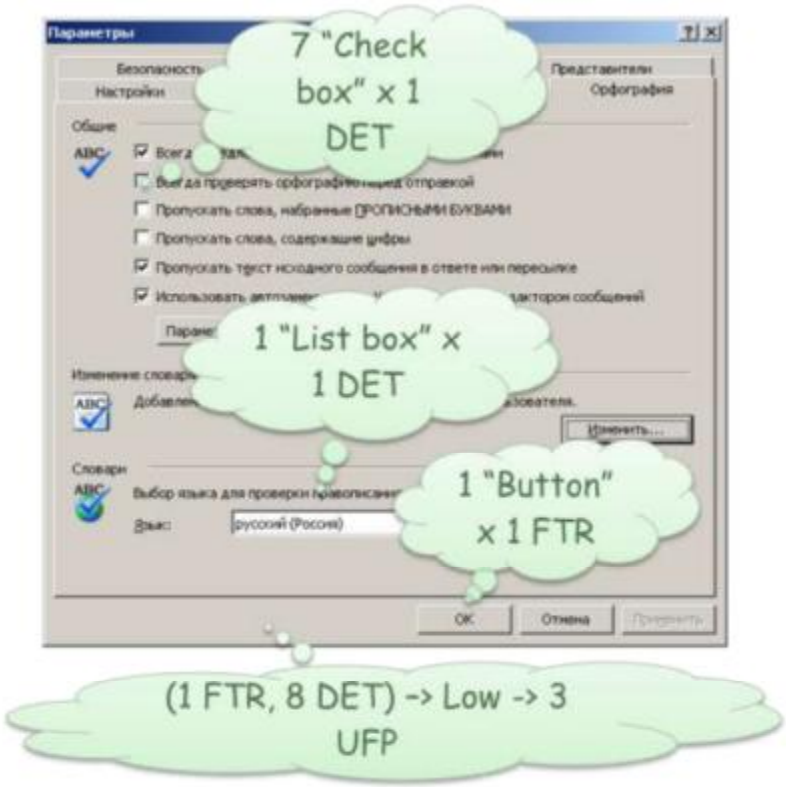

Рисунок 4 – Диалоговое окно, управляющее проверкой орфографии в MS Office Outlook

Каждый “Check box” оценивается, как 1 *DET*. Выпадающий список – 1 *DET*. Каждая управляющая кнопка должна рассматриваться как отдельная транзакция. Например, если оценивать управляющую транзакцию по кнопке «OK», то, для данной транзакции мы имеем 1 *FTR* и 8 *DET*. Поэтому, согласно матрице (Таблица 10), мы можем оценить сложность транзакции, как Low. И, наконец, в соответствие с матрицей (Таблица 12), данная транзакция должна быть оценена в 3 не выровненных функциональных точек (*UFP*).

Определение суммарного количества не выровненных функциональных точек (*UFP*)

Общий объем продукта в не выровненных функциональных точках (*UFP*) определяется путем суммирования по всем информационным объектам (*ILF*, *EIF*) и элементарным операциям (транзакциям EI, EO, EQ).

$$UFP = \sum_{ILF} UFP_i + \sum_{EIF} UFP_i + \sum_{EI} UFP_i + \sum_{EO} UFP_i + \sum_{EQ} UFP_i$$

Определение значения фактора выравнивания (*FAV*)

Помимо функциональных требований на продукт накладываются общесистемные требования, которые ограничивают разработчиков в выборе решения и увеличивают сложность разработки. Для учета этой сложности применяется фактор выравнивания (*VAF*). Значение фактора *VAF* зависит от 14 параметров, которые определяют системные характеристики продукта:

1. Обмен данными (0 – продукт представляет собой автономное приложение; 5 – продукт обменивается данными по более, чем одному телекоммуникационному протоколу).

2. Распределенная обработка данных (0 – продукт не перемещает данные; 5 – распределенная обработка данных выполняется несколькими компонентами системы).

3. Производительность (0 – пользовательские требования по производительности не установлены; 5 – время отклика сильно ограничено критично для всех бизнес-операций, для удовлетворения требованиям необходимы специальные проектные решения и инструменты анализа.

4. Ограничения по аппаратным ресурсам (0 – нет ограничений; 5 – продукт целиком должен функционировать на определенном процессоре и не может быть распределен).

5. Транзакционная нагрузка (0 – транзакций не много, без пиков; 5 – число транзакций велико и неравномерно, требуются специальные решения и инструменты).

6. Интенсивность взаимодействия с пользователем (0 – все транзакции обрабатываются в пакетном режиме; 5 – более 30% транзакций – интерактивные).

7. Эргономика (эффективность работы конечных пользователей) (0 – нет специальных требований; 5 – требования по эффективности очень жесткие).

8. Интенсивность изменения данных (ILF) пользователями (0 – не требуются; 5 – изменения интенсивные, жесткие требования по восстановлению).

9. Сложность обработки (0 – обработка минимальна; 5 – требования безопасности, логическая и математическая сложность, многопоточность).

10. Повторное использование (0 – не требуется; 5 – продукт разрабатывается как стандартный многоразовый компонент).

11. Удобство инсталляции (0 – нет требований; 5 – установка и обновление ПО производится автоматически).

12. Удобство администрирования (0 – не требуется; 5 – система автоматически самовосстанавливается).

13. Портируемость (0 – продукт имеет только 1 инсталляцию на единственном процессоре; 5 – система является распределенной и предполагает установку на различные «железо» и ОС).

14. Гибкость (0 – не требуется; 5 – гибкая система запросов и построение произвольных отчетов, модель данных изменяется пользователем в интерактивном режиме).

14 системных параметров (degree of influence, *DI*) оцениваются по шкале от 0 до 5. Расчет суммарного эффекта 14 системных характеристик (total degree of influence, *TDI*) осуществляется простым суммированием:

$$TDI = \sum DI$$

Расчет значения фактора выравнивания производится по формуле

$VAF = (TDI * 0.01) + 0.65$

Например, если, каждый из 14 системных параметров получил оценку 3, то их суммарный эффект составит $TDI = 3 * 14 = 42$. В этом случае значение фактора выравнивания будет: $VAF = (42 * 0.01) + 0.65 = 1.07$

Расчет количества выровненных функциональных точек (AFP)

Дальнейшая оценка в выровненных функциональных точках зависит от типа оценки. Начальная оценка количества выровненных функциональных точек для программного приложения определяется по следующей формуле:

AFP = UFP * VAF.

Она учитывает только новую функциональностсть, которая реализуется в продукте. Проект разработки продукта оценивается в DFP (development functional point) по формуле:

$DFP = (UFP + CFP) * VAF$,

где *CFP* (conversion functional point) – функциональные точки, подсчитанные для дополнительной функциональности, которая потребуется при установке продукта, например, миграции данных.

Проект доработки и совершенствования продукта оценивается в EFP (enhancement functional point) по формуле:

EFP = (ADD + CHGA + CFP) * VAFA + (DEL* VAFB),

где

ADD – функциональные точки для добавленной функциональности;

CHGA – функциональные точки для измененных функций, рассчитанные после модификации;

VAFA – величина фактора выравнивания рассчитанного после завершения проекта;

DEL – объем удаленной функциональности;

VAFB – величина фактора выравнивания рассчитанного до начала проекта.

Суммарное влияние процедуры выравнивания лежит в пределах +/- 35% относительно объема рассчитанного в *UFP*.

Метод анализа функциональных точек ничего не говорит о трудоемкости разработки оцененного продукта. Вопрос решается просто, если компания разработчик имеет собственную статистику трудозатрат на реализацию функциональных точек.

Методика COCOMO II

Методика COCOMO позволяет оценить трудоемкость и время разработки программного продукта. Впервые была опубликована Бари Боэмом [3] в 1981 году в виде результат анализа 63 проектов компании «TRW Aerospace». В 1997 методика была усовершенствована и получила название COCOMO II. Калибровка параметров производилась по 161 проекту разработки. В модели используется формула регрессии с параметрами, определяемыми на основе отраслевых данных и характеристик конкретного проекта.

Различаются две стадии оценки проекта: предварительная оценка на начальной фазе и детальная оценка после проработки архитектуры.

Формула оценки трудоемкости проекта в чел.*мес. имеет вид:

$$PM = A \times SIZE^E \times \prod_{i=1}^{n} EM_i$$

A = 2,94

$$E = B + 0{,}01 \times \sum_{j=1}^{5} SF_j$$

B = 0,91

где SIZE – размер продукта в KSLOC

EM_i – множители трудоёмкости

SF_j – факторы масштаба

n = 7 – для предварительной оценки

n = 17 – для детальной оценки

Главной особенностью методики является то, что для того, чтобы оценить трудоемкость, необходимо знать размер программного продукта в тысячах строках исходного кода (KSLOC, Kilo Source Lines Of Code). Размер программного продукта может быть, например, оценен экспертами с применением метода PERT.

Факторы масштаба.

В методике используются пять факторов масштаба SFj, которые определяются следующими характеристиками проекта:

1. PREC – прецедентность, наличие опыт аналогичных разработок (Very Low – опыт в продукте и платформе отсутствует; Extra High – продукт и платформа полностью знакомы)

2. FLEX – гибкость процесса разработки (Very Low – процесс строго детерминирован; Extra High – определены только общие цели).

3. RESL – архитектура и разрешение рисков (Very Low – риски неизвестны/не проанализированы; Extra High – риски разрешены на 100%)

4. TEAM – сработанность команды (Very Low – формальные взаимодействия; Extra High – полное доверие, взаимозаменяемость и взаимопомощь).

5. PMAT – зрелость процессов (Very Low – CMM Level 1; Extra High – CMM Level 5

Значение фактора масштаб, в зависимости от оценки его уровня, приведены в Таблице 7

Таблица 7 – Значение фактора масштаба, в зависимости от оценки его уровня

Фактор масштаба	Оценка уровня фактора					
	Very Low	Low	Nominal	High	Very High	Extra High
PREC	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	5.07	4.05	3.04	2.03	1.01	0.00
RESL	7.07	5.65	4.24	2.83	1.41	0.00
TEA	5.48	4.38	3.29	2.19	1.10	0.00

M						
PM AT	7.80	6.24	4.68	3.12	1.56	0.00

Множители трудоемкости

В нашу задачу не входит детальное описание метода COCOMO II, поэтому мы рассмотрим только случай предварительной оценки трудоемкости программного проекта. Для этой оценки необходимо оценить для проекта уровень семи множителей трудоемкости Mi:

1. PERS – квалификация персонала (Extra Low – аналитики и программисты имеют низшую квалификацию, текучесть больше 45%; Extra High - аналитики и программисты имеют высшую квалификацию, текучесть меньше 4%)

2. RCPX – сложность и надежность продукта (Extra Low – продукт простой, специальных требований по надежности нет, БД маленькая, документация не требуется; Extra High - продукт очень сложный, требования по надежности жесткие, БД сверхбольшая, документация требуется в полном объеме)

3. RUSE – разработка для повторного использования (Low – не требуется; Extra High – требуется переиспользование в других продуктах)

4. PDIF – сложность платформы разработки (Extra Low – специальные ограничения по памяти и быстродействию отсутствуют, платформа стабильна; Extra High – жесткие ограничения по памяти и быстродействию, платформа нестабильна)

5. PREX – опыт персонала (Extra Low – новое приложение, инструменты и платформа; Extra High - приложение, инструменты и платформа хорошо известны)

6. FCIL – оборудование (Extra Low – инструменты простейшие, коммуникации затруднены; Extra High – интегрированные средства поддержки жизненного цикла, интерактивные мультимедиа коммуникации)

7. SCED – сжатие расписания (Very Low – 75% от номинальной длительности; Very High – 160% от номинальной длительности)

Влияние множителей трудоемкости в зависимости от их уровня определяется их числовыми значениями, которые представлены в матрице, приведенной ниже (Таблица 8).

Таблица 8 – Значения множителей трудоемкости, в зависимости от оценки их уровня

M_i	Оценка уровня множителя трудоемкости						
	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
PERS	2.12	1.62	1.26	1.00	0.83	0.63	0.5
RCPX	0.49	0.60	0.83	1.00	1.33	1.91	2.72
RUSE	n/a	n/a	0.95	1.001	.07	1.15	1.24
PDIF	n/a	n/a	0.87	1.00	1.29	1.81	2.61
PREX	1.59	1.33	1.22	1.00	0.87	0.74	0.62
FCIL	1.43	1.30	1.10	1.0	0.87	0.73	0.62
SCED	n/a	1.43	1.14	1.00	1.00	1.00	n/a

Из этой таблицы, в частности, следует, если в нашем проекте низкая квалификация аналитиков, то его трудоемкость возрастет примерно в 4 раза

по сравнению с проектом, в котором участвуют аналитики экстра-класса. И это не выдумки теоретиков, а отраслевая статистика!

Длительность проекта в методике COCOMO II рассчитывается по формуле:

$$TDEV = C \times (PM_{NS})^{D+0,2\times0,01\times\sum_{j=1}^{5} SF_j} \times \frac{SCED}{100}$$

где C = 3,67

D = 0,28

PM_{NS} – трудоемкость проекта без учета множителя SCED, определяющего сжатие расписания.

Задание на лабораторную работу

Выполнить анализ трудоёмкости, сроков реализации и экономической эффективности проекта, разработанного на предыдущих лабораторных.

Лабораторная работа 4

Тема: Планирование содержания и состава работ.

Теоретический материал

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.

Существует два подхода к декомпозиции работ проекта. Первый подход предполагает разбиение работ по стадиям жизненного цикла разработки программного обеспечения. В соответствии с этим подходом на верхнем уровне ИСР выделяются следующие проектные продукты:

1. Техническое задание.
2. Эскизный проект.
3. Технический проект.
4. Рабочий проект.
5. Внедрение.

Данный подход применим для реализации масштабных сложных проектов. В коммерческой разработке применяется иной подход. 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.

Ниже приведён пример ИСР для проекта разработки автоматизированной системы продажи документации.

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

1.5.1. Подготовка учебных курсов

1.5.2. Обучение сотрудников Отдела 123

1.5.3. Обучение руководства ОАО XYZ

1.5.4. Обучение администраторов системы

1.6. Ввод в опытную эксплуатацию

1.6.1. Развертывание и настройка прикладного ПО

1.6.2. Проведение приемо-сдаточных испытаний

1.6.3. 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.

Организационная структура исполнителей (OBS – Organization Breakdown Structure). Для обеспечения эффективного управления проектом необходимо знать, какая организация (исполнитель) ответственна за каждый пакет или уровень дерева работ. Это может быть сделано с помощью схемы организационной структуры проекта. В этой схеме руководитель проекта находится на ее верхнем уровне, а на более низких уровнях последовательно располагаются отделы, требуемые для функционального управления работами, или отдельные исполнители, привлекаемые для реализации отдельных пакетов работ. Эти уровни иногда соответствуют уровням WBS.

Матрица ответственности.

Для отражения иерархии подотчетности на проекте и указания обязанностей каждой из групп, входящих в проектную команду, в документ

описания содержания проекта рекомендуется включить матрицу ответственности, наиболее распространенный вариант которой известен как RACI-матрица. Использование данного инструмента особенно актуально в ситуации, когда проектная командасостоит из представителей различных юридических лиц (например, типичная команда на проекте внедрения КИС включает в себя сотрудников заказчика, генерального подрядчика и субподрядчиков). Матрица ответственности решает задачу демонстрации межорганизационного или межгруппового взаимодействия и, как следствие, позволяет избежать недоразумений, которые время от времени возникают в проектах между подразделениями и организациями из-за неясности, к кому следует обращаться по тем или иным вопросам и кто должен принимать по ним решение, а кто - непосредственно реализовать принятую резолюцию.

Важно как можно раньше произвести размежевание всех формальных полномочий, прав и обязанностей, пока команда проекта еще не приступила к активной работе. В противном случае, когда у сотрудников сложится собственное представление о своем месте в проекте, расхождения во мнениях по этим вопросам могут перерасти в затяжные конфликты и оказать значительное негативное влияние на график выполнения проекта.

Построение матрицы ответственности

1. Перечислить основные работы проекта.

По вертикали в матрице отражаются только основные работы проекта (не ниже уровня 2-3 ИСР), но с достаточной степенью детализации для обеспечения возможности указывать разные роли, необходимые для выполнения этих работ. Когда речь идет о крупных проектах и программах, может возникнуть необходимость разработать несколько матриц ответственности с различной степенью детализации.

2. Перечислить группы/роли внутри проектной команды.

По горизонтали в матрице перечисляются группы/ роли внутри проектной команды. Обратите внимание на то, что в матрице ответственности группы/роли, а не имена и фамилии отдельных членов

коллектива. Персональное закрепление проектных работ производится позднее, на этапе разработки расписания проекта.

3. Закодировать матрицу ответственности.

С помощью кодов в ячейках на пересечении соответствующих столбцов с ролями и строк с работами проекта указать степень участия, формальные полномочия и распределение ответственности за выполнение каждой операции. Четкое указание разных уровней формальных полномочий бывает особенно полезно в ситуации, когда множество членов проектной команды желает предъявить особые требования к проекту. На коды, используемые в *матрице ответственности*, каких-либо ограничений не существует, но наибольшее распространение получил метод RACI (Responsible (R), Accountable (A), Consulted (C), Informed(I)), описание которого приведено в таблице 9.

Таблица 9 – Условные обозначения матрицы ответственности (RACI)

Обозначение	Расшифровка	Описание
Исп. (R)	Исполнитель (Responsible)	Несет ответственность за непосредственное исполнение задачи. К каждой задаче должно быть приписано не менее одного исполнителя
Утв. (A)	Утверждающий (Accountable)	Отвечает за конечный результат перед вышестоящим руководством. На каждую работу должен быть назначен строго один подотчетный
Согл. (C)	Согласующий (Consulted)	Согласует принимаемые решения, взаимодействие с ним носит двусторонний характер
Н. (I)	Наблюдатель (Informed)	Его информируют об уже принятом решении, взаимодействие с ним носит

		односторонний характер

4. Инициировать использование матрицы и включить процедуру использования матрицы ответственности в документ "План управления проектом".

Пример построения матрицы ответственности приведён в таблице 10.

Таблица 10 – Распределение функциональных обязанностей команды управления проектом

Функциональные обязанности	Куратор проекта (Спонсор)	Руководитель проекта	Архитектор системы	Администратор проекта
1Планирование				
1.1 Разработка и периодическая актуализация плана	C	A	R	
Утверждение плана	R	A		
1.2 Управление командой проекта				
1.2.1 Назначение сотрудника на роль Руководителя проекта	R			
1.2.2 Формирование команды проекта	A	R		
1.2.3 Определение квалификационных требований и состава рабочих групп специалистов по функциональности ИС	I	A	R	I

Задание на лабораторную работу

В соответствии с разработанной ранее концепцией проекта разработать документ, описывающий содержание проекта. В данном документе должна быть представлена иерархическая структура работ проекта, ролевая модель команды проекта и матрица распределения ответственности.

Лабораторная работа 5

Тема: Разработка и оптимизация базового расписания проекта.

Теоретический материал

Метод построения диаграммы Ганта.

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.

Чтобы проиллюстрировать понятие критического пути рассмотрим пример «суперпроекта». Концепция проекта выглядит следующим образом.

Цель проекта. Сделать завтрак в постель

Результаты проекта. Завтрак в постели из вареного яйца, тоста и апельсинового сока.

Ресурсы. Имеется один оператор и обычное кухонное оборудование.

Сроки. Проект начинается на кухне в 8:00 и завершается в спальне.

Критерий приемки. Используются минимальные трудовые ресурсы и срок. Конечный продукт имеет высокое качество: яйцо свежесваренное, тост теплый, сок холодный.

Обоснование полезности. Проект служит достижению стратегических целей.

Иерархическая структура работ, ориентированная на конечный продукт, с оценкой их длительности представлена на рисунке 5.

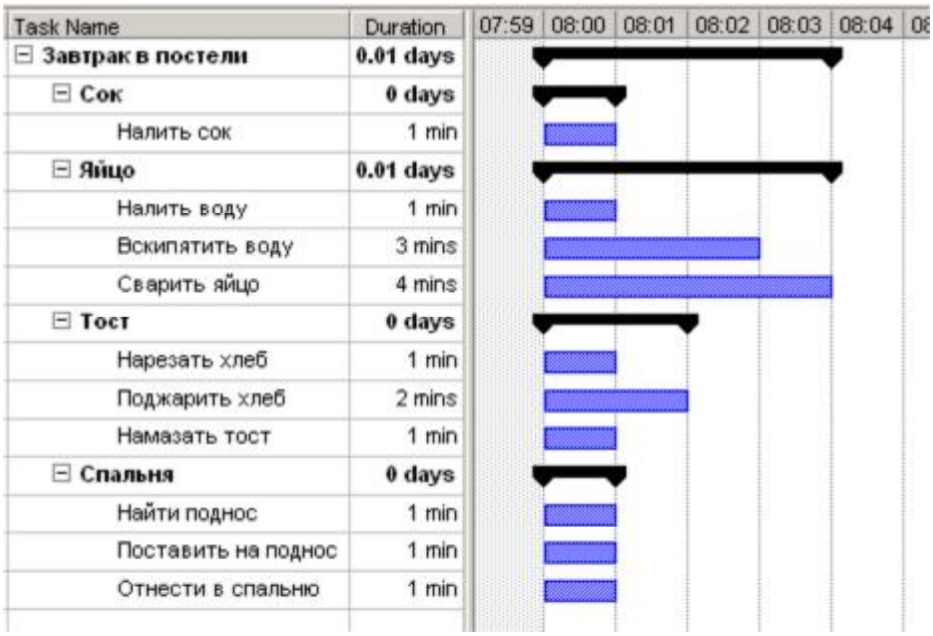

Рисунок 5 – Иерархическая структура работ проекта.

На следующем шаги мы должны учесть зависимости между работами, например, нельзя жарить хлеб, пока мы его не нарезали.

С учетом зависимостей мы получим следующую диаграмму расписания нашего проекта (Рисунок 6).

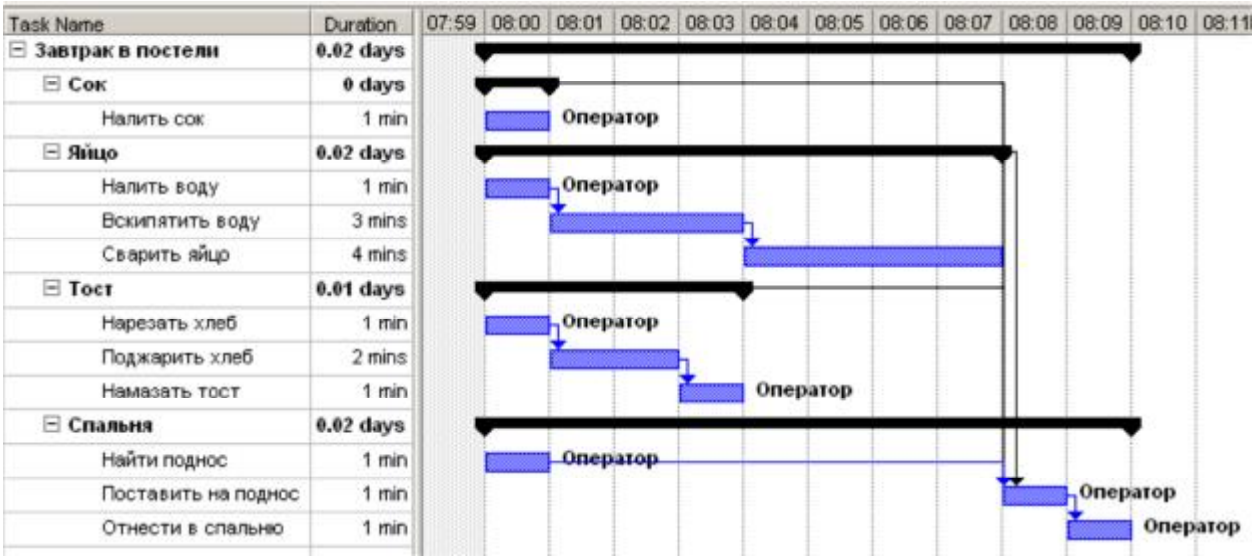

Рисунок 6 – Диаграмма расписания проекта с учётом зависимостей между работами

В результате мы определили, что минимальный срок реализации нашего проекта составляет 10 минут. Однако мы не можем на этом остановиться, поскольку должны еще учесть ограничение по ресурсам. У нас только один оператор. Если мы посмотрим на диаграмму загруженности ресурсов (Рисунок 7), то увидим, что наш критический ресурс загружен на первой минуте на 400%. что недопустимо.

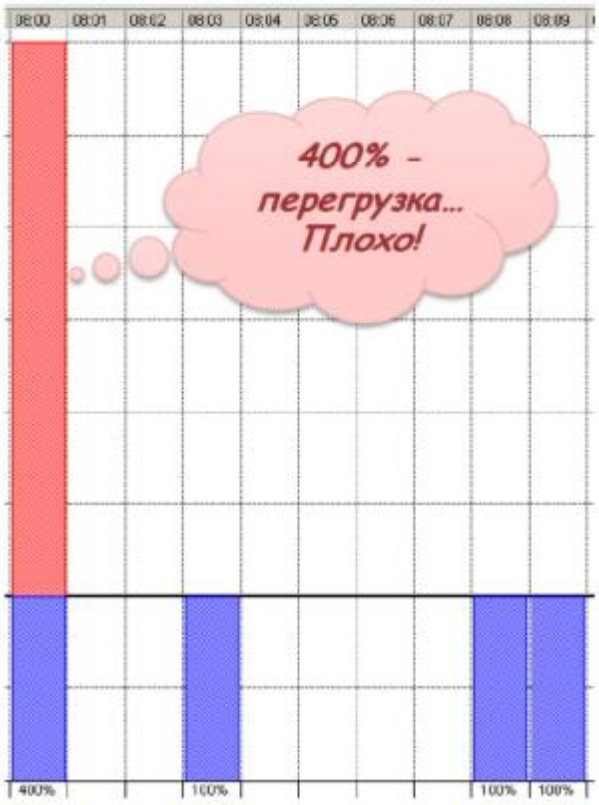

Рисунок 7 – Диаграмма загруженности ресурсов проекта

Следовательно, мы должны выполнить выравнивание ресурсов. Поскольку одним из критериев успеха проекта является его минимальная длительность, то если мы не хотим ее увеличивать, мы должны выявить критический путь в проекте (Рисунок 8) и не сдвигать работы, которые на нем находятся.

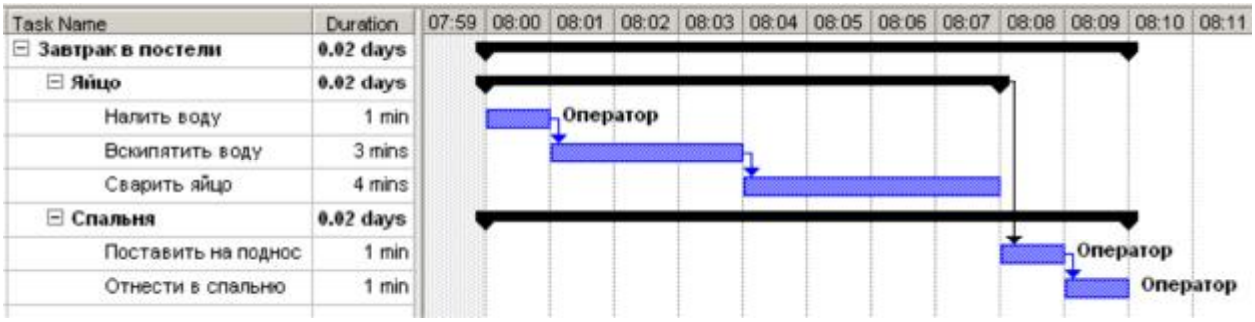

Рисунок 8 – Критический путь проекта

Поэтому, после выравнивания ресурсов, расписание нашего проекта будет выглядеть следующим образом (Рисунок 9).

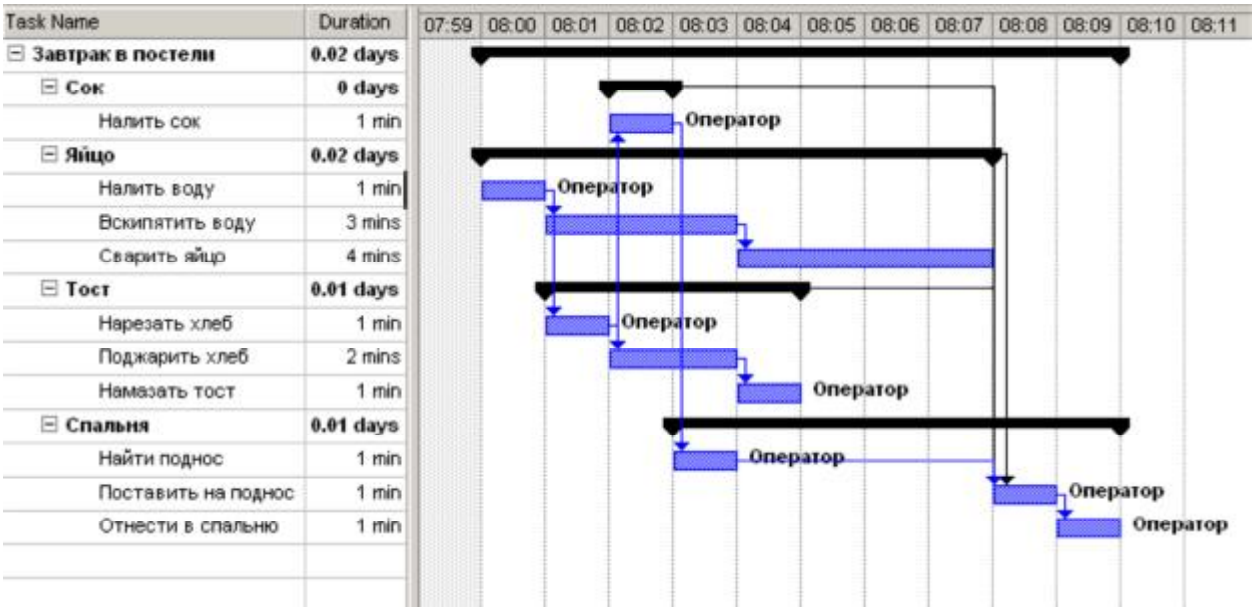

Рисунок 9 – Расписание проекта после выравнивания ресурсов

Теперь диаграмма загруженности ресурсов (Рисунок 10) выглядит приемлемо и у оператора даже появилось три минуты свободного времени на перекур. При этом общая длительность реализации проекта по-прежнему составляет 10 минут.

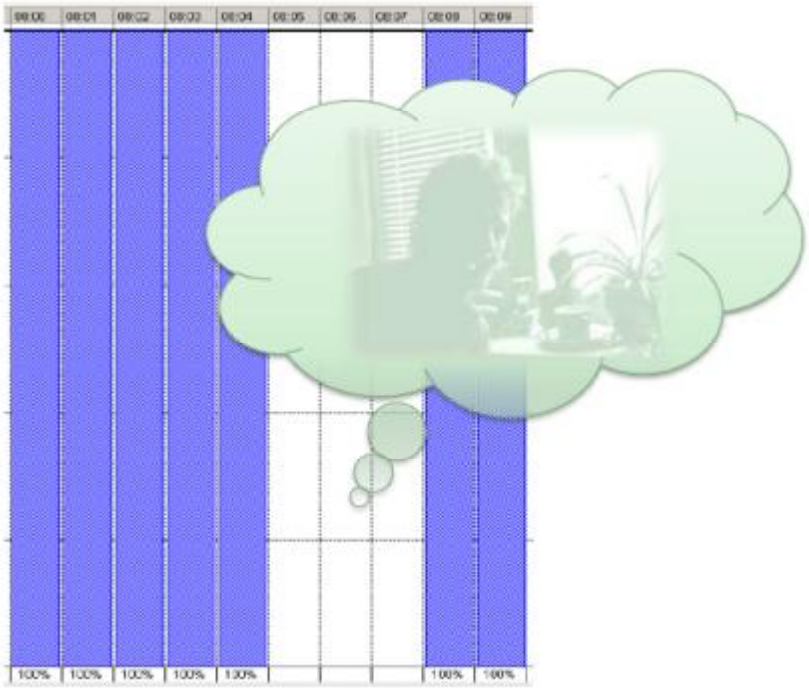

Рисунок 10 – Диаграмма загруженности ресурсов после выравнивания

Диаграмма Ганта является удобным механизмом для визуализации хода работ и контроля за их исполнением. Однако данный механизм не достаточно гибок и удобен в случаях, когда необходимо выполнить оптимизацию работ и ресурсов и проанализировать разные варианты организации работ по проекту. В этом случае применяется методика PERT и строится сетевой график проекта.

Разработка сетевого графика.

Сетевой анализ — это метод планирования и управления работами проектного характера, т.е. работами, операции в которых, как правило, не повторяются.

Анализ любого проекта осуществляется в три этапа:

1. Расчленение проекта на ряд отдельных работ (операций, подпроцессов), из которых затем составляется логическая структура проекта. Под операцией понимается деятельность или процесс, выполнение которых требует затрат временных и/или иных ресурсов.

2. Оценка продолжительности выполнения каждой операции; составление календарного плана выполнения проекта и выделение работ, которые определяют завершение выполнения проекта в целом.

3. Оценка потребностей каждой операции в ресурсах; пересмотр плана выполнения операций с учетом обеспечения ресурсами либо перераспределение денежных или других ресурсов, которое улучшит план.

Сетевой график представляет собой последовательность выполнения работ для получения конечного продукта. Стрелки на сетевом графике показывают взаимосвязи между работами. Ни одна работа не может быть начата до тех пор, пока не будут завершены все непосредственно предшествующие ей работы. Все работы на сетевом графике нумеруются в порядке их выполнения.

На следующем шаге рассчитывается ожидаемый срок выполнения работ:

$$t = \frac{a + 4m + b}{6}$$

где a – оптимистичный прогноз времени выполнения работы.

b – пессимистичный прогноз

m – наиболее вероятный.

Стандартное отклонение операции рассчитывается по формуле:

$$s = \frac{b - a}{6}$$

После оценки параметров работ строится сетевой график, показывющий взаимосвязь между работами на графе. Работы на сетевом графе изображаются стрелками. Каждая работа характеризуется своей продолжительностью. Вершинами сетевого графа являются событиями. События не имеют продолжительности и характеризуются наиболее ранним сроком наступления и наиболее поздним сроком наступления. События показывают моменты времени когда завершается одна работа и начинается другая. Для обознаения событий на сетевом графе используется примитив представленный на рисунке 11.

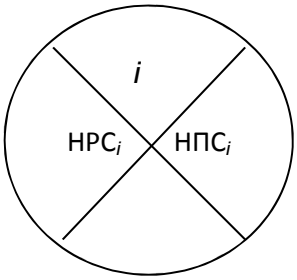

Рисунок 11 – Структурированное изображение вершины стрелочного графа

На рисунке обозначено:

i – номер события;

НРСi – наиболее ранний срок наступления события i;

НПСi – наиболее поздний срок наступления события i.

После построения сетевого графика необходимо оценить основные параметры работ:

1. Наиболее ранний срок наступления события.
2. Наиболее поздний срок наступления события.
3. Резерв времени.

Наиболее ранний срок наступления события. Будем считать, что срок наступления начального события с i=1, всегда равен нулю, т.е. $НРС_1 = 0$. Тогда наиболее ранний срок любого события, следующего за начальным будет вычисляться по простой формуле

$$HPC_j = HPC_1 + t_{1j}$$

где t_{1j} – время выполнения действия (1, j).

Формула для вычисления наиболее раннего срока начала для произвольного события в графе имеет вид:

$$HPC_j = HPC_i + t_{ij}$$

Если событию j предшествует несколько событий, то выбирается самый продолжительный срок:

$$HPC_j = \max_i \{HPC_i + t_{ij}\}$$

Наиболее поздний срок наступления события. Рассчитывается в обратном направлении по сети. Наиболее поздний срок последнего события

совпадает с его наиболее ранним сроком и равен продолжительности проекта. Для вычисления НПС используется формула

$$НПС_i = НПС_j - t_{ij}$$

Если из события i выходит несколько стрелок, то выбирается минимальный срок.

$$НПС_i = \min_j \{НПС_j - t_{ij}\}$$

Критический путь на графе образуют те события у которых ранний и поздний срок начала совпадают.

Оценка резерва времени включает в себя оценку дыух видов резерва.

1. Полный резерв времени работы – это максимальное количество времени, на которое можно задержать начало работы или увеличить ее продолжительность, не изменяя длительности критического пути:

$$Pn = НПС_j - НРС_i - t_{ij}$$

2. Свободный резерв времени работы – это максимальное количество времени, на которое можно увеличить продолжительность данной работы, не изменяя при этом ранних сроков начала последующих работ при условии, что непосредственно предшествующее событие наступило в свой ранний срок:

$$Pc = НРС_j - НРС_i - t_{ij}$$

На основе рассчитанных параметров строится сетевой график. Пример графика представлен на рисунке 12.

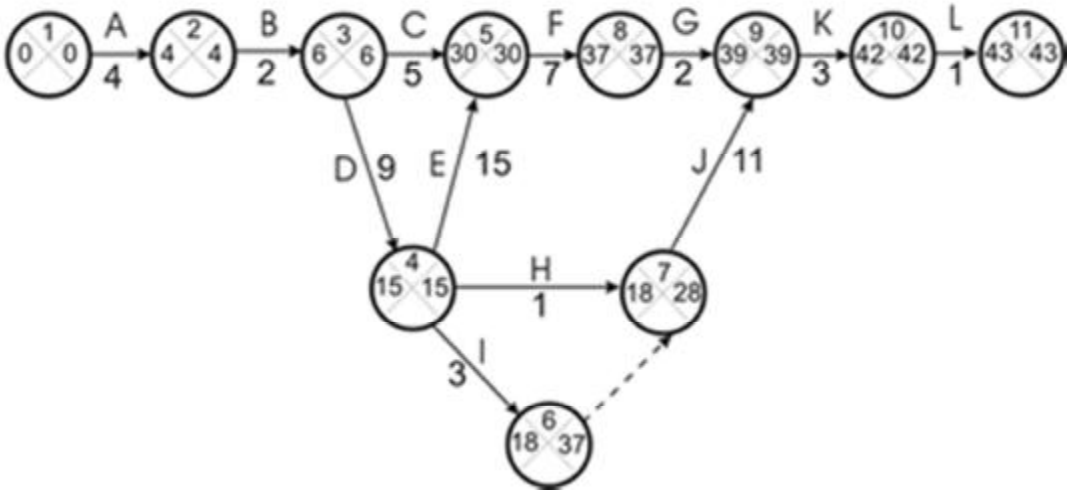

Рисунок 12 – Пример сетевого графика

После построения сетевого графика производится оценка вероятности завершения проекта в заданные сроки. Для этого рассчитывается стандартное отклонение общего времени Т от среднего значения по формуле.

$$s(\overline{T}) = \sqrt{s_{T1}^2 + s_{T2}^2 + \ldots + s_{Tn}^2}$$

Вероятность того, что общее время выполнения проекта окажется меньше Т можно находить с помощью программы Microsoft Excel/Мастер функций/Статистические функции/Нормрасп. Квантиль стандартного нормального распределения будет равен

$$z = \frac{T - \overline{T}}{s(\overline{T})}$$

Если вероятность завершения проекта в выбранные сроки достаточно низкая, то происходит сокращение сроков выполнения проекта засчёт привлечения дополнительных ресурсов. Сокращение времени выполнения проектируемого процесса производится за счет сокращения времени операций критического пути на основе последовательного подбора целесообразного сокращения времени выполнения операций. Здесь последовательно сокращается время выполнения только тех операций, которые лежат на критическом пути, начиная с операций имеющих наименьший средний градиент издержек.

Расчет среднего градиента издержек для каждой i-й операции проекта производится по формуле

$$grad(s_i) = \frac{s_i^{pr} - s_i}{t_i - t_i^{pr}} = \frac{\Delta s_i}{\Delta t_i}$$

Где s_i , t_i – изначально планируемые (нормальные) стоимость и время выполнения операции i;

t_i^{pr} – минимально возможное (предельное) время выполнения операции i;

s_i^{pr} – увеличенная (предельная) стоимость выполнения операции i, соответствующая минимальному времени .

Чем больше градиент издержек, тем дороже обходится сокращение времени выполнения операции. После выбора работ с наименьшим градиентом издержек, происходит их сокращение, пересчитываются все параметры сетевого графика и строится новый сетевой график. При этом

критический путь может измениться. При этом оценивается эффективность проведённого изменения как разница между дополнительной прибылью, полученной за каждый день сокращения сроков выполнения проекта и затратами на сокращение сроков. Если разница положительна, то проведённые сокращения эффективны.

Сокращение продолжается до тех пор, пока позволяют временные и стоимостные ограничения. При наличии двух или более критических путей необходимо сокращать длительности выполнения операций на этих путях на одно и то же количество единиц времени.

Задание на лабораторную работу

На основе разработанной в предыдущей лабораторной работе структуры проекта построить базовое расписание проекта в виде диаграммы Ганта. Вычислить критический путь. Рассчитать параметры работ и построить сетевой график. Проанализировать вероятность завершения проекта в планируемый срок и возможность сокращения сроков реализации проекта.

Лабораторная работа 6

Тема: Анализ и планирование рисков проекта.

Теоретический материал

Понятие и характеристики риска.

Риск проекта – это суммарный эффект вероятностей наступления неопределенных событий, способных оказать отрицательное или положительное влияние на цели проекта.

Риски подразделяются на известные и неизвестные. Известные риски идентифицируются и подлежат управлению – создаются планы реагирования на риски и резервы на возможные потери. Неизвестные риски нельзя определить, и, следовательно, невозможно спланировать действия по реагированию на такой риск. Поэтому создают определённый резерв ресурсов для реагирования на них.

Риск характеризуется следующими компонентами.

Событие риска – потенциально возможное событие, которое может нанести ущерб или принести выгоды проекту.

Вероятность возникновения риска. Все риски имеют вероятность больше нуля и меньше 100%. Риск с вероятностью 0 не может произойти и не считается риском. Риск с вероятностью 100% также не является риском, поскольку это достоверное событие, которое должно быть предусмотрено планом проекта.

Последствия риска, если он случится, выражаются через дни расписания, трудозатраты, деньги и определяют степень воздействия на цели проекта.

Величина риска показатель, объединяющий вероятность возникновения риска и его последствия. Величина риска рассчитывается путем умножения вероятности возникновения риска на соответствующие последствия.

План управления рисками – документ, разрабатываемый в начале проекта и содержащий описания структуры управления рисками проекта и порядок его выполнения в рамках проекта; включается в состав плана управления проектом. План управления рисками обычно содержит следующие элементы:

1. Методика – определение подходов, инструментов и источников данных, которые могут использоваться для управления рисками в данном проекте.

2. 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.

3. Определение операций по управлению рисками, которые необходимо включить в расписание проекта.

4. Определение сроков и частоты выполнения операций по управлению рисками на протяжении всего жизненного цикла проекта.

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

6. Классификация рисков (или категории рисков) – структура, на основании которой производится систематическая и всесторонняя идентификация рисков с нужной степенью детализации.

7. Определение вероятности возникновения рисков и их последствий. Общие определения уровней вероятности и уровней воздействия адаптируются отдельно для каждого проекта в ходе процесса планирования управления рисками и используются в процессе качественного анализа рисков. Можно применить относительную шкалу, на которой вероятность обозначена описательно, со значениями от "крайне маловероятно" до "почти наверное", или шкалу, на которой вероятности соответствует цифровое значение, например: 0,1 - 0,3 - 0,5 - 0,7 — 0,9. Матрица вероятности и

последствий (Таблица 11) содержит комбинации вероятности и воздействия, при помощи которых рискам присваивается определенный ранг: низкий, средний или высший. Матрица может содержать описательные термины или цифровые обозначения и строится на основании шкал оценки вероятности и оценки степени влияния возможного риска. Левый столбец матрицы содержит значения вероятности возникновения риска, в первой строке расположена шкала со значениями возможных последствий. Ячейки заполняется результатами перемножения значений этих шкал.

Таблица 11 – Матрица вероятности и последствий

Вероятность	Угрозы					Благоприятные возможности				
0,9	0,05	0,09	0,18	0,36	0,72	0,72	0,36	0,18	0,09	0,05
0,7	0,04	0,07	0,14	0,2	0,56	0,56	0,28	0,14	0,07	0,04
0,5	0,03	0,05	0,1	0,2	0,4	0,4	0,2	0,1	0,05	0,03
0,3	0,02	0,03	0,06	0,12	0,24	0,24	0,12	0,06	0,03	0,02
0,1	0,01	0,01	0,02	0,04	0,08	0,08	0,04	0,02	0,01	0,01
	0,05	0,1	0,2	0,4	0,8	0,8	0,4	0,2	0,1	0,05

Сопоставляя значение ячейки матрицы со шкалой оценки воздействия, риски можно разделить по категориям – малые, средние и большие. Риски, имеющие очень высокие вероятности, но незначительные последствия, а также риски, имеющие низкие вероятности и незначительные последствия, считаются рисками, не оказывающими воздействия (клетки таблицы красного цвета). Риски с очень большими последствиями, но малой вероятностью, как и риски с незначительными последствиями и высокой вероятностью (клетки белого цвета) имеют среднее воздействие на проект. Риски, которым необходимо уделять особое внимание, имеют достаточно высокую вероятность и существенные последствия (клетки таблицы, зелёного цвета).

8. Уточнение толерантность к рискам участников проекта.

9. Формы отчетности. Определяется формат реестра рисков и его содержание, а также любых других требуемых отчетов по рискам. Определяет, каким образом производится документирование, анализ и обмен информацией о результатах процесса управления рисками.

10. Отслеживание. Документируется порядок регистрации всех аспектов операций по рискам в интересах данного проекта, а также для будущих проектов и для включения в документы по накопленным знаниям. Документируется, в каких случаях и как будет проводиться аудит процессов управления рисками.

В ходе разработки плана управления рисками и последующих действий по их идентификации и оценке составляют карты рисков проекта. Карта рисков представляет собой таблицу, структура которой представлена в таблице 12.

Таблица 12 – Карта рисков

№	Наименование риска	Организация	Событие	Вид риска	Описание риска	Причины	Последствия	Общий риск		
								Вероятность	Последствия	Оценка
1	2	3	4	5	6	7	8	9	10	11

Продолжение таблицы 12

Владелец процесса	Воздействие на риск	Стоимость воздействия	Регламентация процесса	Остаточная оценка риска		
				Вероятность	Последствия	Оценка
12	13	14	15	16	17	18

Анализ рисков проекта происходит в три этапа.

1. Идентификация рисков. На данном этапе можно ограничиться заполнением первых восьми столбцов карты рисков. В столбцах 2-5 необходимо указать наименование риска, конкретное событие и вид риска в

соответствии с приведенной классификацией рисков. Каждое событие (риск) должно однозначно соответствовать одной из групп риска. Например:

Наименование риска (2) — Конкуренты;

Событие (4) — Появление новых конкурентов на рынке;

Вид риска (5) — Стратегические риски.

Описание риска (6) должно включать определение риска с указанием факторов, причин и/или событий, обусловливающих вероятность проявления данного риска, а также обоснование возможных последствий.

Причины (7). Более подробный перечень причин, обусловливающих наступление данного события. Отсутствие контроля или методов воздействия на риск причинами не является.

Последствия (8). Более подробный перечень возможных последствий от наступления рискового события, выраженных в конкретных денежных потерях, потерях времени, качества, потерях репутации, доли рынка и т. д.

2. Качественный анализ рисков. На данном этапе заполняют 9-11 графы карты. Риск оценивается по двум параметрам: вероятность и последствия. На первом этапе риск оценивается до применения методов воздействия на риск. Оценка риска до воздействия является общей оценкой.

Вероятность (9) наступления рискового события оценивается в виде коэффициента, значение которого определяется экспертным методом в соответствии с таблицей 11.

Последствия (10). В данном столбце необходимо указать коэффициент существенности последствий наступления рисковых событий в течение выполнения работ по данному Проекту с предположением того, что данный риск не управляется и не контролируется. Значение коэффициента определяется экспертным методом в соответствии с таблицей 12.

Существенность последствия рисковых событий можно также оценить и в денежном выражении. Для каждого Проекта коэффициентам последствий могут соответствовать различные денежные выражения. В качестве базы для оценки последствий в денежном или другом выражении можно

использовать: годовой оборот, годовую выручку, позицию на рынке, репутацию и т. п.

Оценка (11) риска рассчитывается как произведение коэффициента вероятности на коэффициент последствий и показывает общую оценку риска с предположением того, что данный риск не управляется и не контролируется. Для наглядности и последующего анализа строится матрица вероятности и последствий.

3. Количественный анализ рисков и планирование реагирования на риски. В ходе планирования необходимо заполнить оставшуюся часть карты рисков.

Владелец процесса управления данным риском (12) – лицо, отвечающее за оценку, контроль и управление данным риском, т. е. мотивированное на его снижение и располагающее достаточными полномочиями для воздействия на него.

Воздействие на риск (13) – описание конкретных методов воздействия на риск, применяемых владельцем риска или другими подразделениями. Любое мероприятие или функция, способствующая снижению вероятности или последствий наступления рискового события, может быть рассмотрена как метод воздействия.

Стратегиями реагирования на негативные риски (угрозы) являются принятие, уклонение, передача или снижение.

Стоимость воздействия (14). Экспертная оценка стоимости применяемых методов воздействия на риск. Рекомендуется использовать следующую шкалу: высокая, средняя и низкая стоимость воздействия, выраженную в денежном эквиваленте.

Документы, регламентирующие процессы управления риском (15). Утвержденные внутренние документы, регламентирующие тем или иным образом процессы управления риском: выявление, оценку, анализ, контроль или воздействие. К таким документам может относиться любой документ, в

котором прямо или косвенно регламентирован процесс, позволяющий тем или иным образом снизить оценку данного риска.

Остаточная оценка риска (16-18). Оценивается по аналогии с общим риском по двум параметрам: вероятность и последствия.

Вероятность (16) наступления рискового события с учетом того, что данный риск управляется и контролируется, т. е. после воздействия указанных выше методов управления. Оценивается экспертным методом в соответствии со шкалой идентичной оценки вероятности на предыдущем этапе.

Последствие (17) наступления рискового события с учетом того, что данный риск управляется и контролируется, т. е. после воздействия методов управления. Оценивается экспертным методом в соответствии со шкалой идентичной оценки последствий на предыдущем этапе.

Оценка (18) — остаточная оценка риска с предположением того, что данный риск управляется и контролируется, т. е. после воздействия методов управления. Рассчитывается как произведение вероятности на последствия. После этого для наглядности и последующего анализа ответственным за управление рисками проекта строится матрица вероятности и последствий с учетом реагирования на риски, аналогично представленной матрицы без учета процедур реагирования.

Результатами планирования реагирования на риски проекта должны стать новая версия карты рисков, обновленный план управления проектом, включающий в себя операции по реагированию на риски, и контрактные соглашения для определения ответственности каждой из сторон на случай возникновения каждого отдельного риска.

Задание на лабораторную работу

Разработать план управления рисками проекта создания программного продукта и соответствующий реестр рисков. Реестор рисков должен

включать в себя общую карту рисков (Таблица 12). Для наиболее вероятных рисков с высокими последствиями описать карточки рисков в виде представленном в таблице 13.

Таблица 13 – 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.	

Лабораторная работа 7

Тема: Разработка плана закрытия проекта и оформление отчётности.

Теоретический материал

Для качественного завершения проекта, достижения удовлетворённости сторон, предотвращения потерь и сохранения опыта действия по закрытию проекта должны носить плановый характер.

План закрытия проекта включает в себя все действия, необходимые для административного завершения проекта или фазы, включая:

1. действия и операции, необходимые для удовлетворения критериев завершения или выхода для фазы или проекта;
2. действия и операции, необходимые для передачи продуктов, услуг или результатов проекта в следующую фазу или в производство и/или операционную деятельность;
3. операции, необходимые для сбора документов проекта или фазы, проверки успешности или неудачи проекта, аккумулирования полученных знаний и архивирования информации по проекту для будущего использования организацией.

В ходе проекта накапливается большой объем самой разнообразной информации:

1. чертежи, технические спецификации и руководства по эксплуатации оборудования, закупленного для проекта;
2. техническое описание или схема результата проекта;
3. планы и программы работ по проекту;
4. учетные документы, касающиеся расходов на реализацию проекта;
5. контракты и субконтракты по проекту.

Осуществление закрытия состоит из следующих операций:

1. составить список открытых вопросов и заключительных работ; сдать результаты проекта Заказчику;
2. принять поручение Заказчика о закрытии проекта;

3. закрыть все ресурсы и передать их на новые объекты;

4. выполнить заключительную оценку финансовой ситуации (постпроектный отчет), закрыть все финансовые операции и проследить, чтобы все счета были оплачены;

5. выполнить документирование и анализ опыта выполнения проекта: создать заключительный отчет по проекту и сдать проектную документацию;

6. разрешить все спорные вопросы;

7. оценить работу проектной команды, членов проектной команды и руководителя проекта (в отдельной части отчёта и в виде благодарностей или иных документов морального и материального стимулирования, например, КТУ — коэффициент трудового участия);

8. оформить роспуск и перераспределение членов проектной команды.

Итоговый отчет должен содержать следующую информацию:

1. Итоги проекта:
   - 1.1. Достижение целей проекта
   - 1.2. Дополнительные полезные результаты
   - 1.3. Фактические сроки
   - 1.4. Фактические расходы
   - 1.5. Обоснование отклонения от целей
   - 1.6. Отклонения результатов от требований
2. Уроки проекта:
   - 2.1. Проблемы проекта и способы их решения
   - 2.2. Материалы программные компоненты для последующего использования
   - 2.3. Предложения по изменению технологий или стандартов компании

Задание на лабораторную работу

Разработать план закрытия проекта разработки проргаммного обеспечения. Подготовить итоговый отчёт в соответствии с приведённой структурой.