МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное
автономное образовательное
учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКИМ РАБОТАМ

«Экономика программной инженерии»

Направление подготовки 09.03.04 «Программная инженерия»
Направленность (профиль) «Инженерия сложных программных систем»
Квалификация выпускника: бакалавр

Ставрополь

2026

Введение

Цель освоения дисциплины «Экономика программной инженерии»:

Формирование у студентов системного понимания экономических аспектов разработки программных продуктов, включая методы оценки трудоемкости и стоимости, бюджетирование, ценообразование, налогообложение в ИТ-сфере.

Задачами обучения являются:

- изучение особенностей программного продукта как товара на профильном рынке и типов ИТ-компаний;

- освоение современных методов и моделей оценки трудоемкости и стоимости разработки ПО (Functional Points, COCOMO II, PERT);

- формирование навыков формирования бюджета проекта разработки и его оптимизации;

- изучение метрик финансового анализа (ROI, NPV, IRR) и методов оценки экономической эффективности ИТ-проектов;

- освоение стратегий ценообразования, моделей монетизации и лицензирования программных продуктов;

- изучение правовых аспектов: создание и регистрация ИТ-компании, налогообложение, госзакупки, защита интеллектуальной собственности;

- формирование навыков документирования и презентации экономических обоснований проектов.

1. Перечень планируемых результатов обучения по дисциплин (модулю), соотнесенных с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов

ПК-1 – Способен осуществлять управление программными проектами: планировать, контролировать сроки и ресурсы, применять системы контроля версий и инструменты управления задачами	ИД-1$_{ПК-1}$ Применяет методы оценки трудоемкости (PERT, функциональные точки, COCOMO II) для планирования сроков и ресурсов программного проекта ИД-4$_{ПК-1}$ Анализирует план-фактные отклонения по срокам и бюджету, предлагает корректирующие действия	Способен применять методы оценки трудоемкости (PERT, функциональные точки, COCOMO II) для планирования сроков и ресурсов программного проекта Способен анализировать план-фактные отклонения по срокам и бюджету, предлагать корректирующие действия
ПК-9 – Умеет оформлять техническую документацию, готовить презентации и отчеты, вести деловую коммуникацию на русском и иностранном языках в профессиональной среде	ИД-1$_{ПК-9}$ Оформляет техническую документацию (ТЗ, пояснительная записка, руководство пользователя, API-спецификация) в форматах Markdown, OpenAPI, DocBook в соответствии с ГОСТ	Способен оформлять экономические обоснования проектов, включая расчеты трудоемкости, бюджета и финансовых показателей
ПК-10 – Готов соблюдать правовые нормы, стандарты и требования информационной безопасности, участвовать в разработке нормативной и технической	ИД-1$_{ПК-10}$ Применяет основные правовые нормы (лицензирование ПО, авторское право, ФЗ-152, ФЗ-149) и стандарты ИБ (ГОСТ Р ИСО/МЭК 27001, ГОСТ Р 56545-2015) при разработке и эксплуатации ПО	Способен учитывать правовые нормы при выборе организационно-правовой формы и системы налогообложения для ИТ-компании

Лабораторная работа № 1

Выбор идеи программного продукта и формирование требований

Цель: Инициировать проект для дальнейшего экономического анализа.

Теоретическое обоснование

Экономический анализ любого ИТ-проекта начинается с этапа инициации, на котором формируется концепция продукта и собираются требования. От качества этого этапа напрямую зависят точность последующих оценок трудоёмкости, бюджета и экономической эффективности.

Идея программного продукта – это краткое описание того, какой потребность рынка или пользователя удовлетворяет продукт, какую проблему решает и как это делает уникальным образом. Хорошая идея должна быть проверяема (falsifiable) и измерима.

Функциональные требования описывают, что система должна делать: конкретные действия, реакции на ввод пользователя, бизнес-логику. Пример: «Приложение должно автоматически категоризировать расходы по тегам». Нефункциональные требования описывают, как система должна работать: производительность (время отклика), безопасность (шифрование, аутентификация), масштабируемость, надёжность (время безотказной работы), удобство использования и т.д.

Целевая аудитория – это группа потребителей, для которых продукт создаёт наибольшую ценность. Аудиторию сегментируют по демографическим, поведенческим, психографическим признакам. Для B2B-продуктов описывают портрет лица, принимающего решение (ЛПР).

Позиционирование на рынке – это место продукта в сознании потребителя относительно конкурентов. Используют карту позиционирования (оси: цена – качество, функциональность – простота и др.) и анализ конкурентов (SWOT-анализ, пять сил Портера).

Важным понятием является MVP (Minimum Viable Product) – минимальный набор функций, достаточный для запуска и получения обратной связи от первых пользователей. MVP позволяет сократить первоначальные

инвестиции и проверить гипотезы спроса.

Для документирования требований используют пользовательские истории (user stories) формата: «Как <роль>, я хочу <действие>, чтобы <ценность>». Приоритизация требований выполняется методом MoSCoW: Must have (обязательные), Should have (важные), Could have (желательные), Won't have (не будут сделаны сейчас).

Также на этом этапе создают видение продукта (vision statement) по шаблону: «Для [целевой аудитории], у которой есть [потребность], наш продукт – это [категория продукта], который предоставляет [ключевое преимущество]. В отличие от [конкурентов], он [главное отличие]».

В итоге формируется бэклог требований – структурированный список, который служит входными данными для оценки размера проекта (функциональные точки) и трудоёмкости.

Методика и порядок выполнения работы

Задание 1. Задания по вариантам

Вариант	
1	Разработать идею мобильного приложения для управления личными финансами с функциями автоматической категоризации трат
2	Разработать идею веб-сервиса для онлайн-бронирования репетиторов с системой рейтинга
3	Разработать идею десктопного приложения для автоматизации документооборота в малом бизнесе
4	Разработать идею SaaS-платформы для управления проектами с интеграцией с мессенджерами
5	Разработать идею мобильного приложения для трекинга привычек с элементами геймификации
6	Разработать идею интернет-магазина нишевых товаров (на выбор) с системой рекомендаций
7	Разработать идею сервиса для автоматического распознавания и оцифровки чеков
8	Разработать идею B2B-платформы для организации удалённой работы команды поддержки
9	Разработать идею образовательного портала с адаптивным обучением и тестированием
10	Разработать идею маркетплейса для фрилансеров с эскроу-

	сервисом

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Название и цель продукта
3) Список функциональных требований (не менее 15) с приоритизацией
4) Список нефункциональных требований (производительность, безопасность, масштабируемость и т.п.)
5) Описание целевой аудитории (портрет пользователя, сегменты)
6) Карта позиционирования (2-3 конкурента)
7) Вывод о перспективности идеи.
8) Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое функциональные и нефункциональные требования? Приведите примеры.
2. Какие методы сбора требований вы знаете?
3. Как определить целевую аудиторию программного продукта?
4. Что такое MVP (Minimum Viable Product) и зачем он нужен?
5. Перечислите основные элементы vision-документа.
6. В чём разница между продуктом и проектом?
7. Как провести первичный анализ конкурентов?
8. Что такое позиционирование продукта?
9. Какие критерии используются для приоритизации требований?
10. Как описать ценностное предложение по шаблону В. Франка?

Критерии оценивания

Параметр	Баллы
Обоснованность идеи и актуальность	4

Полнота и качество требований	6
Чёткость описания целевой аудитории	4
Корректность позиционирования	3
Оформление отчёта	3

Лабораторная работа № 2

Оценка трудоемкости разработки наивным методом и методом PERT

Цель: Освоить базовые методы оценки трудозатрат.

Теоретическое обоснование

Оценка трудоёмкости (effort estimation) – один из самых сложных и критичных процессов в управлении ИТ-проектами. Недооценка приводит к срыву сроков, перерасходу бюджета; переоценка – к потере конкурентного преимущества.

Наивный метод (экспертная оценка) – это интуитивная оценка, основанная на опыте одного или нескольких экспертов. Метод прост и быстр, но подвержен систематическим ошибкам: эффекту якорения, оптимизму/пессимизму, влиянию недавних проектов. Для повышения точности применяют **метод Дельфи** (независимые анонимные оценки, несколько раундов) или **Planning Poker** (используемый в Scrum).

Метод PERT (Program Evaluation and Review Technique) был разработан для управления проектами ВМФ США. В отличие от наивного метода, PERT использует три оценки для каждой задачи:

1) Оптимистическая (O) – время при самых благоприятных условиях (вероятность ~1%).
2) Пессимистическая (P) – время при самых неблагоприятных условиях (вероятность ~1%).
3) Наиболее вероятная (M) – время при нормальных условиях.

Ожидаемая трудоёмкость (или длительность) вычисляется по формуле:

$$E = \frac{O + 4M + P}{6}$$

Дисперсия (мера неопределённости):

$$\sigma^2 = \left(\frac{P - O}{6}\right)^2$$

Чем больше разрыв между P и O, тем выше риск.

Сетевая диаграмма – это граф, где вершины – события (начала/окончания задач), а дуги – задачи с указанием длительности. Для построения сетевого графика определяют зависимости между задачами:

- FS (Finish-to-Start): задача B начинается после окончания A.
- SS (Start-to-Start): B начинается вместе с A.
- FF (Finish-to-Finish): B заканчивается вместе с A.
- SF (Start-to-Finish): B заканчивается после начала A (редко).

Критический путь – это самый длинный полный путь в сетевом графике от начала до конца. Любая задержка на критическом пути сдвигает срок завершения всего проекта. Некритические задачи имеют резерв времени (float) – задержку, не влияющую на финальный срок.

Расчёт ранних и поздних сроков:

- Ранний старт (ES) = max(ранних окончаний предшественников)
- Ранний финиш (EF) = ES + длительность
- Поздний финиш (LF) = min(поздних стартов последователей)
- Поздний старт (LS) = LF – длительность
- Полный резерв = LS – ES (или LF – EF)

После оценки длительностей определяют **оптимальное количество разработчиков**. Задача не всегда может быть распараллелена: некоторые задачи требуют последовательного выполнения (например, сначала проектирование, потом кодирование). Закон Брукса гласит: добавление новых разработчиков в отстающий проект может ещё больше его замедлить из-за коммуникационных накладных расходов. Оптимальный размер команды находится по формуле (эвристика): для проекта длительностью T и трудоёмкостью E человеко-месяцев средняя численность = E/T, но с учётом коммуникаций (метрика «линии связи» = n(n-1)/2).

В итоге PERT даёт не только точечную оценку, но и вероятностное

распределение сроков, что позволяет планировать резервы времени.

Методика и порядок выполнения работы

Задание 1. Для продукта из ЛР1 выполнить декомпозицию на 10-15 задач и:

1. Рассчитать трудоёмкость наивным методом, затем PERT. Сравнить.
2. Построить сетевую диаграмму и найти критический путь при 1 разработчике.
3. Найти критический путь при параллелизации (3 разработчика).
4. Оценить вероятность завершения за заданный срок (на основе дисперсии).
5. Выполнить расчёт с учётом зависимости задач (FS, SS, FF).
6. Использовать PERT для каждой задачи и вычислить общую ожидаемую трудоёмкость.
7. Оптимизировать состав команды, чтобы уложиться в директивный срок.
8. Провести анализ "что-если" – увеличить объём требований на 20%.
9. Сравнить результаты PERT с наивным методом в виде графика.
10. Рассчитать резерв времени для некритических задач.

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Список задач с декомпозицией (WBS)
3) Таблица с оценками O, M, P для каждой задачи
4) Расчёт ожидаемой трудоёмкости и дисперсии
5) Сетевая диаграмма (рисунок)
6) Указание критического пути и его длины
7) Определение оптимального количества разработчиков
8) Сравнение с наивным методом.
9) Ответы на контрольные вопросы.

Контрольные вопросы

1. В чём недостатки наивного метода оценки?
2. Что такое оптимистическая, пессимистическая и наиболее вероятная оценки?
3. По какой формуле рассчитывается ожидаемая трудоёмкость в PERT?
4. Что такое критический путь и как его найти?
5. Как дисперсия оценки влияет на риск несоблюдения сроков?
6. Какие типы зависимостей между задачами существуют?
7. Как определить оптимальное количество разработчиков?
8. В чём разница между трудоёмкостью и длительностью?
9. Какой метод точнее – наивный или PERT? Почему?
10. Что такое «парадокс Брукса» и как он связан с оценкой трудоёмкости.

Критерии оценивания

Параметр	Баллы
Корректность декомпозиции	5
Правильность расчётов PERT	8
Построение сетевой диаграммы	5
Обоснование выбора состава команды	4
Анализ и сравнение методов	3

Лабораторная работа № 3

Оценка размера проекта методом функциональных точек

Цель и содержание работы: Освоить метод функциональных точек для оценки размера ПО.

Теоретическое обоснование

Метод **функциональных точек (Function Points, FP)** был предложен Алланом Албрехтом в IBM в 1979 году для измерения размера ПО с точки зрения функциональности, предоставляемой пользователю, независимо от языка и

технологии реализации. Это особенно важно на ранних этапах, когда ещё неизвестны языки программирования.

Согласно стандарту IFPUG (International Function Point Users Group), выделяют **пять типов компонентов**:

1. Внешние вводы (External Inputs, EI) – элементарные процессы, которые вносят данные извне в систему (например, форма регистрации, загрузка файла). Сложность определяется по количеству полей данных и количеству обращаемых файлов.
2. Внешние выводы (External Outputs, EO) – процессы, которые извлекают данные из системы и предоставляют их пользователю с обработкой или расчётами (отчёты, графики, экспорт).
3. Внешние запросы (External Queries, EQ) – простые запросы без расчётов и изменения состояния (поиск, просмотр).
4. Внутренние логические файлы (Internal Logical Files, ILF) – группы логически связанных данных, поддерживаемые внутри системы (таблицы БД, коллекции, хранилища).
5. Внешние интерфейсные файлы (External Interface Files, EIF) – данные, на которые система ссылается, но не поддерживает сама (например, справочник из внешней системы).

Каждому компоненту присваивается **сложность** (низкая, средняя, высокая) на основе эвристических таблиц IFPUG. Например, для ILF: низкая – 1-19 DET (элементы данных), средняя – 20-50, высокая – более 50; плюс количество RET (записи). Весовые коэффициенты:

- EI: 3, 4, 6
- EO: 4, 5, 7
- EQ: 3, 4, 6
- ILF: 7, 10, 15
- EIF: 5, 7, 10

Некорректированные функциональные точки (UFP) = сумма произведений количества компонентов каждого типа и сложности на веса.

Затем учитываются **14 общих характеристик системы (General System Characteristics, GSC)** по шкале от 0 (не влияет) до 5 (существенно влияет):

1. Резервное копирование и восстановление
2. Связь с данными
3. Распределённая обработка
4. Производительность
5. Интенсивность использования оборудования
6. Частота транзакций
7. Доля пользователей, работающих онлайн
8. Сложность обновления данных
9. Сложность интерфейсов
10. Сложность обработки
11. Возможность повторного использования
12. Удобство установки
13. Многопользовательский режим
14. Гибкость изменений

Сумма оценок GSC (TDI) используется в формуле:

$$\text{VAF} = 0,65 + 0,01 \times \text{TDI}$$

VAF (Value Adjustment Factor) находится в диапазоне от 0,65 до 1,35. Таким образом, **скорректированные функциональные точки**:

$$\text{AFP} = \text{UFP} \times \text{VAF}$$

Метод FP позволяет пересчитать размер в строки кода (SLOC) с помощью таблиц плотности для конкретных языков. Например:

- Ассемблер: 1 FP ≈ 320 SLOC
- C++: 1 FP ≈ 50-80 SLOC
- Java: 1 FP ≈ 40-60 SLOC
- Python: 1 FP ≈ 15-25 SLOC
- SQL: 1 FP ≈ 10-15 SLOC

Однако важно понимать, что плотность сильно зависит от стиля кодирования, использования библиотек и генераторов кода.

Достоинства FP: независимость от языка, применимость на ранних этапах, возможность сравнения продуктивности команд. Недостатки: трудоёмкость подсчёта, субъективность при оценке сложности, слабая поддержка современных парадигм (компонентная, сервис-ориентированная архитектура).

Методика и порядок выполнения работы

Задание 1. Для требований из ЛР1:

1. Выделить все EI, EO, EQ, ILF, EIF и оценить их сложность.
2. Рассчитать UFP (нескорректированные функциональные точки).
3. Оценить 14 факторов настройки (VAF) и вычислить сумму.
4. Рассчитать скорректированные FP.
5. Перевести FP в строки кода для языка Java (средняя плотность 50 SLOC/FP).
6. Перевести FP в строки кода для Python (15 SLOC/FP).
7. Перевести FP в строки кода для C++ (80 SLOC/FP).
8. Сравнить размер в SLOC для трёх языков и сделать вывод.
9. Вычислить удельную производительность команды (FP на человеко-месяц).
10. Оценить, как изменится количество FP при добавлении трёх новых функций.

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Таблица с выделенными компонентами и их сложностью
3) Расчёт UFP
4) Заполненная таблица 14 факторов (VAF) с оценками
5) Расчёт скорректированных FP
6) Пересчёт в SLOC для двух языков программирования
7) Анализ полученного размера проекта.
8) Ответы на контрольные вопросы.

Контрольные вопросы

1. Для чего нужен метод функциональных точек?
2. В чём разница между внешним вводом и внешним запросом?
3. Что такое внутренний логический файл? Приведите пример.
4. Назовите 14 общих характеристик системы (GSC).
5. Как рассчитывается скорректированное количество функциональных точек?
6. Почему коэффициент VAF находится в диапазоне от 0,65 до 1,35?
7. Как перевести FP в строки кода?
8. Какие недостатки у метода функциональных точек?
9. В чём отличие FP от Use Case Points?
10. Как использовать FP для оценки трудоёмкости на ранних этапах?

Критерии оценивания

Параметр	Баллы
Правильность идентификации компонентов	6
Корректность расчёта UFP	4
Обоснованность оценок VAF	4
Точность пересчёта в SLOC	3
Выводы и оформление	3

Лабораторная работа 4

Оценка трудоемкости методом COCOMO II

Цель: Применить модель COCOMO II для расчета трудоемкости.

Теоретическое обоснование

COCOMO II (Constructive Cost Model) – это параметрическая модель оценки затрат на разработку ПО, разработанная Барри Бёмом в 1995 году как эволюция классической COCOMO 81. Модель предназначена для современных

методов разработки, включая повторное использование, автоматическую генерацию кода, компонентную разработку и спиральную модель жизненного цикла.

Основной входной параметр – **размер в тысячах строк кода (KSLOC)** или функциональные точки. COCOMO II использует три уровня точности:

- Базовая модель (Application Composition) – для прототипирования и разработки с использованием готовых компонентов. Оценка в объектных точках (Object Points).
- Ранняя модель проектирования (Early Design) – используется до завершения детального проектирования. Применяются 7 мультипликаторов затрат и масштабные факторы.
- Пост-архитектурная модель (Post-Architecture) – самая подробная, используется после утверждения архитектуры. Включает 17 драйверов затрат (Effort Multipliers, EM) и 5 масштабных факторов (Scale Factors, SF).

Базовая формула трудоёмкости (PM, Person-Months) для пост-архитектурной модели:

$$PM = A \times (KSLOC)^E \times \prod_{i=1}^{17} EM_i$$

где:

$A = 2,94$ (калибровочная константа)

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

$B = 1,01$ (базовый показатель)

Масштабные факторы (SF) отражают экономию или потерю масштаба:

- PREC (прецеденты) – насколько проект похож на предыдущие.
- FLEX (гибкость процесса) – насколько процесс адаптируем.
- RESL (архитектурный анализ рисков).
- TEAM (сплочённость команды).
- PMAT (зрелость процесса по CMMI).

Каждый SF оценивается от 1 (очень низкий) до 6 (очень высокий). Сумма SF обычно от 5 до 30, что даёт E от 1,01 до 1,26.

Драйверы затрат (EM) делятся на 4 группы:

- Продукт (RELY – требуемая надёжность, DATA – размер БД, CPLX – сложность продукта, RUSE – повторное использование, DOCU – документация).
- Платформа (TIME – ограничения времени выполнения, STOR – ограничения памяти, PVOL – нестабильность платформы).
- Персонал (ACAP – способности аналитиков, PCAP – способности программистов, PCON – текучесть кадров, AEXP – опыт в приложении, PEXP – опыт в платформе, LTEX – опыт в языке и инструментах).
- Проект (TOOL – использование инструментов, SITE – распределённость команды, SCED – сжатие сроков).

Каждый EM имеет значения от 0,71 (очень низкий) до 1,77 (очень высокий). Произведение EM даёт общий корректирующий коэффициент от 0,2 до 4,0.

После расчёта трудоёмкости вычисляется **календарная продолжительность (TDEV, Time to DEVelopment)**:

$$TDEV = C \times (PM)^D$$

где $C = 3,67, D = 0,28 + 0,01 \times \sum_{j=1}^{5} SF_j$ (для пост-архитектурной модели). TDEV измеряется в месяцах.

Рекомендуемое количество разработчиков = PM / TDEV, но оно не должно превышать определённого максимума из-за коммуникационных затрат.

COCOMO II также позволяет учитывать повторное использование через эквивалентный KSLOC (эквивалентные строки кода, включая модификацию интеграцию). Для автоматически сгенерированного кода учитывается только стоимость интеграции и тестирования (коэффициент 0,1-0,3 от размера).

Сравнение с PERT: COCOMO II даёт более объективную оценку, основанную на исторических данных, но требует калибровки под конкретную организацию. PERT лучше учитывает вариативность на уровне задач. Лучшие результаты достигаются при комбинации методов: COCOMO II для предварительной оценки, PERT для детального планирования.

Методика и порядок выполнения работы

Задание 1. Для SLOC из ЛР3 (на выбранном языке):

1. Применить базовую модель COCOMO II, выбрав масштабный фактор B.
2. Оценить 17 мультипликаторов затрат для промежуточной модели.
3. Рассчитать трудоёмкость PM в человеко-месяцах по промежуточной модели.
4. Вычислить календарную продолжительность TDEV.
5. Сравнить результат с PERT-оценкой из ЛР2 (процент расхождения).
6. Проанализировать влияние самого сильного и самого слабого драйвера затрат.
7. Оценить трудоёмкость при использовании автоматической генерации кода (понижение KSLOC на 30%).
8. Рассчитать необходимое среднее количество разработчиков (PM/TDEV).
9. Построить график зависимости PM от изменения масштабного фактора.
10. Предложить три способа снижения трудоёмкости на основе COCOMO II.

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Исходные данные: язык программирования, KSLOC из ЛР3
3) Выбранные масштабные факторы и расчёт B
4) Таблица драйверов затрат с обоснованием каждой оценки
5) Расчёт PM по базовой и промежуточной моделям
6) Расчёт TDEV и количества разработчиков
7) Сравнение с PERT-оценкой (таблица, график)
8) Выводы о расхождениях и их причинах.
9) Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое KSLOC и как он связан с FP?
2. Какие масштабные факторы (precedentedness, development flexibility и др.) влияют на показатель B?
3. Перечислите основные драйверы затрат COCOMO II.
4. По какой формуле рассчитывается трудоёмкость в базовой модели?
5. Как учитывается опыт команды в COCOMO II?
6. Что такое TDEV и от чего он зависит?
7. Почему COCOMO II считается более точным, чем PERT?
8. Какие ограничения есть у модели COCOMO II?
9. Как модель учитывает повторное использование компонентов?
10. Для каких типов проектов (малые, большие, Agile) COCOMO II применим лучше всего?

Критерии оценивания

Параметр	Баллы
Корректность пересчёта FP → KSLOC	3
Обоснованность выбора масштабных факторов	5
Полнота и аргументация мультипликаторов	7
Правильность расчётов PM и TDEV	5
Глубина сравнительного анализа	5

Лабораторная работа № 5

Формирование бюджета проекта разработки

Цель работы: Разработать бюджет проекта на основе рассчитанной трудоемкости.

Теоретическое обоснование

Бюджет ИТ-проекта – это финансовый план, определяющий затраты, необходимые для создания и внедрения программного продукта. Корректно составленный бюджет позволяет контролировать расходы, оценивать потребность в финансировании, рассчитывать точку безубыточности и служит основой для инвестиционного анализа.

Бюджет разработки делится на следующие категории:

1. Прямые затраты (Direct Costs) – расходы, непосредственно связанные с производством продукта:
    - Фонд оплаты труда (ФОТ) – заработная плата команды разработки с учётом всех отчислений. Рассчитывается как сумма по каждой роли: ФОТ=∑(ставка в месяц×длительность участия в месяцах). Ставки берутся рыночные (средние по региону или отрасли) или фактические. Важно включить не только «голую» зарплату, но и налоги (НДФЛ, страховые взносы – в РФ примерно 30% от начисленной зарплаты), бонусы, компенсации.
    - Лицензии на ПО – операционные системы, IDE, базы данных, инструменты тестирования, CI/CD (например, Jira, Confluence, GitLab Ultimate, IntelliJ IDEA Ultimate). Многие поставщики имеют специальные тарифы для стартапов.
    - Оборудование – рабочие станции, серверы, периферия. Учитывается как покупка, так и аренда (или амортизация, если срок службы > 1 года). Для облачных проектов – виртуальные машины, хранилища, сети.

- Коммуникационные услуги – интернет, телефония, почтовые сервисы.
- Облачная инфраструктура (IaaS/PaaS) для сред разработки, тестирования и интеграции (AWS, Azure, Google Cloud, Yandex Cloud). Оценивается исходя из потребляемых ресурсов: CPU, RAM, диск, трафик.
- Внешний аудит, тестирование безопасности – при необходимости.

2. Косвенные (накладные) расходы (Overhead) – затраты, которые нельзя прямо отнести на проект, но которые необходимы для функционирования организации:
   - Аренда офиса (или коворкинга)
   - Коммунальные услуги
   - Административно-управленческий персонал (бухгалтерия, HR, юристы) – доля их времени
   - Обучение сотрудников
   - Канцелярия, хозтовары
   - Командировки

   Накладные расходы обычно рассчитываются как процент от прямых затрат (от 20% до 50% в зависимости от структуры компании). Для стартапа без офиса процент может быть ниже.
3. Резервы на риски (Contingency) – определяются на основе анализа рисков (см. ЛР8). Типовой подход – процент от бюджета без резервов (5-20% для ИТ-проектов).
4. Капитальные затраты (CAPEX) и операционные (OPEX) – CAPEX включают покупку оборудования и лицензий на длительный срок, OPEX – аренду, зарплату, облачные услуги.

Процесс формирования бюджета:

1. Определить состав команды из оценки трудоёмкости (ЛР2/ЛР4) – роли, время участия.

2. Назначить ставки (рыночные данные: hh.ru, Glassdoor, отраслевые обзоры).
3. Рассчитать ФОТ с учётом налогов.
4. Оценить затраты на инфраструктуру, инструменты, связь.
5. Вычислить накладные расходы.
6. Добавить резерв на риски.
7. Сформировать сводный бюджет в виде таблицы с разбивкой по месяцам или этапам (создание графика освоения бюджета – cash flow).

Анализ структуры затрат позволяет выявить «узкие места». Обычно ФОТ составляет 60-80% бюджета разработки. Оптимизация может идти по путям:

- Использование open-source инструментов вместо платных.
- Привлечение удалённых сотрудников из регионов с меньшими ставками.
- Использование облачных сред с оплатой по факту (pay-as-you-go) вместо покупки серверов.
- Аутсорсинг непрофильных задач (например, тестирование, дизайн).

Аутсорсинг – передача части работ внешней компании. При аутсорсинге ставка выше (из-за маржи подрядчика), но отпадают накладные расходы на аренду, оборудование, управление. Сравнение должно быть по полной стоимости.

Бюджет утверждается спонсором проекта и служит базой для контроля (метод освоенного объёма – EVM).

Методика и порядок выполнения работы

Задание 1. На основе трудоёмкости из ЛР2 и/или ЛР4:

1. Определить состав команды (роли, количество человек, месяцы участия).
2. Назначить рыночные ставки для каждого члена команды (обосновать).

3. Рассчитать общий ФОТ за весь период разработки.
4. Оценить затраты на лицензии и инструменты (IDE, CI/CD, тестирование).
5. Оценить затраты на облачную инфраструктуру (AWS, Azure и т.п.) на период разработки.
6. Рассчитать накладные расходы (30% от ФОТ + прямые затраты).
7. Сформировать сводный бюджет в виде таблицы с разбивкой по месяцам.
8. Выявить статьи с наибольшим удельным весом и предложить оптимизацию.
9. Рассчитать бюджет при аутсорсинге (ставка выше, но без накладных).
10. Сравнить бюджет для двух вариантов состава команды (минимальный и рекомендуемый).

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Штатное расписание с указанием ролей, ставок и времени участия
3) Расчёт ФОТ с разбивкой по месяцам
4) Смета прямых затрат (лицензии, оборудование, связь, облако)
5) Расчёт накладных расходов
6) Сводный бюджет (таблица с итогами)
7) Анализ структуры затрат (диаграмма)
8) Предложения по оптимизации с количественной оценкой эффекта
9) Ответы на контрольные вопросы.

Контрольные вопросы

1. Какие статьи затрат относятся к прямым, а какие к косвенным?
2. Что входит в фонд оплаты труда разработчика?
3. Как рассчитать стоимость одного человеко-месяца?
4. Какие налоги и социальные отчисления нужно учесть в РФ?

5. Что такое накладные расходы и какой типовой процент используется?
6. Как учесть затраты на оборудование (амортизация или аренда)?
7. Какие модели лицензирования ПО существуют?
8. Что такое резерв на непредвиденные расходы (контингенс)?
9. Как составить бюджет с разбивкой по этапам (cash flow)?
10. Какие методы оптимизации бюджета вы знаете?

Критерии оценивания

Параметр	Баллы
Обоснованность состава команды и ставок	5
Полнота учёта затрат	6
Корректность расчёта ФОТ и накладных	5
Качество анализа структуры	4
Реалистичность предложений по оптимизации	5

Лабораторная работа № 6

Расчет показателей экономической эффективности ИТ-проекта

Цель: Оценить коммерческую эффективность проекта.

Теоретическое обоснование

Оценка экономической эффективности (коммерческой) необходима для принятия решения об инвестировании в ИТ-проект. Основой является **прогноз денежных потоков (Cash Flow)** на горизонте 3-5 лет, поскольку большинство ИТ-проектов имеют высокие начальные затраты и отложенный доход.

Денежный поток (CF, Cash Flow) в период t рассчитывается как:

$$CF_t = \text{Доходы}_t - \text{Операционные расходы}_t - \text{Налоги}_t - \text{Инвестиции}_t$$

где инвестиции включают затраты на разработку (из ЛР5). Операционные расходы после запуска могут включать хостинг, поддержку, маркетинг.

Ставка дисконтирования (r) отражает альтернативную стоимость капитала и риск проекта. Для ИТ-проектов типичные значения: 10-15% для надёжных внутренних проектов, 20-30% для стартапов. Ставка может быть рассчитана по модели CAPM (Capital Asset Pricing Model) или методу кумулятивного построения.

Основные показатели:

1. **NPV (Net Present Value)** – чистая приведённая стоимость:

$$NPV = \sum_{t=0}^{T} \frac{CF_t}{(1+r)^t}$$

Правило принятия: NPV > 0 – проект приемлем, чем выше, тем лучше. NPV учитывает стоимость денег во времени и весь горизонт планирования.

2. **IRR (Internal Rate of Return)** – ставка дисконтирования, при которой NPV=0:

$$\sum_{t=0}^{T} \frac{CF_t}{(1+IRR)^t} = 0$$

IRR сравнивают с барьерной ставкой (r). Если **IRR > r**, проект эффективен. Недостатки: возможны множественные IRR при знакопеременных потоках, не учитывает масштаб.

3. **ROI (Return on Investment)** – рентабельность инвестиций:

$$ROI = \frac{\sum_{t=0}^{T} CF_t - \text{Инвестиции}}{\text{Инвестиции}} \times 100\%$$

Простой и понятный показатель, но не учитывает дисконтирование.

4. **Срок окупаемости (Payback Period, PP)** – время, за которое накопленный денежный поток становится неотрицательным. Дисконтированный срок окупаемости (DPP) учитывает дисконтирование:

$$\sum_{t=0}^{DPP} \frac{CF_t}{(1+r)^t} \geq 0$$

Анализ чувствительности позволяет оценить устойчивость проекта к изменениям ключевых параметров (цена, объём продаж, затраты на разработку, задержки выхода на рынок). Для каждого параметра задаются отклонения (например, ±20%) и пересчитывается NPV. Результат представляется в виде торнадо-диаграммы или графика «NPV от изменения параметра». Наиболее чувствительные параметры требуют особого внимания и мониторинга.

Также проводят сценарный анализ: пессимистичный (снижение доходов на 30%, рост затрат на 20%), реалистичный, оптимистичный (рост доходов на 30%). Для каждого сценария рассчитывают NPV и вероятность наступления.

Анализ безубыточности определяет точку, в которой доходы покрывают затраты (без учёта дисконтирования). Формула для подписной модели:

$$\text{Количество клиентов} = \frac{\text{Постоянные затраты}}{\text{Средний доход на клиента - переменные затраты на клиента}}.$$

На основе расчётов готовится инвестиционное заключение: рекомендуется ли проект к финансированию, какой запас прочности (например, на сколько процентов может снизиться выручка при сохранении положительного NPV).

Методика и порядок выполнения работы

Задание 1. Для продукта из ЛР1 с использованием бюджета из ЛР5:

1. Выбрать модель монетизации (подписка, лицензия, реклама, транзакции).
2. Сделать прогноз доходов на 3 года (пессимистичный, реалистичный, оптимистичный).
3. Построить прогнозный отчёт о движении денежных средств (Cash Flow).
4. Выбрать ставку дисконтирования (r) и обосновать её (например, 15%).
5. Рассчитать NPV для реалистичного сценария.

6. Рассчитать IRR (подбором или с помощью формулы).
7. Рассчитать ROI и срок окупаемости (простой и дисконтированный).
8. Провести анализ чувствительности NPV к изменению цены на ±20%.
9. Провести анализ чувствительности к изменению объёма продаж на ±20%.
10. Сделать вывод о целесообразности проекта (приемлемость показателей).

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Обоснование модели монетизации и прогноз доходов (три сценария).
3) Таблица денежных потоков по годам (доходы, операционные расходы, налоги, CF).
4) Расчёт NPV, IRR, ROI, срока окупаемости.
5) Два графика анализа чувствительности (цена, объём).
6) Заключение о коммерческой эффективности.
7) Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое временная стоимость денег и почему её учитывают?
2. Как выбрать ставку дисконтирования для ИТ-проекта?
3. Формула расчёта NPV.
4. Какое значение NPV считается приемлемым?
5. В чём разница между IRR и ROI?
6. Что такое срок окупаемости и его недостатки?
7. Как проводится анализ чувствительности?
8. Какие ещё показатели эффективности существуют (DPP, PI, MIRR)?
9. Как учесть инфляцию в денежных потоках?
10. Почему для стартапов часто требуются более высокие ставки дисконтирования??

Критерии оценивания

Параметр	Баллы
Реалистичность прогноза доходов	5
Корректность расчёта денежных потоков	5
Правильность вычисления всех финансовых показателей	7
Качество анализа чувствительности	5
Обоснованность вывода	3

Лабораторная работа № 7

Разработка бизнес-модели и модели монетизации

Цель: Спроектировать бизнес-модель программного продукта.

Теоретическое обоснование

Бизнес-модель – это описание того, как организация создаёт, доставляет и захватывает ценность. Для ИТ-продуктов наиболее популярным инструментом является канва бизнес-модели (Business Model Canvas) А. Остервальдера и И. Пинье. Она состоит из 9 блоков, которые охватывают четыре основные сферы: клиенты, предложение, инфраструктура, финансовая устойчивость.

Блоки канвы:

1. **Ценностные предложения (Value Propositions)** – набор продуктов и услуг, создающих ценность для конкретного сегмента. Формулируется как выгода, решение проблемы, снижение рисков, удобство.
2. **Потребительские сегменты (Customer Segments)** – группы людей или организаций, которым компания намерена продавать. Для ИТ: массовый рынок (B2C), нишевый рынок, многосторонние платформы.
3. **Каналы сбыта (Channels)** – как продукт доставляется клиенту: веб-сайт, маркетплейсы (App Store, Google Play), прямые продажи, партнёры.
4. **Взаимоотношения с клиентами (Customer Relationships)** – тип взаимодействия: персональная поддержка, самообслуживание, автоматизированные сервисы, сообщества, совместное создание.
5. **Потоки доходов (Revenue Streams)** – как компания зарабатывает: продажа лицензий, подписка (SaaS), реклама, комиссия с транзакций, фримиум (freemium), платные дополнения (in-app purchases), лизинг, краудфандинг.
6. **Ключевые ресурсы (Key Resources)** – что необходимо для

работы: человеческие (разработчики, маркетологи), материальные (серверы, офис), интеллектуальные (патенты, алгоритмы), финансовые.

7. **Ключевые виды деятельности (Key Activities)** – что компания делает ежедневно: разработка, маркетинг, поддержка клиентов, анализ данных.
8. **Ключевые партнёры (Key Partnerships)** – внешние организации: поставщики облачных услуг, интеграторы, дистрибьюторы, платёжные системы.
9. **Структура издержек (Cost Structure)** – постоянные и переменные затраты, соответствующие бюджету из ЛР5.

Модель монетизации – это подмножество бизнес-модели, детально описывающее, как продукт приносит деньги. Основные стратегии:

- **Подписка (Subscription/SaaS)** – регулярные платежи (ежемесячные, ежегодные). Преимущество: предсказуемый доход, низкий порог входа. Примеры: Netflix, Spotify, Microsoft 365.
- **Freemium** – базовый функционал бесплатно, расширенный – за плату. Конверсия обычно 2-5%. Требует больших объёмов бесплатных пользователей.
- **Рекламная модель** – доход от показа рекламы. Эффективна при большом трафике. Примеры: Google, Яндекс, многие мобильные игры.
- **Транзакционная модель** – комиссия с каждой сделки (маркетплейсы, платёжные сервисы). Пример: Airbnb, eBay.
- **Лицензионная модель** – единоразовая плата за право использования (on-premise). Всё ещё популярна в B2B, но уступает подписке.
- **Плата за использование (Pay-per-use)** – например, API-вызовы, объём переданных данных. Примеры: Twilio, AWS.
- **Внутренняя модель** – продукт не продаётся, а создаётся для внутренних нужд компании. Экономическая эффективность оценивается через ROI или улучшение операционных показателей.

Ценовая стратегия:

- Ценообразование на основе издержек (cost-plus): цена = себестоимость + наценка.
- Ценообразование на основе ценности (value-based): цена определяется воспринимаемой ценностью для клиента.
- Цены проникновения – низкая цена для захвата рынка.
- Скимминг – высокая цена для инновационного продукта, затем снижение.

Единичная экономика (unit economics):

- **LTV (Lifetime Value)** – суммарная прибыль от одного клиента за всё время использования. Для подписки: LTV = средний платёж в месяц × средняя длительность подписки.
- **CAC (Customer Acquisition Cost)** – стоимость привлечения одного платящего клиента (маркетинг, реклама, скидки).
- Здоровое соотношение LTV / CAC > 3.
- **Срок окупаемости CAC** – время, за которое клиент генерирует доход, равный затратам на его привлечение (должен быть менее 12 месяцев).

Анализ конкурентов включает изучение их тарифных планов, функциональности бесплатных версий, ценовых акций, каналов привлечения. На основе этого обосновывается собственная модель.

Методика и порядок выполнения работы

Задание 1. Для продукта из ЛР1:

1. Заполнить канву бизнес-модели (все 9 блоков).
2. Описать ценностное предложение по шаблону Value Proposition Canvas.
3. Выбрать модель монетизации и обосновать тремя аргументами.
4. Разработать 3 тарифных плана (например, Starter, Pro, Enterprise).
5. Проанализировать конкурентов: их модели монетизации и цены.

6. Обосновать выбранную ценовую стратегию (цена проникновения, скимминга, рыночная).
7. Рассчитать LTV (Lifetime Value) клиента для подписной модели.
8. Рассчитать CAC (Customer Acquisition Cost) и соотношение LTV/CAC.
9. Оценить точку безубыточности по количеству клиентов.
10. Предложить способы увеличения LTV (апсейл, кросс-сейл).

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Заполненная канва бизнес-модели (таблица или графический вид).
3) Описание ценностного предложения.
4) Модель монетизации: выбор и обоснование.
5) Таблица тарифных планов с описанием функций.
6) Анализ конкурентов (2-3 компании, сравнение).
7) Расчёт LTV, CAC, точки безубыточности.
8) Вывод о жизнеспособности бизнес-модели.
9) Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое бизнес-модель и зачем она нужна?
2. Перечислите 9 блоков канвы бизнес-модели А. Остервальдера.
3. Чем отличается B2B от B2C модели монетизации?
4. Что такое freemium-модель и для каких продуктов она подходит?
5. Как рассчитать LTV и CAC?
6. Какие существуют типы лицензий на ПО (проприетарные, открытые)?
7. Что такое единичная экономика (unit economics)?
8. Какие каналы привлечения клиентов наиболее эффективны для SaaS?
9. Как провести конкурентный анализ моделей монетизации?
10. В чём разница между ценообразованием «cost-plus» и value-based?

Критерии оценивания

Параметр	Баллы
Полнота и логичность канвы	6
Обоснованность выбора модели монетизации	5
Качество тарифной сетки и ценовой стратегии	5
Корректность расчётов LTV/CAC	5
Глубина конкурентного анализа	4

Лабораторная работа № 8

Анализ рисков проекта и разработка мер реагирования

Цель: Идентифицировать экономические риски и разработать план управления ими.

Теоретическое обоснование

Риск – это неопределённое событие или условие, которое в случае наступления оказывает положительное или отрицательное влияние на цели проекта (сроки, бюджет, содержание, качество). Управление рисками – систематический процесс идентификации, анализа, планирования реагирования и мониторинга.

Процесс управления рисками по **PMBOK**:

1. Планирование управления рисками – определение подходов, инструментов, уровней толерантности.
2. Идентификация рисков – выявление возможных событий. Методы: мозговой штурм, метод Дельфи, интервью, анализ допущений, контрольные списки (risk breakdown structure), SWOT-анализ, анализ сценариев.
3. Качественный анализ – оценка вероятности наступления (0-100% или шкала 1-5) и влияния на проект (обычно 1-5 по каждому из критериев: затраты, сроки, качество, репутация). Риски ранжируются по матрице вероятность-влияние. Определяются риски «красной зоны» – требующие немедленного реагирования.
4. Количественный анализ – численная оценка возможных последствий. Методы:
    - Ожидаемая стоимость (Expected Monetary Value, EMV) = вероятность × ущерб. Сумма EMV по всем рискам даёт резерв.
    - Анализ чувствительности (диаграмма «торнадо»).
    - Моделирование Монте-Карло – распределение вероятностей для дат завершения и бюджета.
5. Планирование реагирования – выбор стратегии для каждого ключевого риска:
    - Избегание (Avoid) – устранение причины риска (например, выбор

другой технологии, исключение функции).
    - Минимизация (Mitigate) – снижение вероятности или последствий (дополнительное тестирование, обучение команды, создание прототипов).
    - Передача (Transfer) – передача ответственности третьей стороне (страхование, аутсорсинг, гарантийные обязательства).
    - Принятие (Accept) – активное (создание резерва) или пассивное (признание риска и отслеживание).
6. Мониторинг и контроль рисков – регулярный пересмотр реестра рисков, отслеживание триггеров, аудит реагирования.

Классификация рисков для ИТ-проектов:

- Технические: сложность интеграции, недостаточная производительность, уязвимости безопасности, смена технологий, отказ оборудования.
- Организационные/управленческие: потеря ключевых сотрудников, конфликты в команде, плохая коммуникация, нечёткие требования, изменение объёма работ (scope creep).
- Рыночные: появление более сильного конкурента, снижение спроса, неверно выбранная модель монетизации, регуляторные изменения (GDPR, закон о персональных данных).
- Финансовые: рост курса валют (если закупки в валюте), инфляция, недофинансирование, задержка платежей от клиентов.
- Правовые: нарушение авторских прав, проблемы с лицензиями, судебные иски.

Расчёт резерва на риски (контингенс):

- Простой метод: процент от базового бюджета (5-15%).
- Метод ожидаемой стоимости: резерв = Σ (вероятность_i × влияние_i в деньгах).
- Метод имитационного моделирования (Монте-Карло) даёт распределение и позволяет установить резерв на заданном

доверительном интервале (например, 80%).

План действий при наступлении риска (кризисный план) должен содержать: описание триггера (сигнала), ответственного, набор шагов, альтернативные ресурсы, бюджет на реагирование.

Мониторинг рисков осуществляется на регулярных совещаниях, с использованием реестра рисков – таблицы с полями: ID, название, категория, вероятность, влияние, ранг, стратегия, план действий, ответственный, статус, резерв.

Важно различать риски (неопределённые события) и проблемы (уже случившиеся). Риск-менеджмент – это проактивная деятельность.

Методика и порядок выполнения работы

Задание 1. Для проекта из предыдущих ЛР:

1. Идентифицировать 10-15 рисков (минимум 3 по каждой категории).
2. Провести качественный анализ: вероятность и влияние (матрица).
3. Построить матрицу "вероятность-влияние" и выделить критические риски.
4. Для трёх ключевых рисков разработать меры реагирования (минимизацию).
5. Для одного риска предложить передачу (страхование или аутсорсинг).
6. Рассчитать денежный резерв на риски (методом ожидаемой стоимости).
7. Оценить влияние рисков на бюджет (в процентах от бюджета).
8. Разработать план мониторинга рисков (контрольные точки, метрики).
9. Составить план действий при наступлении риска (кризисный план).
10. Сравнить два сценария: с учётом резервов и без – на показатели NPV.

Содержание отчета и его форма

1) Титульный лист с названием ЛР, ФИО, вариант.
2) Реестр рисков (таблица с ID, названием, категорией)
3) Качественный анализ (вероятность, влияние, ранг)
4) Матрица вероятности и влияния (рисунок)
5) План реагирования для 5 ключевых рисков (стратегия, действия, ответственный)

6) Расчёт резерва на риски (в денежном выражении)
7) План мониторинга (контрольные события, частота).
8) Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое риск проекта и чем он отличается от неопределённости?
2. Какие методы идентификации рисков существуют?
3. Как оценить вероятность риска и величину ущерба?
4. Что такое матрица рисков (вероятность × влияние)?
5. Какие стратегии реагирования на риски вы знаете?
6. Что такое резерв на риски (контингенс) и как его рассчитать?
7. В чём разница между качественным и количественным анализом рисков?
8. Какие риски наиболее характерны для ИТ-проектов?
9. Как часто нужно пересматривать реестр рисков?
10. Что такое "риск-аппетит" организации?

Критерии оценивания

Параметр	Баллы
Полнота и релевантность реестра рисков	5
Корректность качественного анализа	5
Реалистичность и эффективность мер реагирования	7
Правильность расчёта резерва	4
Качество плана мониторинга	4

Лабораторная работа № 9

Подготовка и защита итогового проекта

Цель: Представить комплексное экономическое обоснование проекта.

Теоретическое обоснование

Итоговая работа по курсу экономики ИТ-проектов представляет собой **комплексное экономическое обоснование (бизнес-план)** разрабатываемого программного продукта. В отличие от отдельных лабораторных работ, здесь требуется синтез всех полученных знаний и согласование данных между разделами.

Цель итогового проекта – продемонстрировать способность:

- обосновать идею продукта с рыночной точки зрения;
- выполнить оценку трудоёмкости несколькими методами и сравнить результаты;
- составить реалистичный бюджет;
- разработать бизнес-модель и модель монетизации;
- рассчитать ключевые финансовые показатели;
- идентифицировать риски и предложить меры реагирования;
- представить результаты в форме, доступной для инвесторов или руководителей.

Структура итогового отчёта (рекомендуемая):

- Введение – актуальность, цель работы, объект (продукт), методы.
- Глава 1. Продукт и требования – описание идеи, функциональные и нефункциональные требования, целевая аудитория, MVP, позиционирование (из ЛР1).
- Глава 2. Оценка трудоёмкости и размера – декомпозиция, PERT, функциональные точки, COCOMO II; сравнение результатов, выбор обоснованной оценки для дальнейшего использования.
- Глава 3. Бюджет разработки – команда, ставки, ФОТ, прямые и накладные расходы, резервы, сводный бюджет, анализ структуры затрат.
- Глава 4. Бизнес-модель и монетизация – канва бизнес-модели, модель монетизации, тарифные планы, LTV/CAC, конкурентный анализ.
- Глава 5. Финансовые показатели – прогноз доходов на 3-5 лет, денежные

потоки, NPV, IRR, ROI, срок окупаемости, анализ чувствительности, сценарный анализ.

- Глава 6. Риски – реестр рисков, качественный и количественный анализ, резерв, план реагирования, мониторинг.
- Заключение – итоговый вывод о целесообразности проекта, запас прочности, рекомендации инвестору.
- Список литературы – нормативные, методические и научные источники.
- Приложения – подробные расчётные таблицы, сетевые диаграммы, матрицы рисков, графики.

Требования к согласованности данных (критически важны):

- Трудоёмкость в человеко-месяцах (из PERT или COCOMO) должна соответствовать ФОТ в бюджете.
- Срок разработки (TDEV) должен совпадать с длительностью, указанной в календарном плане.
- Бюджет разработки является начальными инвестициями в расчёте NPV.
- Модель монетизации определяет прогноз доходов.
- Анализ чувствительности должен включать риски, идентифицированные в Главе 6.

Подготовка презентации – 10-12 слайдов, отражающих суть каждого раздела. На защите важно не просто пересказывать слайды, а демонстрировать причинно-следственные связи, аргументировать допущения, отвечать на вопросы о компромиссах (например, почему выбран именно такой тариф, почему ставка дисконтирования 20%, как повлияет задержка на 2 месяца)/

Типичные вопросы на защите (кроме контрольных из ЛР1-ЛР8):

- Какие альтернативные варианты вы рассматривали и почему от них отказались?
- Как изменится NPV, если выход на рынок задержится на полгода?
- Какое максимальное снижение цены вы можете допустить без убыточности?
- Какие метрики вы будете отслеживать после запуска, чтобы подтвердить экономическую модель?

- Какие допущения в COCOMO II являются наиболее спорными в вашем проекте?

Оценка защиты включает как качество материалов (отчёт, презентация), так и умение вести дискуссию, защищать свою позицию, признавать ограничения. Итоговая оценка выставляется по совокупности баллов за все разделы и защиту.

Методика и порядок выполнения работы

Задание 1. Студент выбирает один сквозной проект из предложенных тем (или свою):

1. Образовательная платформа с ИИ-тьютором.
2. Сервис для автоматизации закупок малого бизнеса.
3. Мобильное приложение для здоровья и фитнеса (подписка).
4. B2B-платформа для видеоконференций с интеграцией CRM.
5. Маркетплейс цифровых товаров (шаблоны, фото, музыка).
6. Система управления инцидентами для IT-службы.
7. Платформа для краудфандинга творческих проектов.
8. Приложение для бронирования столиков в ресторанах с кэшбэком.
9. Сервис проверки контрагентов по открытым данным.
10. Корпоративный портал знаний на основе Wiki.

Содержание отчета и его форма

1) Титульный лист, содержание, введение
2) Глава 1. Описание продукта и требований (ЛР1)
3) Глава 2. Оценка трудоёмкости (ЛР2, ЛР3, ЛР4) – сравнение методов
4) Глава 3. Бюджет разработки (ЛР5)
5) Глава 4. Бизнес-модель и монетизация (ЛР7)
6) Глава 5. Финансовые показатели (ЛР6) – с учётом рисков
7) Глава 6. Анализ рисков (ЛР8)
8) Заключение (общий вывод об экономической целесообразности)
9) Список литературы, приложения (расчётные таблицы, диаграммы).

Требования к презентации (10-12 слайдов)

1. Продукт и целевая аудитория
2. Ключевые функциональные требования
3. Итоги оценки трудоёмкости (сравнение методов)
4. Бюджет (структура затрат)
5. Модель монетизации и прогноз доходов
6. Финансовые показатели (NPV, IRR, ROI, окупаемость)
7. Ключевые риски и резервы
8. Заключение (инвестиционная привлекательность)

Вопросы для защиты работы

1. Какое у вашего продукта уникальное ценностное предложение?
2. Почему вы выбрали именно эту модель монетизации?
3. Какая оценка трудоёмкости оказалась самой точной и почему?
4. Какой драйвер затрат в COCOMO II повлиял сильнее всего?
5. Как вы обосновали ставку дисконтирования при расчёте NPV?
6. Какие риски вы считаете самыми критичными и как их минимизировать?
7. Каков запас прочности вашего проекта (например, снижение выручки на сколько % допускается)?
8. Как изменится срок окупаемости при увеличении бюджета на 20%?
9. Какие альтернативные бизнес-модели вы рассматривали и от чего отказались?
10. Каков следующий шаг после успешного запуска (масштабирование, новые рынки)?

Критерии оценивания

Параметр	Баллы
Качество и полнота итогового отчёта	8
Логическая связность и согласованность данных	6
Качество презентации (дизайн, структура, наглядность)	5
Умение отвечать на вопросы (аргументированность)	6
Соблюдение регламента (10 мин доклад)	2

Общее впечатление и профессиональный подход	3

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное
автономное образовательное
учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К САМОСТОЯТЕЛЬНОЙ РАБОТЕ

«Экономика программной инженерии»

Направление подготовки 09.03.04 «Программная инженерия» Направленность (профиль) «Инженерия сложных программных систем»
Квалификация выпускника: бакалавр

Ставрополь

2026

ВВЕДЕНИЕ

Цель освоения дисциплины

Формирование у студентов системного понимания экономических аспектов разработки программных продуктов, включая методы оценки трудоемкости и стоимости, бюджетирование, ценообразование, налогообложение в ИТ-сфере, а также развитие практических навыков экономического обоснования проектных решений и управления ресурсами в программной инженерии.

Задачами обучения являются:

изучение особенностей программного продукта как товара на профильном рынке и типов ИТ-компаний;

освоение современных методов и моделей оценки трудоемкости и стоимости разработки ПО (Functional Points, COCOMO II, PERT);

формирование навыков формирования бюджета проекта разработки и его оптимизации;

изучение метрик финансового анализа (ROI, NPV, IRR) и методов оценки экономической эффективности ИТ-проектов;

освоение стратегий ценообразования, моделей монетизации и лицензирования программных продуктов;

изучение правовых аспектов: создание и регистрация ИТ-компании, налогообложение, госзакупки, защита интеллектуальной собственности;

формирование навыков документирования и презентации экономических обоснований проектов.

Знания, полученные в результате освоения дисциплины «Экономика программной инженерии» могут быть использованы при подготовке отчетов по преддипломной практике и выпускной квалификационной работы.

В ходе изучения дисциплины «Экономика программной инженерии» у обучающегося формируются набор следующих компетенций:

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов
ПК-1 – Способен осуществлять управление программными проектами: планировать, контролировать сроки и ресурсы, применять системы контроля версий и инструменты управления задачами	ИД-1$_{ПК-1}$ Применяет методы оценки трудоемкости (PERT, функциональные точки, COCOMO II) для планирования сроков и ресурсов программного проекта ИД-4$_{ПК-1}$ Анализирует план-фактные отклонения по срокам и бюджету, предлагает корректирующие действия	Способен применять методы оценки трудоемкости (PERT, функциональные точки, COCOMO II) для планирования сроков и ресурсов программного проекта Способен анализировать план-фактные отклонения по срокам и бюджету, предлагать корректирующие действия
ПК-9 – Умеет оформлять техническую документацию, готовить презентации и отчеты, вести деловую коммуникацию на русском и иностранном языках в профессиональной среде	ИД-1$_{ПК-9}$ Оформляет техническую документацию (ТЗ, пояснительная записка, руководство пользователя, API-спецификация) в форматах Markdown, OpenAPI, DocBook в соответствии с ГОСТ	Способен оформлять экономические обоснования проектов, включая расчеты трудоемкости, бюджета и финансовых показателей
ПК-10 – Готов соблюдать правовые нормы, стандарты и требования информационной безопасности, участвовать в разработке нормативной и технической	ИД-1$_{ПК-10}$ Применяет основные правовые нормы (лицензирование ПО, авторское право, ФЗ-152, ФЗ-149) и стандарты ИБ (ГОСТ Р ИСО/МЭК 27001, ГОСТ Р 56545-2015) при разработке и эксплуатации ПО	Способен учитывать правовые нормы при выборе организационно-правовой формы и системы налогообложения для ИТ-компании

1. ОРГАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Самостоятельная работа студентов является важнейшим условием формирования научного способа познания. Она проводится накануне каждого лабораторного занятия и включает изучение необходимого для выполнения лабораторных работ и подготовки к практическим занятиям теоретического материала, а также подготовки отчетов по выполненным лабораторным и практическим занятиям. Подготовленный материал оформляется в виде отчета. Самостоятельные занятия (СЗ) являются одной из активных форм обучения. Самостоятельные занятия по дисциплине «Экономика программной инженерии» имеют целью: - закрепить и углубить знания, полученные студентами на лекциях и в процессе лабораторных и практических занятий; - привить практические навыки в оформлении организационных, распорядительных и информационно-справочных документов. Самостоятельные занятия проводятся в специализированных аудиториях, оборудованных средствами вычислительной техники и возможностью пользования Internet-ресурсами, в библиотеке. Предлагаемые методические рекомендации содержат информацию для студентов, необходимую при подготовке и проведении лабораторных занятий по дисциплине «Экономика программной инженерии».

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Самостоятельная работа по дисциплине «Экономика программной инженерии» состоит из двух частей:

1. Самостоятельное изучение вопросов;
2. Самостоятельная разработка дизайн проекта на базе умений и навыков,полученных на лабораторных работах.

3. СОДЕРЖАНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

3.1. Темы для самостоятельного изучения

1. Правовые и налоговые аспекты в ИТ-бизнесе.
2. Ценообразование и монетизация программных продуктов.
3. Экономические модели Open Source.

В процессе самостоятельного изучения вопросов курса студент изучает предложенные их по рекомендуемому преподавателем списку литературы и предоставляет преподавателю краткий конспект, по которому в дальнейшем проводится индивидуальное собеседование.

На зачетной неделе студент должен предоставить готовый проект.

4. ОРГАНИЗАЦИЯ КОНТРОЛЯ ЗНАНИЙ, ПОЛУЧЕННЫХ ПРИ ВЫПОЛНЕНИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

В процессе собеседования и защиты самостоятельного проекта:
Оценка «отлично» выставляется студенту, если теоретическое содержание вопросов самостоятельного изучения освоено полностью, без пробелов; исчерпывающе, последовательно, четко и логически стройно излагает материал; свободно справляется с задачами, вопросами и другими видами применения знаний; использует в ответе дополнительный материал все предусмотренные программой задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному; анализирует полученные результаты; проявляет самостоятельность при выполнении заданий.

Оценка «хорошо» выставляется студенту, если теоретическое и практическое содержание вопросов самостоятельной работы освоено полностью, необходимые практические компетенции в основном сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения достаточно высокое. Студент твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос.

Оценка «удовлетворительно» выставляется студенту, если теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, большинство предусмотренных программой заданий выполнено, но в них имеются ошибки, при ответе на поставленный вопрос студент допускает неточности, недостаточно правильные формулировки, наблюдаются нарушения логической последовательности в изложении программного материала. Практические навыки освоены частично.

Оценка «неудовлетворительно» выставляется студенту, если он не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы, необходимые практические компетенции не сформированы, большинство предусмотренных программой обучения учебных заданий не выполнено, качество их выполнения оценено числом баллов, близким к минимальному.

5. РЕКОМЕНДАЦИИ ПО РАБОТЕ С ЛИТЕРАТУРОЙ И ИСТОЧНИКАМИ

5.1 Рекомендации студентам по организации работы с литературой и источниками Изучение литературы и источников необходимо начинать с прочтения соответствующих глав учебных изданий, учебных пособий или литературы, рекомендованной в качестве основной или дополнительной по дисциплине «Экономика программной инженерии», которые прямо или косвенно относятся к отрабатываемой теме. Изучая литературу и источники, студенту рекомендуется вести краткий конспект. Однако не следует переписывать все содержание изучаемой темы, нужно выписывать лишь основные идеи и главные на ваш взгляд мысли. В отдельных случаях, когда встречаются важные определения, понятия, необходимый фактический материал и примеры, статистическая информация, имеющие отношение к изучаемой теме, студенту следует выписать их в виде цитат с полным указанием библиографических источников. Конспектирование рекомендуемой литературы и источников необходимо вести с распределением собранных материалов по отдельным главам и параграфам согласно учебно-тематическому плану. Необходимо выписывать все выходные данные по используемой литературе и источникам. Важным этапом при работе с рекомендуемой литературой иисточниками является изучение законодательных и нормативных актов федерального, регионального, местного и ведомственного уровней. При изучении Указов Президента РФ, Законов и Кодексов РФ, постановлений, положений, рекомендаций и т.д., студент должен выяснить все изменения и дополнения, которые могли быть внесены после их выхода в свет. Основой технологии интенсификации обучения на платформе цифровых образовательных технологий являются учебно-иллюстрационные материалы (опорный конспект) по дисциплине «Экономика программной инженерии».