Университет ИТМО

Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 Программная инженерия
Дисциплина «Экономика программной инженерии»

Отчет

По лабораторной работе №1

Выполнили:
Белогаев Д. В.
Кузнецов М. А.

Преподаватель:
Машина Е. А.

Санкт-Петербург, 2023 г.

Задание

Вариант задания: https://taiga.io

Для выданного веб-проекта:

1. Сформировать набор функциональных требований для разработки проекта.
2. Оценить трудоемкость разработки проекта наивным методом.
3. Оценить трудоемкость разработки проекта методом PERT (Project Evaluation and Review Technique**).** Нарисовать сетевую диаграмму взаимосвязи работ и методом критического пути рассчитать минимальную продолжительность разработки. Предложить оптимальное количество разработчиков и оценить срок выполнения проекта.
4. Оценить размер проекта методом функциональных точек, затем, исходя из предположения, что собранной статистики по завершенным проектам нет, рассчитать трудоемкость методом COCOMO II (Обновленная таблица количества строк на точку для разных языков программирования)
5. Оценить размер проекта методом оценки вариантов использования (Use Case Points). Для расчета фактора продуктивности PF использовать любой свой завершенный проект с известными временными трудозатратами, оценив его размер методом UCP.
6. Сравнить полученные результаты и сделать выводы.

Функциональные требования

Preparations:

№	Название	Описание	Оценка, мин. /чел. час	Оценка, вероятн. /чел. час	Оценка, макс. /чел. час
1.	Прототип сайта	Разработка и составление прототипа для open-source решения для организации команды по методикам scrum/kanban	25	35	40
2.	Определение стека	Определение стека разработки решения	5	5	10
3.	Документно составляющая	Проработка всех нормативных вещей	50	55	60
4.	Аренда сервера	Проведение необходимых мероприятий по поиску провайдера и анализу требуемой	25	35	50

		функциональности для поиска подходящих вариантов			

Frontend:

№	Название	Описание	Оценка, мин. /чел. час	Оценка, вероятн. /чел. час	Оценка, макс. /чел. час
1.	Меню авторизации	Разработка интерфейса логина\регистрации	15	20	25
2.	Поддержка английского и испанского языков	Добавление поддержки переключения языка отображения контента	20	20	30
3.	Раздел Cloud	Основной компонент, необходимо реализовать поддержку плана Cloud-хостинга, в котором пользователи выбирают taiga.io в качестве хостинг провайдера своего проекта, и Self-Hosted, где пользователи сами отвечают за хостинг. Предусматривает наличие форм обратной связи, выбор нескольких планов.	50	60	70
4.	Компоненты Agile	Создание главных компонентов для организации командного пространства: Kanban, Scrum, Issues, Dashboards, Integrations. Предусматривает наличие форм обратной связи.	60	70	70
5.	Раздел Community	Реализация небольшого пространства, где пользователи смогут делиться своими мыслями и помогать друг другу, обсуждать вопросы, делать гайды.	50	70	80
6.	Интеграция	Добавить удобную	5	5	10

	соц. сетей	интеграцию с другими пространствами и соц. сетями проекта и партнерскими проектами			
7.	Нормативно-правовой раздел	Создание раздела со всеми необходимыми документами.	5	5	10

Backend:

№	Название	Описание	Оценка, мин. /чел. час	Оценка, вероятн. /чел. час	Оценка, макс. /чел. час
1.	Настройка БД, СХД	Проведение необходимых действий и мероприятий для разработки решения по хранению данных проектов, пользователей.	40	50	55
2.	Авторизация	Разработка и создание авторизации в сервисе	20	25	30
3.	Бизнес-логика Cloud	Разработка бизнес-логики раздела Cloud	60	70	75
4.	Бизнес-логика компонентов Agile	Разработка бизнес-логики компонентов Scrum	70	80	90

Testing:

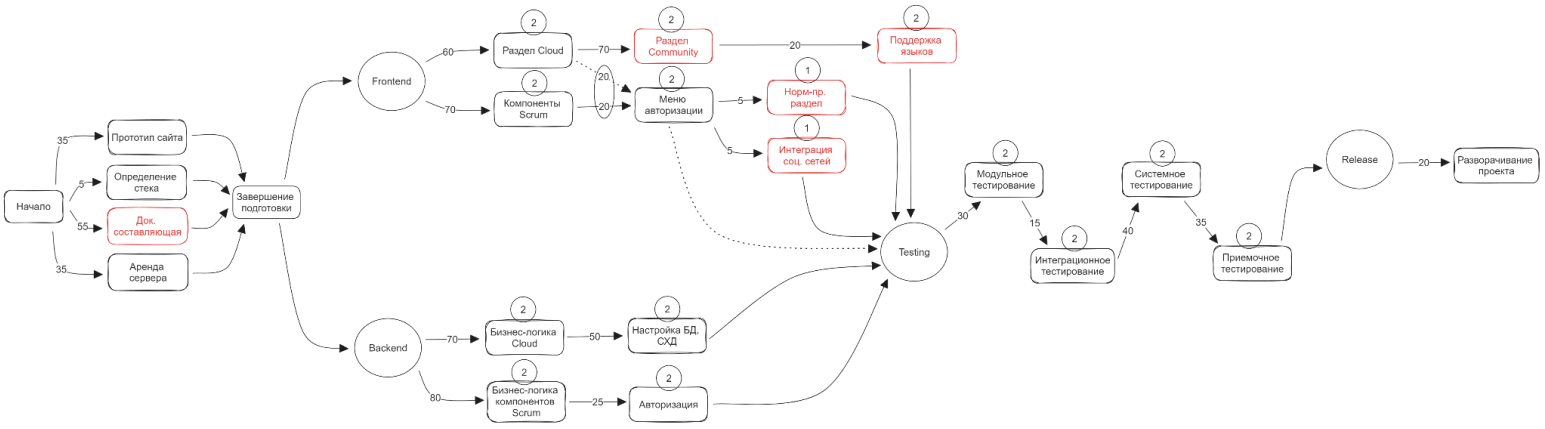
№	Название	Описание	Оценка, мин. /чел. час	Оценка, вероятн. /чел. час	Оценка, макс. /чел. час
1.	Модульное(unit) тестирование	Привлечение QA специалистов для написания и поддержки unit тестов компонентов продукта	20	30	35
2.	Интеграционное тестирование	Продолжение предыдущего шага и написание тестов для проверки интеграции	10	15	20

		компонентов продукта друг с другом			
3.	Системное тестирование	Полное тестирование сценариев	30	40	50
4.	Приемочное тестирование	Проведение alpha, beta тестов	30	35	40

Release:

№	Название	Описание	Оценка, мин. /чел. час	Оценка, вероятн. /чел. час	Оценка, макс. /чел. час
1.	Разворачивание проекта	Финальный запуск проекта на серверах, настройка инфры	20	30	40

Сетевая диаграмма взаимосвязи работ и критический путь



Финальная оценка

Подсчет: Белогаев-Кузнецов ЭПИ_1 расчеты

Критический путь: 345./ч.

Весь путь: 445ч./ч.

Метод PERT: 776ч./ч.

Выполнение проекта: при ориентире на минимальный критический путь получаем, что для выполнения необходимо 345./ч.

Команда:

- 4x Frontend-разработчик – по двое на каждое из направлений: Cloud и Scrum-компоненты. После окончания задач данные группы из 2ух человек продолжают делать менее затратные вещи.
- 4x Backend-разработчик – по двое на каждое из направлений: Cloud и Scrum-компоненты. Затем также два направления – IAM и работа по созданию и организации СХД, БД.
- 2x Тестировщик – проходят полный процесс проведения тестов для созданных продуктов

Общая оценка	Суммарная СКО	Суммарная трудоемкость	Количество человек
753,33	11,55	776,42	10

Оценка размера проекта методом функциональных точек

При анализе методом функциональных точек надо выполнить следующую последовательность шагов (Рисунок 1):

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).

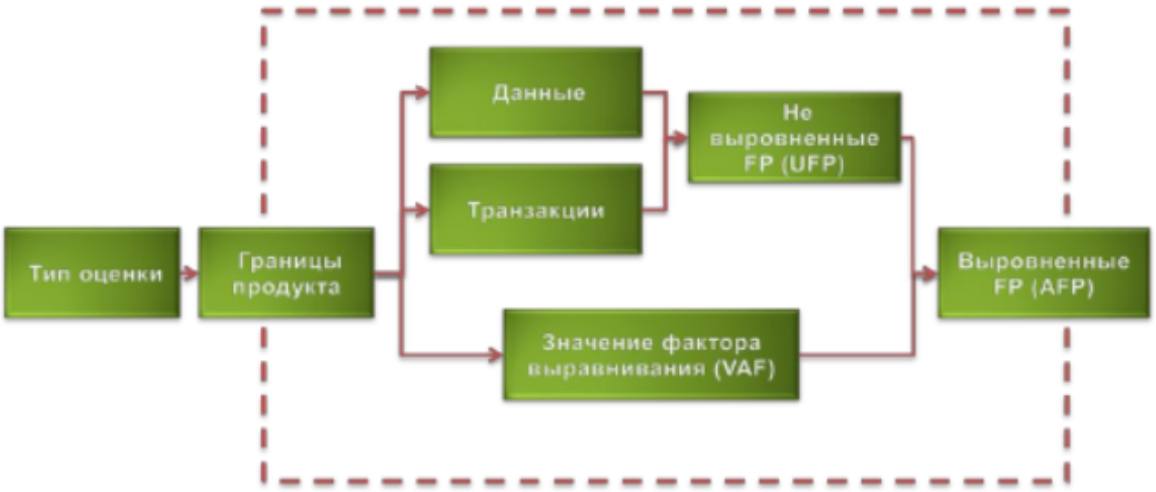


Рисунок 1. Процедура анализа по методу функциональных точек

1 этап: 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.

   В нашем случае по условию необходимо посчитать именно для *проекта разработки*.

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).

Исходя из прошлого пункта, мы будем рассматривать *все разрабатываемые функции*

Третий шаг. Границы продукта (Рисунок 2) 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 2. 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.

По отношению к продукту является внешним все партнерские компании, социальные сети и мессенджеры, все зависимые проекты. Будут входить исключить те функции, которыми должно обладать приложение:

- Интуитивно понятная и простая, но в то же время оснащенная полной доской Канбан

- Полный набор досок для планирования и отображения отдельных спринтов из списка невыполненных работ
- Интегрированное и полное отслеживание проблем/багов
- Полный набор информационных панелей и возможностей создания отчетов
- Обзор хода выполнения мероприятий и завершенных результатов
- Встроенная адаптация
- Различные интеграции (частично через Zapier)
- Функция импорта и экспорта проектов

Подсчет функциональных точек, связанных с данными

Четвертый 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	Low	Low	Average
2-5 RET	Low	Average	High
6+ RET	Average	High	High

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)**
Low	7	5
Average	10	7
High	15	10

№	Название	RET	DET	Сложность	UFP
1	Личный кабинет	Личная информация (1)	Email/username, fullname, projects, closed US, contacts, likes, watched, bio (8)	Low	7
2	Форма регистрации	Данные входа, личная информация (2)	Email/username, пароль, fullname, privacy policy checkbox (4)	Low	7

3	Форма обратной связи	сообщение (1)	сообщение (1)	Low	7
4	Форма для покупки премиум версии	Контакты пользователя, Личная информация пользователя (2)	имя, фамилия, email, password (4)	Low	7

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 9):

- EI (external inputs) — внешние входные транзакции, элементарная операция по обработке данных или управляющей информации, поступающих в систему извне.
- 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 9. 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 10 и Таблица 11.

Таблица 10. Матрица сложности внешних входных транзакций (EI)

EI	**1-4 DET**	**5-15 DET**	**16+ DET**
0-1 FTR	Low	Low	Average
2 FTR	Low	Average	High
3+ FTR	Average	High	High

Таблица 11. Матрица сложности внешних выходных транзакций и внешних запросов (EO & EQ)

EO & EQ	**1-5 DET**	**6-19 DET**	**20+ DET**
0-1 FTR	Low	Low	Average
2-3 FTR	Low	Average	High
4+ FTR	Average	High	High

Оценка транзакций в не выровненных функциональных точках (UFP) может быть получена из матрицы (Таблица 12)

Таблица 12. Сложность транзакций в не выровненных функциональных точках (UFP)

Сложность транзакций	**Количество UFP (EI & EQ)**	**Количество UFP (EO)**
Low	3	4
Average	4	5
High	6	7

№	**Название**	**Тип**	**FTR**	**DET**	**Сложность**	**UFP**
1	Форма обратной связи	EI	1	4	Low	3
2	Форма для покупки премиум аккаунт	EI	1	4	Low	3

3	Форма регистрации	EI	1	4	Low	3
4	Просмотр профиля	EQ	1	8	Low	3
5	Поиск проектов	EQ	1	1	Low	3
6	Просмотр дашбордов	EO	1	3	Low	4
7	Просмотр статических страниц	EI	1	1	Low	3
8	Информация о проекте	EO	3	1	Low	4

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).

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

UFP = 28 + 26 = 54

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

$$VAF = (TDI * 0.01) + 0.65$$

№	**Параметр**	**Вес (DI)**
1	Обмен данными	3
2	Распределенная обработка данных	3
3	Производительность	3
4	Ограничения по аппаратным ресурсам	0
5	Транзакционная нагрузка	2
6	Интенсивность взаимодействия с пользователем	3
7	Эргономика	4
8	Интенсивность изменения данных	1
9	Сложность обработки	4
10	Повторное использование	1
11	Удобство инсталляции	0
12	Удобство администрирования	5
13	Портируемость	1
14	Гибкость	3

$$VAF = (33 * 0.01) + 0.65 = 0.98$$

Расчет количества выровненных функциональных точек (AFP)

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

$$AFP = UFP * VAF.$$

Она учитывает только новую функциональность, 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. Если такой статистики нет, то для оценки трудоемкости и сроков проекта можно использовать метод COCOMO II.

$$AFP = UFP * VAF = 54 * 0.98 = 52.92$$

COCOMO II.

Оценка размера программного продукта в KSLOC (Таблица коэффициентов**)**

Предполагаем, что бек будет написан на джаве, фронт на typescript, время для разработки интерфейса нужно больше, поэтому:
KSLOC = *UFP* * *SIZE* = (54 × 1/3 × 0.053) + (54 × 2/3 × 0.053) = 0.94446 + 1.91754 = **2.862**

Факторы масштаба

В методике используются пять факторов масштаба SF_j, 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)

Таблица 14. 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

TEAM	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	7.80	6.24	4.68	3.12	1.56	0.00

Оценка уровней множителей 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 M,-:

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 15).

Таблица 15. Значения множителей трудоемкости, в зависимости от оценки их уровня

	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	1.00	1.07	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!

Оценка трудоемкости проекта

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 трудоемкости
- SF_j — факторы масштаба
- n=7 — для предварительной оценки
- n=17 — для детальной оценки

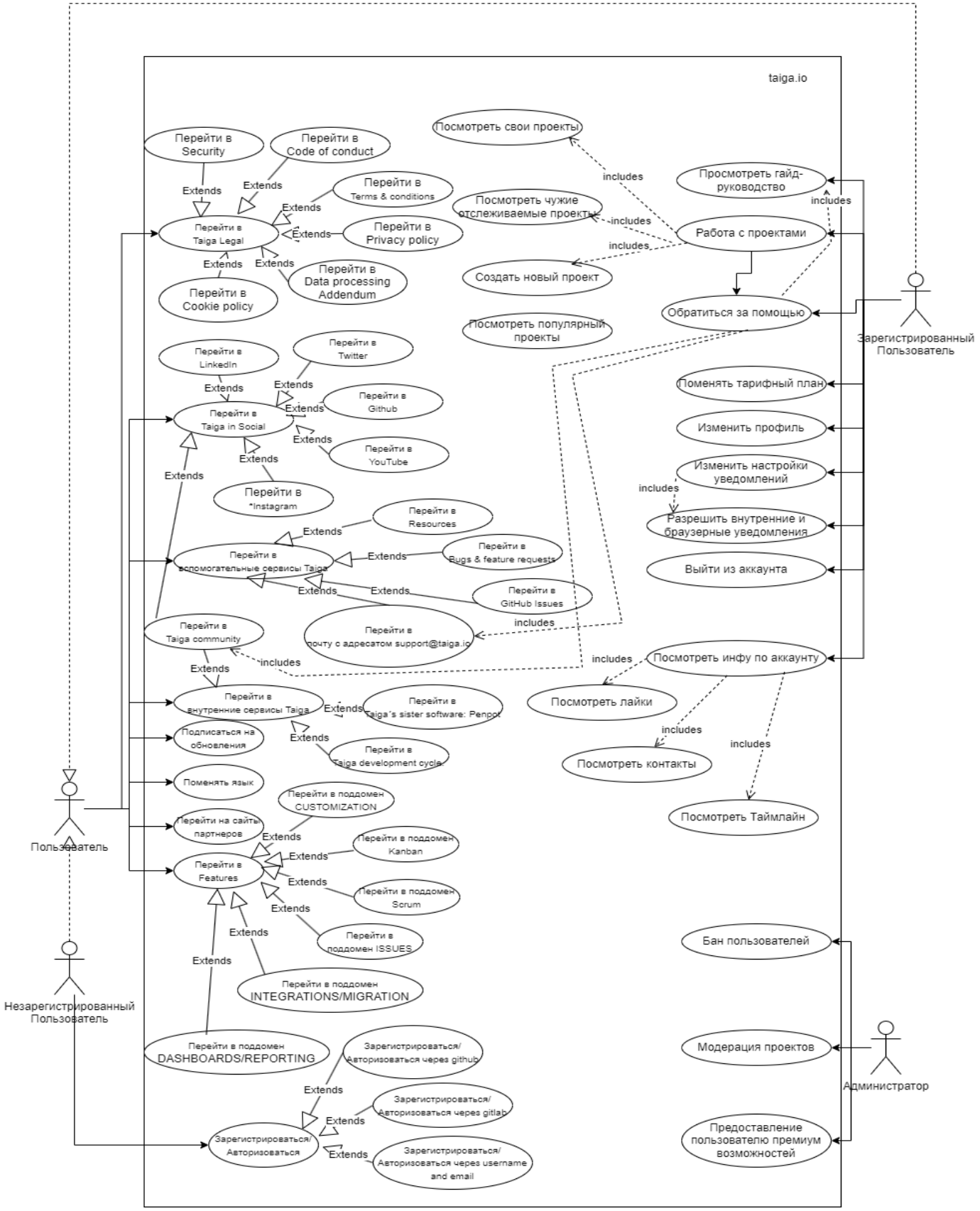
$E = 0.91 + 0.01 \times (2.48 + 2.03 + 5.65 + 0 + 6.24) = 1.074$

$PM = 2.94 \times 2.862^{1.074} \times (1.00 * 0.83 * n/a * 0.87 * 1.00 * 0.87 * 1.00) \approx 2.94 * 3.0936 * 0.628227 \approx$ 5.714 ч./мес.

5.714 ч./мес. = **959.952 ч./ч.**

Use Case Points

Use Case Diagram



Оценка веса прецедентов

Сложность	Вес (UUCW)	Количество	Затраты
Low	5	20	100
Medium	10	5	50
High	15	3	45
Нескорректированный вес варианта использования (UUCW)			195

Оценка веса акторов

Сложность	Вес (AUW)	Количество	Затраты
Low	1	2	2
Medium	2	0	0
High	3	2	6
Масса актера без корректировки (UAW)			8

Оценка веса технических факторов

Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Распределённость	2	2	4
Производительность	1	3	3
Эффективность для пользователя	1	4	4
Сложная внутренняя обработка	1	2	2
Повторное использование кода	1	0	0
Простота установки	0.5	2	1
Простота использования	0.5	4	2
Переносимость	2	0	0
Простота изменений	1	5	5
Многопоточность	1	3	3

Дополнительные возможности безопасности	1	2	2
Доступ к другим системам	1	1	1
Необходимы тренажеры для пользователей	1	0	0
Общий технический фактор (TFactor)			27
TCF = 0.6 + (TF/100)			0.87

Оценка веса факторов окружения

Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Уверенное использование UML/RUP	1.5	4	6.0
Кол-во работников на неполный рабочий день	-1	4	-4
Опытность аналитика	0.5	4	2.0
Опыт работы с приложениями	0.5	2	1.0
Опыт ОО разработки	1.0	2	2.0
Мотивация	1.0	3	3
Сложный язык разработки	-1.0	3	–3.0
Неизменность требований	2	3	6.0
Общий фактор окружающей среды (EFactor)			13.0
ECF = 1.4 + (-0.03 * EF)			1.01

Подсчет UCP

UCP` = (UCW + UAW) * TCF * ECF = (195+8) * 0.87 * 1.01 = 177.62

Подсчёт фактора продуктивности (PF) на основе прошлого проекта

В качестве примера мы выбрали проект по БЛПС, в которую входит выполнение всех лабораторных работ для двух человек.

Список UseCase-ов:

#	Сценарий
1	Регистрация
2	Авторизация
3	Добавление товара в корзину
4	Удаление товара из корзины
5	Просмотр описания и цены товара
6	Изменить личные данные в профиле пользователя (Личный кабинет)
7	Купить то, что находится в корзине
8	Ввести смс
9	Ввести данные карты
10	Ввести номер телефона
11	Ввести адрес доставки

Оценка веса прецедентов

Сложность	Вес (UUCW)	Количество	Затраты
Low	5	15	75
Medium	10	3	30
High	15	0	0
Нескорректированный вес варианта использования (UUCW)			105

Оценка веса акторов

Сложность	Вес (AUW)	Количество	Затраты
Low	1	1	1
Medium	2	0	0
High	3	1	3

Масса актера без корректировки (UAW)	4

Оценка веса технических факторов

Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Распределённость	2.0	0	0.0
Производительность	1.0	1	1.0
Эффективность для пользователя	1.0	2	2.0
Сложная внутренняя обработка	1.0	1	1.0
Повторное использование кода	2.0	1	2.0
Простота установки	0.5	1	0.5
Простота использования	0.5	1	0.5
Переносимость	2.0	1	2.0
Простота изменений	2.0	3	6.0
Многопоточность	1.0	1	1.0
Дополнительные возможности безопасности	1.0	1	1.0
Доступ к другим системам	1.0	1	1.0
Необходимы тренажеры для пользователей	1.0	1	1.0
Общий технический фактор (TFactor)			19
TCF = 0.6 + (TF/100)			0.79

Оценка веса факторов окружения

Фактор	Вес (W)	Номинальная стоимость(F)	Затраты
Уверенное использование UML/RUP	1.5	2	3.0

Кол-во работников на неполный рабочий день	-1	2	-2
Опытность аналитика	0.5	2	1.0
Опыт работы с приложениями	0.5	2	1.0
Опыт ОО разработки	1.0	2	2.0
Мотивация	1.0	4	4
Сложный язык разработки	-1.0	3	–3.0
Неизменность требований	2	4	8.0
Общий фактор окружающей среды (EFactor)			14.0
ECF = 1.4 + (-0.03 * EF)			0.98

Подсчет UCP

UCP` = (UCW + UAW) * TCF * ECF = (105+4)*0.79*0.98 = 84.3878

Подсчет трудоемкости проекта:

Предыдущая работа была выполнена за 36 часов с расчетом на 2-ух человек в команде, итого:

PF = E/UCP = 36*2 / 84.3878 = 0.8532

E = PF * UCP = 0.8532*959.952 ≈ **819 ч./ч.**

Итоговая трудоемкость различными методами:

- Наивный метод: 445 человеко-часов
- PERT: 776 человеко-часов
- COCOMO-II: 959 человеко-часов
- UCP: 819 человеко-часов

Наивный метод позволил быстро оценить трудоемкость проекта, не проводя дополнительных вычислений. Однако результат получился явно меньше остальных, что дает ясно понять, что полагаться на него для объективной оценки довольно трудно.

Метод PERT потребовал небольших усилий для проведения вычислений и позволил уточнить полученные результаты. Однако он базируется на результатах наивного метода.

Методы функциональных точек и COCOMO-II дали результат, значительно отличающийся от других в большую сторону. Это произошло вследствие того, что мы ввели значительно большее количество оценочных параметров и рисков, которые в свою очередь повлияли на скорость разработки

Вывод

Во время выполнения лабораторной работы мы:

- изучили различные методы оценки временных и ресурсных затрат на разработку проекта
- изучили метод наивной оценки
- метод PERT
- метод функциональных точек
- метод COCOMO II и User Case Points.