УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Экономика программной инженерии»

Лабораторная работа 1

по дисциплине
«Экономика программной инженерии»

Студенты

Лаптев Анатолий

Сударушкин Ярослав

P34101

Преподаватель:

Машина Е. А.

Санкт-Петербург 2023

ОГЛАВЛЕНИЕ

Задание

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.

Сайт по варианту: https://reef.live

Выполнение

Функциональные требования

- Система должна представлять возможность просмотра приветственного раздела, рассказывающего о возможностях компании
- Система должна представлять возможность просмотра раздела «Новости», собирающего информацию с других популярных сайтов
- Система должна представлять возможность доступа ознакомления с разрешениями компании
- Система должна представлять возможность поиска подходящих апартаментов из доступного списка

- Система должна представлять возможность отправлять обратную связь, вопросы, а также заявки на апартаменты
- Система должна представлять возможность просмотра общей информации о компании, такие как контакты и расположение

Наивный метод

#	Название	Описание	Optimistic (h-h)	Pessimistic (h-h)	Optimal (h-h)
1	Подготовка		66	172	110
1.1	Прототип сайта	Идет разработка сайт под заказ Продажи de luxe апартаментов в Сочи. Официальный бренд вместе с дизайном, лого и т.д. уже имеется. Требуется прописать функциональные пользовательские сценарии, прототип анимация и интерфейса и т.д.	50	100	70
1.2	Выбор технологий (фундамент)	Важно определиться с используемыми технологиями: Frontend - AngularJS подойдет нам лучше всего, потому что информационный сайт с красивыми анимациями можно очень быстро на нем написать. Backend - простой PHP сервер для простой обработки запросов на обратную связь и запросов на поиск апартаментов	8	32	16
1.3	Аренда хостинга	В интернете довольно много вариантов для столь простого сайта, поэтому нужно найти лучший по отношению цена/качество.	8	40	24
2	Frontend		80	130	100
2.1	Главная	https://reef.live Красивая лента со слайдами (8 разделов)	10	15	12
2.2	Описание	https://reef.live/about/ Тоже лента со слайдами	10	15	12
2.3	Новости	https://reef.live/news/	5	10	7
2.4	Документы	https://reef.live/documents/ страница с ссылками на скачивание документов	5	10	7
2.5	Апарта-менты	https://reef.live/apartments/ https://reef.live/apartments/parameters/ Страница с выбором параметров апартаментов, таблица с ордерингами и фильтрами. Самая трудоемкая часть сайта	30	45	36

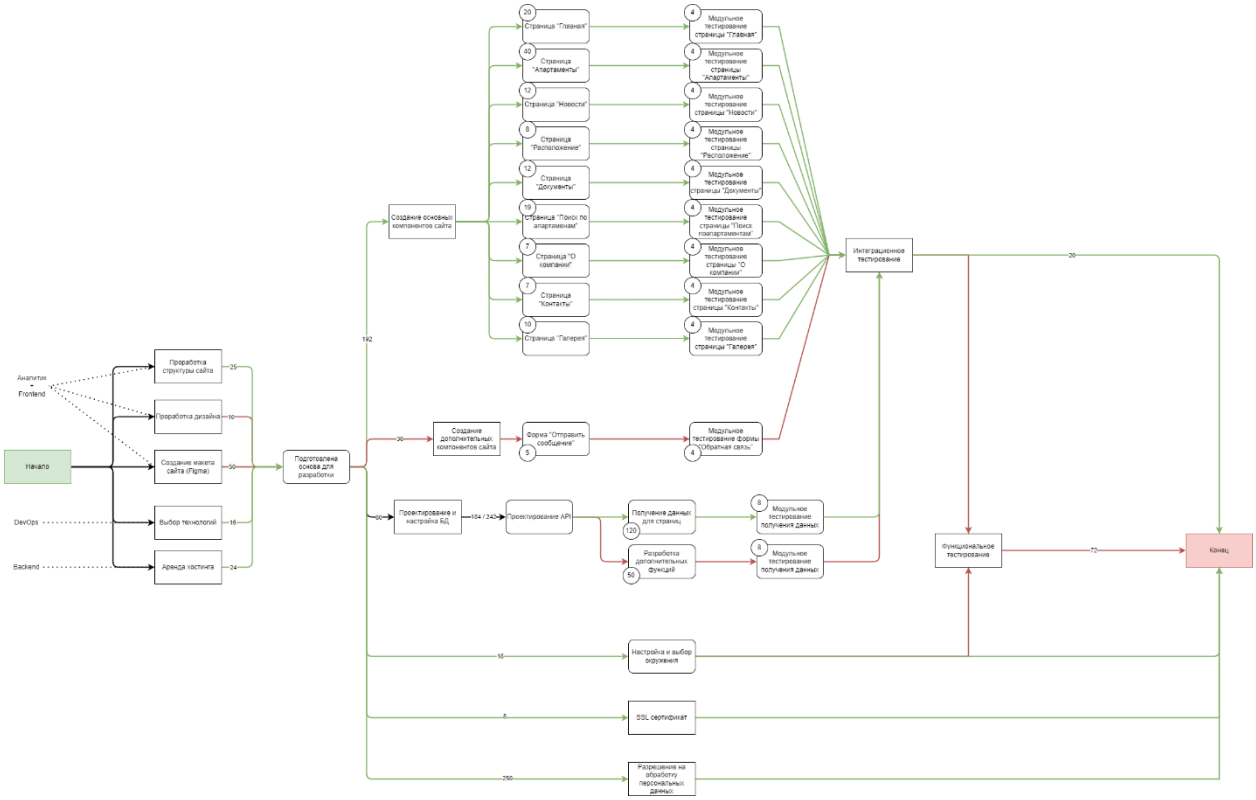
2.6	Галерея	https://reef.live/gallery/	10	15	12
2.7	Расположение	https://reef.live/location/	5	10	7
2.8	Контакты	https://reef.live/contacts/ Самая обычная форма обратной связи	5	10	7
3	Backend		50	75	60
3.1	Апартаменты	Хранение данных об апартаментах	30	45	36
3.2	Контакты	Сервис для реализации обратной связи	10	15	12
3.3	Документы	Хранение документации	10	15	12
4	Testing		72	144	112
4.1	Модульное тестирование	Проверка бэка на надежность, на наличие SQL инъекций и корректную работу запросов к БД. Никакой более сложной логики не предусмотрено, как и других модулей, соотв требуется проверить основной функционал	48	96	72
4.2	Интеграционное тестирование	Тестирование общения фронта и бэка, корректная обработка форм и тд	12	24	20
4.3	Функциональное тестирование	Тестирование полных пользовательских сценариев	12	24	20
5	Релиз		70	144	108
5.1	Тестирование (alpha + beta)	Проверка основного функционала и корректного отображения сайта и его таблиц в различных верстках	48	96	72
5.2	SSL certificate	Self-Signed или отсутствие не подойдет, настройка https соединения	5	9	8
5.3	Выбор окружения	На выбранный хост скинуть скомпилированный html страничку с ангуляром, добавить php сервер	12	24	18
5.4	Настройка	Настройка хоста под наши нужды, проверка общения клиента и сервера	5	15	10
		ИТОГ	338	665	490

Метод PERT

#	Название	Optimistic (h-h)	Pessimistic (h-h)	Optimal (h-h)	Ei =(Pi+Oi+4Mi) / 6	СКОi = (Pi−Oi) / 6
1.1	Прототип сайта	50	100	70	71,67	5,00
1.2	Выбор технологий (фундамент)	8	32	16	17,33	2,67
1.3	Аренда хостинга	8	40	24	24,00	2,67
2.1	Главная	10	15	12	12,17	0,50
2.2	Описание	10	15	12	12,17	0,50
2.3	Новости	5	10	7	7,17	0,50
2.4	Документы	5	10	7	7,17	0,50
2.5	Апартаменты	30	45	36	36,50	1,50
2.6	Галерея	10	15	12	12,17	0,50
2.7	Расположение	5	10	7	7,17	0,50
2.8	Контакты	5	10	7	7,17	0,50
3.1	Апартаменты	30	45	36	36,50	1,50
3.2	Контакты	10	15	12	12,17	0,50
3.3	Документы	10	15	12	12,17	0,50
4.1	Модульное тестирование	48	96	72	72,00	4,00
4.2	Интеграционное тестирование	12	24	20	19,33	0,67
4.3	Функциональное тестирование	12	24	20	19,33	0,67
5.1	Тестирование (alpha + beta)	48	96	72	72,00	4,00
5.2	SSL certificate	5	9	8	7,67	0,17
5.3	Выбор окружения	12	24	18	18,00	1,00
5.4	Настройка	5	15	10	10,00	0,83

E = 493,83
СКО = 8,98
$E_{95\%}$ = 511,79

Метод Критического пути



Критический путь: 408 ч./ч.

Длинный путь: 825 ч./ч.

Выполнение проекта: при ориентире на минимальное время разработки (критический

путь) получаем, что для выполнения нам необходимо 408 ч./ч.

Команда:

- 1x Аналитик
- 2x Fronend-разработчик
- 2x Backend-разработчик
- 2x Тестировщика

Рабочий день считаем: 8 часов (6 разработки + 1 обед + 1 тех. перерыв)

Таким образом, наша команда сможет выполнить проект за:

- Frontend (с модульным тестированием): 108 часов (17 раб. день)
- Backend (с модульным тестированием): 156 часов (раб. день)
- Интеграционное тестирование: 20 часов (4 раб. день)
- Сбор данных: 250 часов (31 раб. день)

Релиз: 6 часов (1 раб. день)

Рассчитаем время разработки и общее время для завершения проекта:

Frontend и Backend можно делать параллельно, после чего уже делать общее интеграционное тестирование, а также сбор данных для обработки может тоже происходить параллельно. В конце после всех настроек - релиз.

Время разработки: 108 + 20 + 156 = 284 часов

Общее время: 35 + 4 + 1 = 40 рабочих дней

Метод Функциональный точек

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

Продукт. Оценивается объем уже существующего и установленного продукта.

Определение области оценки и границ продукта

Все функции. Рассчитываем все необходимые (реально используемые), а не дополнительные или только основные функции.

Подсчет функциональных точек, связанных с данными

- 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.

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

№	Название	RET	DET	Сложность	UFP
1	Форма обратной связи	Контакты пользователя, сообщение (2)	Имя, email, сообщение (3)	Low	7

Подсчет функциональных точек, связанных с транзакциями

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

Типы транзакций:

- 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.

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:

- 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.

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

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

№	Название	Тип	FTR	DET	Сложность	UFP
1	Форма обратной связи	EI	1	4	Low	3
2	Поиск	EQ	3	1	Low	3
3	Просмотр свежих новостей	EO	0	1	Low	4
4	Просмотр статических страниц	EI	1	1	Low	3
5	Информация об апартаментах	EO	3	1	Low	4

Определение суммарного количества не выровненных функциональных точек (UFP)

UFP = 16 + 17 = 33

Определение значения фактора выравнивания (VAF)

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:

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

14	Гибкость	0

DI (degree of influence)

TDI (total degree of influence)

$TDI = \sum DI = 8$

$VAF = (TDI * 0.01) + 0.65 = 0.73$

Расчет количества выровненных функциональных точек (AFP)

$AFP = UPF \times VAF = 33 * 0.73 = 25.41$

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.

Метод COCOMO II

Оценка размера программного продукта в KSLOC

Стек технологий:

- *Angular (TS / JS)*
- *Backend (PHP)*

Разделим функциональность между слоями: 3/4 fronend и 1/4 backend.

Подсчитаем размер по KSLOC:

$KSLOC = UFP * SIZE = (78 \times 3/4 \times 0.060) + (78 \times 1/4 \times 0.053) = 3.51 + 1.0335 = 4.5435$

Оценка уровней факторов масштаба

- **PREC - прецедентность, наличие опыта аналогичных разработок**
- **FLEX** - гибкость процесса разработки
- **RESL** - архитектура и разрешение рисков
- **TEAM - сработанность команды**
- **PMAT - зрелость процессов**

Название фактора	Уровень фактора	Значение уровня
PREC	High	2.48
FLEX	High	2.03
RESL	Low	5.65
TEAM	High	2.19
PMAT	Very Low	7.80

Оценка уровней множителей трудоемкости

Для предварительной оценки проекта необходимо оценить уровень семи множителей трудоемкости M:

- PERS - квалификация персонала
- RCPX - сложность и надежность продукта
- RUSE - разработка для повторного использования
- PDIF - сложность платформы разработки
- PREX - опыт персонала
- FCIL - оборудование
- CSED - требуемое выполнение графика работ

Название	Уровень	Значение
PERS	Nominal	1.00
RCPX	Very Low	0.60
RUSE	Low	0.95
PDIF	Low	0.87
PREX	High	0.87
FCIL	Nominal	1.00
CSED	Nominal	1.00

Оценка трудоемкости проекта

$$PM = A \times SIZE^{E} \times \prod_{i=1}^{n} EM_i \qquad A = 2.94$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j \qquad B = 0.91$$

- SIZE — размер продукта в KSLOC
- EM_i — множители трудоемкости
- SF_j — факторы масштаба
- n=7 — для предварительной оценки
- n=17 — для детальной оценки

$$E = 0.91 + 0.01 \times (2.48 + 2.03 + 5.65 + 2.19 + 7.80) = 1.1115$$

$$PM = 2.94 \times 4.5435^{1.1115} \times (1.00 * 0.60 * 0.95 * 0.87 * 0.87 * 1.00 * 1.00) \approx 6.82261 \text{ ч./мес.}$$

Метод Use Case Points

Прецеденты

Слож-ность	Вес (UUCW)	Количе-ство	За-траты
Low	5	6	30
Medium	10	1	10
High	15	1	15
Нескорректированный вес варианта использования (UUCW)			**55**

Акторы

Сложность	Вес (UUCW)	Количество	Затраты
Low	1	2	2
Medium	2	0	0
High	3	1	3
Масса актера без корректировки (UAW)			**5**

Тех Факторы

Фактор	Вес (W)	Номинальная стоимость (F)	Затраты
Распределенность	1	0	0

Производительность	2	3	6
Эффективность для пользователя	3	4	12
Сложная внутрення обработка	1	0	0
Повторение использованного кода	2	1	2
Простота установки	1	1	1
Простота использования	3	3	9
Переносимость	1	0	0
Простота изменений	3	5	15
Многопоточность	1	0	0
Дополнительные возможности безопасности	1	1	1
Доступ к другим системам	1	2	2
Необходимые трненажеры для пользователей	1	0	0
Общий технический фактор (TF)			**48**
TCF = (TF/100 + 0.6)			**1,08**

Факторы окружения

Фактор	Вес (W)	Номинальная стоимость (F)	Затраты
Знаком с моделью проекта, которая используется	1,5	4	6
Опыт применения	0,5	3	1,5
Опыт в веб разработке	1	4	4
Возможность ведущего аналитика	0,5	2	1
Мотивация	1	2	2

Стабильные требования	1,5	2	3
Частичная занятость	-1	3	-3
Сложность языка программирования	-1	4	-4
Общий фактор окружающей среды (EFactor)			**10,5**
ECF = 1.4 + (-0.03 * EF)			**1,085**

Подсчет UCP

UCP'	**70,308**
E = PF * UCP = 85 ч/ч + юр работа + работа с партнерами + заполнение всех данных = 85 + 250 + 160 + 40 = **535 ч/ч**	

Вывод

Как мы видим, COCOMO II и наивный метод показывают практически идентичные результаты. Неудивительно, в COCOMO II мы ориентируемся на различные реальные параметры, которые мы анализируем из личного опыта, что схоже с наивным методом. Они схожие, т.к. ориентируются на наши предположения и реальный опыт. В случае с использованием метода UCP время расчета получилось меньше ожидаемого. Скорее всего такая ситуация могла получиться из-за не совсем верного расчета для уже готового проекта. Вероятно он был оценен в большем количестве UCP, чем на самом деле содержит,

либо мы ошиблись с реальным временем его реализации, указав слишком малое время. Кроме того, стоит учитывать, что лабораторные работы сильно отличаются от реальной разработки, так что сильные различия в проектах тоже могут повлиять.