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:

*Фам Мань Туан*

Преподаватель:

*Машина Екатерина Алексеевна*

Санкт-Петербург, 2023 г.

Оглавление

[Задание 2](#_Toc154051198)

[Функциональные требования 2](#_Toc154051199)

[Оценка трудоемкости работ 7](#_Toc154051200)

[Сетевая диаграмма взаимосвязи работ и критический путь 9](#_Toc154051201)

[Оценка размера проекта методом функциональных точек 10](#_Toc154051202)

[Расчет трудоемкости методом COCOMO II 11](#_Toc154051203)

[Оценка с помощью User Case Points 12](#_Toc154051204)

[Фактор технической сложности (TCF) 14](#_Toc154051205)

[Сравнение использованных методов 15](#_Toc154051206)

[Вывод 15](#_Toc154051207)

# Задание

Для выданного веб-проекта:

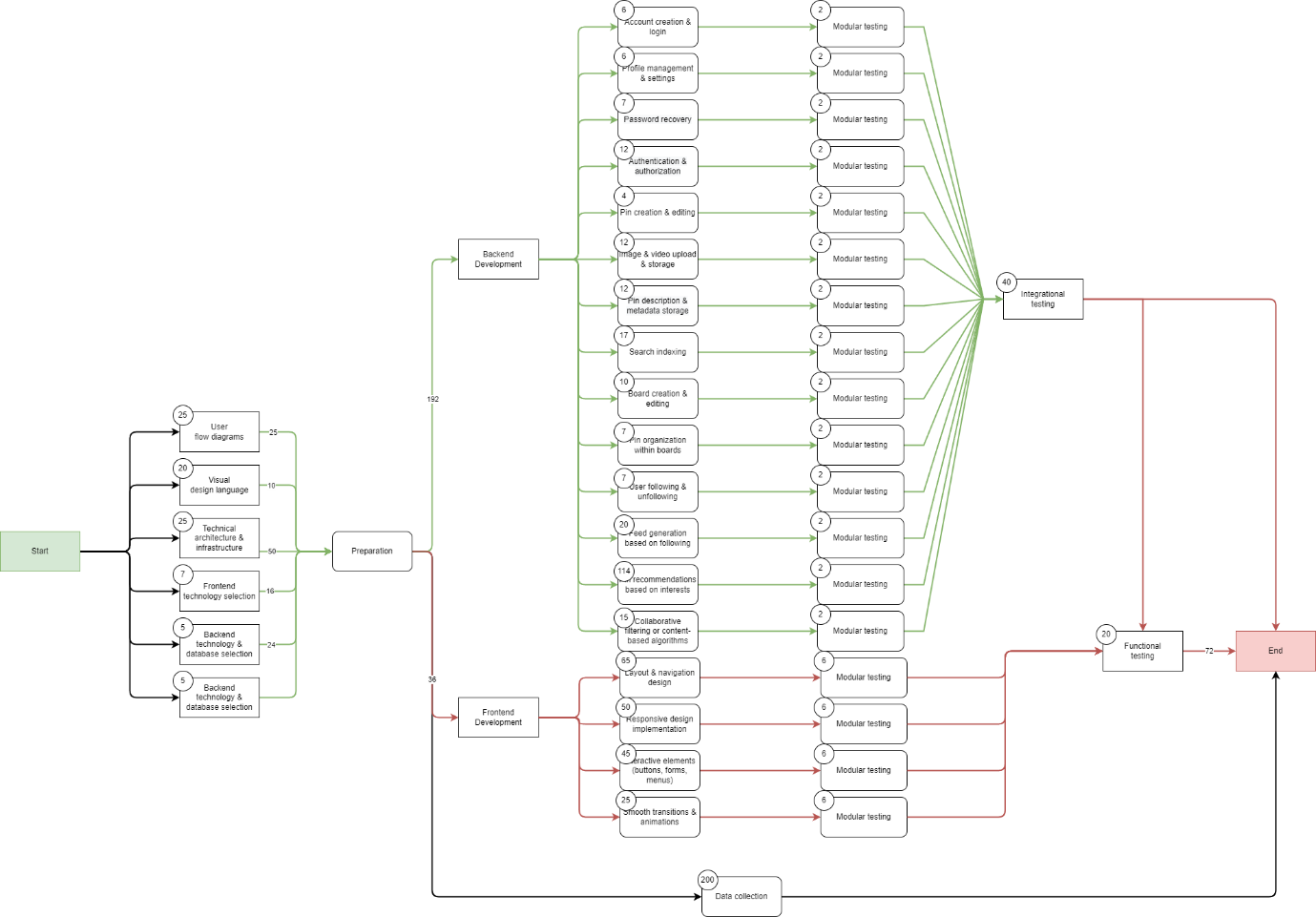
1. Сформировать набор функциональных требований для разработки проекта.
2. Оценить трудоемкость разработки проекта наивным методом.
3. Оценить трудоемкость разработки проекта методом PERT (Project Evaluation and Review Technique**).**Нарисовать сетевую диаграмму взаимосвязи работ и методом критического 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](https://www.qsm.com/resources/function-point-languages-table))
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.

Вариант задания: [pinterest.ru](https://gym.apatity-edu.ru/)

## Набор функциональных требований + наивный метод + метод PERT

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Function Category | Function | Optimistic (player-hours) | Pessimistic (player-hours) | Optimal (player-hours) | Ei | CKOi |
| Preparation | User flow diagrams | 15 | 35 | 25 | 25,000 | 3,333 |
|  | Visual design language | 15 | 25 | 20 | 20,000 | 1,667 |
|  | Technical architecture & infrastructure | 20 | 30 | 25 | 25,000 | 1,667 |
|  | Frontend technology selection | 5 | 10 | 7 | 7,167 | 0,833 |
|  | Backend technology & database selection | 3 | 7 | 5 | 5,000 | 0,667 |
|  | Cloud infrastructure & services selection | 3 | 7 | 5 | 5,000 | 0,667 |
| Backend Development | Account creation & login | 4 | 8 | 6 | 6,000 | 0,667 |
|  | Profile management & settings | 4 | 8 | 6 | 6,000 | 0,667 |
|  | Password recovery | 5 | 10 | 7 | 7,167 | 0,833 |
|  | Authentication & authorization | 10 | 15 | 12 | 12,167 | 0,833 |
|  | Pin creation & editing | 3 | 7 | 4 | 4,333 | 0,667 |
|  | Image & video upload & storage | 10 | 15 | 12 | 12,167 | 0,833 |
|  | Pin description & metadata storage | 10 | 15 | 12 | 12,167 | 0,833 |
|  | Search indexing | 15 | 20 | 17 | 17,167 | 0,833 |
|  | Board creation & editing | 7 | 15 | 10 | 10,333 | 1,333 |
|  | Pin organization within boards | 5 | 10 | 7 | 7,167 | 0,833 |
|  | User following & unfollowing | 5 | 10 | 7 | 7,167 | 0,833 |
|  | Feed generation based on following | 15 | 25 | 20 | 20,000 | 1,667 |
|  | Pin recommendations based on interests | 10 | 20 | 14 | 14,333 | 1,667 |
|  | Collaborative filtering or content-based algorithms | 10 | 20 | 15 | 15,000 | 1,667 |
| Frontend Development | Layout & navigation design | 50 | 80 | 65 | 65,000 | 5,000 |
|  | Responsive design implementation | 40 | 70 | 50 | 51,667 | 5,000 |
|  | Interactive elements (buttons, forms, menus) | 40 | 50 | 45 | 45,000 | 1,667 |
|  | Smooth transitions & animations | 20 | 30 | 25 | 25,000 | 1,667 |
| Testing | Modular testing | 45 | 60 | 52 | 52,167 | 2,500 |
|  | Integrational testing | 30 | 60 | 40 | 41,667 | 5,000 |
|  | Functional testing | 40 | 50 | 45 | 45,000 | 1,667 |
| Maintenance | Bug fixes | Varies based on issues | Varies based on issues | Varies based on issues |  |  |
|  | Security updates | Varies based on needs | Varies based on needs | Varies based on needs |  |  |
|  | Feature updates | Varies based on features | Varies based on features | Varies based on features |  |  |
|  |  | 439 | 712 | 558 | 563,833 | 11,142 |
|  |  |  |  |  | 586,117 |  |

## Сетевая диаграмма взаимосвязи работ и критический путь



Critical path: 558 hh.

Long way: 778 hh.

Project implementation: with a focus on minimal development time (critical

path) we find that we need 558 hh to complete.

Team:

* 1x Analyst
* 2x Frontend developer
* 2x Backend developer
* 2x Testers

We count a working day as: 8 hours (6 working hours + 1 lunch time + 1 technical break)

Thus, our team will be able to complete the project in:

* Frontend (with unit testing): 177 hours (30 working days)
* Backend (with unit testing): 209 hours (35 working days)
* Testing: 85 hours (12 working days)
* Data collection: 200 hours (25 working days)
* Release: 6 hours (1 working day)

Let's calculate the development time and the total time to complete the project:

Frontend and Backend can be done in parallel, after which you can do the same

integration testing, as well as collecting data for processing can also

happen in parallel. At the end, after all the settings - release.

Development time: 87 + 209 + 200 = 496 hours

Total time: 15 + 35 + 12 = 62 working days

# Оценка размера проекта методом функциональных точек

Тип оценки: продукт

Область оценки: все функции.

Подсчет функциональных точек, связанных с данными

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Name | RET | DET | Complexity | UFP |
| 1 | User | User information  (1) | name,  surname,  gender, date of birth,  city,  email (7) | Low | 7 |
| 2 | Image | Image information  (1) | id, url, title, description, upload date, upload user, width, height, size, format (10) | Low | 10 |
| 3 | Board | Board information  (1) | id, user id, pin number, number of likes, number of saves, view count (6) | Low | 6 |
| 4 | Pin | Pin information  (1) | id, user id, image url, title, description, creation date (6) | Low | 6 |

Sum(UFP) = 29

Подсчет функциональных точек, связанных с транзакциями

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Name | Type | FTR | DET | Complexity | UFP |
| 1 | Image Upload and Storage | EI | 1 | 10 | Low | 3 |
| 2 | Image Display | EO | 1 | 6 | Low | 4 |
| 3 | Pin Creation | EI | 2 | 6 | Average | 4 |
| 4 | Pin Editing | EI | 2 | 6 | Average | 4 |
| 5 | Pin Deletion | EI | 2 | 4 | Low | 3 |
| 6 | Pin Display | EO | 2 | 10 | Average | 5 |
| 7 | Pin Search | EQ | 3 | 14 | Average | 4 |
| 8 | Board Creation | EI | 1 | 6 | Low | 3 |
| 9 | Board Editing | EI | 2 | 12 | Average | 4 |
| 10 | Board Deletion | EI | 1 | 6 | Low | 3 |
| 11 | Board Display | EO | 3 | 16 | Average | 5 |
| 12 | User Registration | EI | 1 | 7 | Low | 3 |
| 13 | User Login | EI | 1 | 4 | Low | 3 |
| 14 | Account Settings | EI | 1 | 7 | Low | 3 |
| 15 | Following Users | EI | 2 | 4 | Low | 3 |
| 16 | Feed Generation | EO | 3 | 14 | Average | 5 |

Sum UFP = 29+59 = 88

|  |  |  |
| --- | --- | --- |
| № | Parameter | Weight |
| 1 | Data exchange | 2 |
| 2 | Distributed data processing | 0 |
| 3 | Performance | 0 |
| 4 | Hardware resource limitation | 0 |
| 5 | Transaction load | 0 |
| 6 | Intensity of user interaction | 2 |
| 7 | Ergonomics | 3 |
| 8 | Data exchange rate | 1 |
| 9 | Processing complexity | 0 |
| 10 | Reuse | 1 |
| 11 | Ease of installation | 0 |
| 12 | Ease of administration | 2 |
| 13 | Portability | 1 |
| 14 | Flexibility | 0 |

Sum DI = 24

VAF = TDI \* 0,01 + 0,65 = 0,24 + 0,65 = 0,89

AFP = VAF \* UFP = 0,89 \* 88 = 78,32

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.

# Расчет трудоемкости методом COCOMO II

Фронтенд проекта реализован на JavaScript + JQuery, бэкенд – на C . С учётом того, что создание интерфейса сайта является важной частью процесса разработки, примем доли фронтенда и бэкенда от всего проекта как FQ = 0,5, BQ = 0.5 соответственно. Объём проекта в KSLOC на основании метода функциональных точек (оценка количества строк, необходимых на реализацию одной не выровненной функциональной точки для JavaScript – 0,056 т. строк, для PHP – 0,060 т. строк): KSLOC = (FQ \* UFP \* 0,056)+ (BQ \* UFP \* 0,060) = (0,5 \* 88 \* 0,056)+ (0,5 \* 88 \* 0,060) = 4.104 т.строк

|  |  |  |  |
| --- | --- | --- | --- |
| No | Factor | Level | SFj |
| 1 | PREC | High | 2.48 |
| 2 | FLEX | Nominal | 03.04 |
| 3 | RESL | Nominal | 4.24 |
| 4 | TEAM | Very High | 1.10 |
| 5 | PMAT | Low | 6.24 |

Sum (SF) = 17.1

|  |  |  |  |
| --- | --- | --- | --- |
| No | Factor | Level | Mi |
| 1 | PERS | High | 0.83 |
| 2 | RCPX | Very Low | 0.60 |
| 3 | RUSE | Nominal | 1.00 |
| 4 | PDIF | Low | 0.87 |
| 5 | PREX | High | 0.87 |
| 6 | FCIL | Nominal | 1.00 |
| 7 | SCED | Nominal | 1.00 |

A = 2,94; B = 0,91

=0.91+0.0.1\*17.10=1.081

=2.94\*4.104^1.081\*(0.83\*0.6\*1\*0.87\*0.87\*1\*1)= 5.099 ч. /мес = 815.853 ч/ч

# Оценка с помощью User Case Points

**Неадаптированный вес участника (UAW)**

|  |  |  |  |
| --- | --- | --- | --- |
| Classification | Weight (UAW) | Number of members | Product |
| Простой | 1 | 0 | 0 |
| Среднее | 2 | 1 | 2 |
| Сложный | 3 | 1 | 3 |
|  |  |  |  |

UAW = 5

**Нескорректированный вес варианта использования (UUCW)**

|  |  |  |  |
| --- | --- | --- | --- |
| Classification | Weight (UCW) | Number of transactions | Product |
| Простые | 5 | 3 | 15 |
| Средние | 10 | 3 | 30 |
| Сложные | 15 | 10 | 150 |
|  |  |  |  |

UCW = 195

**Фактор технической сложности (TCF)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Factor | Description | Weight | Complexity | Product |
| T1 | Distributed system | 2,0 | 3 | 6 |
| T2 | Performance | 1,0 | 5 | 5 |
| T3 | Efficiency for the end user | 1,0 | 5 | 5 |
| T4 | Complexity of internal processing | 1,0 | 2 | 2 |
| T5 | Code reuse | 1,0 | 0 | 0 |
| T6 | Feasibility to install | 0,5 | 2 | 1 |
| T7 | Feasibility of use | 0,5 | 5 | 2,5 |
| T8 | Portability | 2,0 | 3 | 6 |
| T9 | Feasibility to change | 1,0 | 3 | 3 |
| T10 | Multithreading | 1,0 | 3 | 3 |
| T11 | Security features | 1,0 | 5 | 5 |
| T12 | Access for third parties | 1,0 | 3 | 3 |
| T13 | End user training | 1,0 | 1 | 1 |
|  |

C1 = 0,6; C2 = 0,01

TCF = C1+C2\*Sum(W\*F) = 1.025

**Фактор сложности окружающей среды (ECF)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Фактор | Описание | Weight | Effect | Product |
| E1 | Introduction to the development process | 1,5 | 1 | 1,5 |
| E2 | Object-oriented team experience | 0,5 | 1 | 0,5 |
| E3 | Lead analyst experience | 1,0 | 1 | 1 |
| E4 | Team motivation | 0,5 | 1 | 0,5 |
| E5 | Stability of requirements | 1,0 | 0 | 0 |
| E6 | Part-time workers | 2,0 | 0 | 0 |
| E7 | Complexity of programming languages | -1,0 | 5 | -5 |
| E8 | Application experience | -1,0 | 3 | -3 |

C1 = 1,4; C2 = -0,03

ECF = C1 + C2\*Sum(Wi \* Fi) = 1,535

UUCP = (UCW + UAW) \* TCF \* ECF = (5+195)\* 1,025\*1,535= 314,675

Для оценки будет использована завершенная курсовая работа «ИСБД»

**Список прецедентов проекта**

|  |  |
| --- | --- |
| № | Описание |
| 1 | User authentication |
| 2 | Display user information |
| 3 | Display game information |
| 4 | Create a game |
| 5 | Create a developer |
| 6 | Create a publisher |
| 7 | Create a platform |
| 8 | Create a genre |
| 9 | Rate a game |
| 10 | Post a review |
| 11 | Calculate average rating for a game |
| 12 | Display the game chart |
| 13 | Display reviews by game |

**Неадаптированный вес участника (UAW)**

|  |  |  |  |
| --- | --- | --- | --- |
| Classification | Weight (UAW) | Number of members | Product |
| Простой | 1 | 1 | 1 |
| Среднее | 2 | 1 | 2 |
| Сложный | 3 | 0 | 0 |

UAW = 3

**Нескорректированный вес варианта использования (UCW)**

|  |  |  |  |
| --- | --- | --- | --- |
| Classification | Weight (UCW) | Number of transactions | Product |
| Простые | 5 | 9 | 45 |
| Средние | 10 | 2 | 20 |
| Сложные | 15 | 2 | 30 |

UCW = 95

### Фактор технической сложности (TCF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Factor | Description | Weight | Complexity | Product |
| T1 | Distributed system | 2,0 | 2 | 4 |
| T2 | Performance | 1,0 | 3 | 3 |
| T3 | Efficiency for the end user | 1,0 | 3 | 3 |
| T4 | Complexity of internal processing | 1,0 | 0 | 0 |
| T5 | Code reuse | 1,0 | 0 | 0 |
| T6 | Feasibility to install | 0,5 | 0 | 0 |
| T7 | Feasibility of use | 0,5 | 3 | 1,5 |
| T8 | Portability | 2,0 | 2 | 4 |
| T9 | Feasibility to change | 1,0 | 2 | 2 |
| T10 | Multithreading | 1,0 | 1 | 1 |
| T11 | Security features | 1,0 | 1 | 1 |
| T12 | Access for third parties | 1,0 | 3 | 3 |
| T13 | End user training | 1,0 | 0 | 0 |

C1 = 0,6;C2 = 0,01

TCF = C1+C2\*Sum(W\*F) = 0.825

**Фактор сложности окружающей среды (ECF)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Factor | Description | Weight | Complexity | Product |
| E1 | Introduction to the development process | 1,5 | 3 | 4,5 |
| E2 | Object-oriented team experience | 0,5 | 3 | 1,5 |
| E3 | Lead analyst experience | 1,0 | 1 | 1 |
| E4 | Team motivation | 0,5 | 1 | 0,5 |
| E5 | Stability of requirements | 1,0 | 0 | 0 |
| E6 | Part-time workers | 2,0 | 2 | 4 |
| E7 | Complexity of programming languages | -1,0 | 1 | -1 |
| E8 | Application experience | -1,0 | 1 | -1 |

C1 = 1,4; C2 = -0,03

ECF = C1 + C2\*Sum(Wi \* Fi)= 1,145

**Точки варианта использования (UCP)**

UUCP = (UCW + UAW) \* TCF \* ECF = (3+95)\*0.825\*1.145 = 92.573

С учётом фактического времени разработки, получаем следующий фактор продуктивности:

PF = E/UUCP = 170/92.573= 1,836

UCP проекта pinterest

UCP = UUCP \* PF = 314,675\*1,836= 577,743 ч/часов

# Сравнение использованных методов

Итоговая трудоемкость различными методами:

* Naive method: 558 человеко-часов
* PERT: 586,117 человека-часов
* МФТ+ COCOMO II: 815.853 человека-часов
* UCP: 603.257 человека-часов

Использование наивного метода позволяет быстро оценить трудозатраты без дополнительных вычислений. Однако полагаться на него достаточно рискованно, поскольку этот метод не учитывает опыт участников напрямую.

Метод PERT требует небольших трудозатрат на вычисления и позволяет уточнить результаты наивного метода. Несмотря на низкую сложность вычислений, использование PERT кажется оправданным, но он все равно зависит от результатов наивного метода и не учитывает оценку опыта участников.

Метод COCOMO II приводит к результатам, существенно отличающимся в большую сторону. Это объясняется тем, что данный метод слабо приспособлен для небольших проектов.

Метод UCP дает результат, сопоставимый с наивным методом оценки. Его результаты кажутся более близкими к реальности, благодаря более глубокому анализу проекта по сравнению с первыми двумя методами. Полученный результат выглядит более правдоподобным, чем результат по методу COCOMO II.

# Вывод

Во время выполнения лабораторной работы мы изучили различные методы оценки временных и ресурсных затрат на разработку проекта, а именно наивная оценка, PERT, метод функциональных точек, метод COCOMO II и User Case Points.