Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работе № 7

по курсу «Экономика программной инженерии»

на тему: «Оценка параметров программного проекта с использованием метода функциональных точек и модели COCOMO II»

Вариант № 1

Студент ИУ7-83Б _____ Дегтярев А. И.
(Группа) (Подпись, дата) (И. О. Фамилия)

Студент ИУ7-83Б _____ Миронов Г. А.
(Группа) (Подпись, дата) (И. О. Фамилия)

Студент ИУ7-82Б _____ Сучкова Т. М.
(Группа) (Подпись, дата) (И. О. Фамилия)

Преподаватель _____ Барышникова М. Ю.
(Подпись, дата) (И. О. Фамилия)

2023 г.

1 Задание

1.1 Цель работы

Продолжение знакомства с существующими методиками предварительной оценки параметров программного проекта и практическая оценка затрат по модели COCOMO II.

1.2 Задание

1. Ознакомиться с прилагаемым к лабораторной работе теоретическим материалом (презентациями к лекции №8).

2. На основе своего варианта задания рассчитать количество функциональных точек для разрабатываемого программного приложения. С этой целью разработать программный инструмент.

3. Произвести оценку трудозатрат и длительности разработки по методике COCOMO II с использованием моделей композиции приложения и ранней разработки архитектуры.

4. Определить среднюю численность команды разработчиков.

5. На основе экспертной оценки стоимости человеко-месяца произвести предварительную оценку бюджета проекта.

6. Дать заключение о применимости метода функциональных точек и модели COCOMO II, а также их сравнение с базовой моделью COCOMO для решения поставленной задачи с учетом своего варианта.

2 Методика функциональных точек

Данный метод используется для измерения производительности взамен устаревшего линейного подхода, где производительность измерялась количеством строк программного кода.

Преимуществом данного метода является то, что поскольку применение функциональных точек основано на изучении требований, то оценка необходимых трудозатрат может быть выполнена на самых ранних стадиях работы над проектом.

Определение числа функциональных точек является методом количественной оценки ПО, применяемым для измерения функциональных характеристик процессов его разработки и сопровождения независимо от технологии, использованной для его реализации.

Трудоемкость вычисляется на основе функциональности разрабатываемой системы, которая, в свою очередь, определяется путем выявления функциональных типов —логических групп взаимосвязанных данных, используемых и поддерживаемых приложением, а также элементарных процессов, связанных с вводом и выводом информации.

3 COCOMO II

COCOMO II является развитием стандартного COCOMO. В методику входят три различные модели оценки стоимости.

3.1 Модель композиции приложения

Эта модель, которая подходит для проектов, созданных с помощью современных инструментальных средств. Единицей измерения служит объектная точка.

3.2 Модель ранней разработки архитектуры

Применяется для получения приблизительных оценок проектных затрат периода выполнения проекта перед тем как будет определена архитектура в целом. В этом случае используется небольшой набор новых драйверов затрат и новых уравнений оценки. В качестве единиц измерения используются функциональные точки либо KSLOC.

3.3 Постархитектурная модель

Наиболее детализированная модель COCOMO II, которая используется после разработки архитектуры проекта. В состав этой модели включены новые драйверы затрат, новые правила подсчета строк кода, а также новые уравнения.

4 Проект

Компания получила заказ на разработку автоматизированной информационной системы оплаты штрафов ГИБДД. Оплата штрафов возможна через веб-интерфейс веб-портала и через приложение для мобильного телефона.

4.1 Роли пользователей

В системе предусмотрено два вида пользователей: Пользователь (User) и Администратор (Administrator). Пользователь может просматривать и оплачивать штрафы,

Администратору доступен просмотр всех выплат, просмотр данных пользователей, их редактирование и создание новых пользователей.

Все записи типа «Пользователь» в системе имеют следующие поля: id, логин, пароль, тип, регистрационный номер водительского удостоверения, номер банковской карты.

4.2 Модули

Информационная система состоит из следующих модулей:

— приложение для мобильного телефона;

— веб-портал;

— модуль регистрации и авторизации;

— модуль обмена данными с системой ГИБДД;

— модуль проведения платежных транзакций.

4.2.1 Приложение для мобильного телефона

Содержимое:

— страница регистрации, где пользователь заполняет следующие поля: логин, пароль, номер водительского удостоверения, номер банковской карты;

— страницу просмотра штрафов, на которой отображаются неоплаченные штрафы, и есть кнопка “оплатить” для каждого штрафа. После нажатия кнопки пользователю приходит ответ о положительном или отрицательном результате выполнения операции.

4.2.2 Веб-портал

Обеспечивает те же функциональные возможности для Пользователя и в дополнение к этому он имеет панель Администратора.

4.2.3 Модуль регистрации и авторизации

Позволяет добавлять пользователей в базу данных.

4.2.4 Модуль обмена данными с системой ГИБДД

Предназначен для получения списка штрафов, а также оповещения ГИБДД об оплате штрафа. При отправке сообщения с номером водительского удостоверения информационной системе ГИБДД модуль обмена данными получает список штрафов.

При этом каждый штраф описывается следующими полями: номер постановления, дата постановления, имя, фамилия, отчество нарушителя, сумма штрафа.

На сообщение об оплате система ГИБДД присылает подтверждение об удаления штрафа из списка неоплаченных или отказ.

Примечание: Система не хранит штрафы в своей базе данных, а получает их из системы ГИБДД при каждом запросе.

4.2.5 Модуль проведения платежных транзакций

Модуль по защищенному протоколу отправляет платежной системе запрос с указанием номера карты пользователя, номера счета ГИБДД и суммы на выполнение проведения оплаты.

Система отсылает положительный или отрицательный ответ о результате выполнения. В отличие от штрафов все операции по оплате сохраняются в базу данных.

4.3 Характеристики команды, продукта и проекта

— обмен данными — 5;

— распределенная обработка — 5;

— производительность — 3;

— эксплуатационные ограничения по аппаратным ресурсам — 0;

— транзакционная нагрузка — 3;

— интенсивность взаимодействия с пользователем (оперативный ввод данных) — 2;

— эргономические характеристики, влияющие на эффективность работы конечных пользователей — 0;

— оперативное обновление — 4;

— сложность обработки — 4;

— повторное использование — 3;

— легкость инсталляции — 0;

— легкость эксплуатации/администрирования — 3;

— портируемость — 5;

— гибкость — 0.

При разработке ПО 30 % кода будет написано на SQL, 10 % — на JavaScript, 60 % — на Java.

К разработке проекта планируется привлечь довольно слаженную команду высокопрофессиональных разработчиков, у которых, однако, практически отсутствует опыт в разработке систем подобного типа. При этом заказчик настаивает на довольно строгом процессе с периодической демонстрацией рабочих продуктов, соответствующих этапам жизненного цикла. Учитывая новизну проекта для команды, на этапе его подготовки был осуществлен

относительно детальный анализ рисков, свойственных архитектуре разрабатываемой системы. Организация находится чуть выше второго уровня зрелости процессов разработки.

Надежность и уровень сложности (RCPX) разрабатываемой системы оцениваются как очень высокие, проект не предусматривает специальных усилий на повторное использование компонентов (RUSE). Возможности персонала (PERS) можно охарактеризовать как очень высокие, однако опыт членов команды в данной сфере (PREX) является скорее низким. Сложность платформы (PDIF) средняя. Разработка предусматривает интенсивное использование инструментальных средств поддержки (FCIL). Учитывая новизну проекта для команды и высокие требования по надежности, Заказчик не настаивает на жестком графике (SCED).

5 Лабораторная работа

5.1 Описание проекта

5.2 Показатели проекта

В данной работе рассматриваются модель композиции приложения и модель ранней разработки архитектуры.

Показатели проекта:

— новизна проекта (PREC) — почти полное отсутствие прецедентов, в значительной мере непредсказуемый проект;

— гибкость процесса разработки (FLEX) — точный, строгий процесс разработки;

— разрешение рисков в архитектуре системы (RESL) — в целом (75%)

— сплоченность команды (TEAM) — взаимодействие как в едином целом;

— уровень развития процесса разработки (PMAT) — уровень 2;

5.3 Расчет по методу функциональных точек

FTR — количество связанных с каждым функциональным типом файлов типа ссылок.

DET — количество связанных с каждым функциональным типом элементарных данных.

RET — количество типов элементов записей.

EI (внешний ввод) — элементарный процесс, перемещающий данные из внешней среды в приложение.

EO (внешний вывод) — элементарный процесс, перемещающий данные, вычисленные в приложении, во внешнюю среду.

EQ (внешний запрос) — элементарный процесс, состоящий из комбинации «запрос/ответ», не связанный с вычислением производных данных или обновлением внутренних логических файлов (базы данных).

ILF (внутренний логический файл) — выделяемые пользователем логически связанные группы данных или блоки управляющей информации,

которые поддерживаются внутри продукта и обслуживаются через внешние вводы.

EIF (внешний интерфейсный файл) — выделяемые пользователем логически связанные группы данных или блоки управляющей информации, на которые ссылается продукт, но которые поддерживаются вне продукта.

В нашем приложении используются 4 внутренних файла: таблица с логинами и паролями, таблица с типом заявки, именем бумаги, ценой и количеством, таблица с названием бумаги. Также существует одна внешняя таблица с информацией о бирже с названием бумаги, ценой и изменением.

Вычисление EI

— Регистрация

FTR = 1 (один внутренний логический файл);

DET = 4 (логин, пароль, номер водительского удостоверения и номер банковской карты).

— Оплата штрафа

FTR = 1 (один внутренний логический файл);

DET = 1 (дата оплаты).

— Добавление пользователя администратором;

FTR = 1 (один внутренний логический файл);

DET = 5 (логин, пароль, тип, номер карты, номер удостоверения).

— Редактирование пользователя администратором;

FTR = 1 (один внутренний логический файл);

DET = 5 (логин, пароль, тип, номер карты, номер удостоверения).

Уровень сложности – низкий.

Вычисление EO

— Вывод списка штрафов

FTR = 1 (один внутренний логический файл);

DET = 4 (номер, ФИО, дата, сумма).

— Вывод сообщения о статусе оплаты

FTR = 1 (один внутренний логический файл);

DET = 1 (статус оплаты).

— Вывод информации о пользователе

FTR = 1 (один внутренний логический файл);

DET = 6 (все поля из базы).

Уровень сложности – низкий.

Вычисление EQ

— Запрос на авторизацию

FTR = 1 (один внутренний логический файл);

DET = 3 (логин, пароль, флажок).

— Запрос на вывод списка

FTR = 1 (один внутренний логический файл);

DET = 4 (номер, ФИО, дата, сумма).

Уровень сложности – низкий.

Вычисление ILF

— ILF

RET = 3 (элементы записи);

DET = 6 (элементы данных).

Уровень сложности – низкий.

Вычисление EIF

— Веб

RET = 3 (элементы записи);

DET = 10 (элементы данных).

— Мобильное приложение

RET = 3 (элементы записи);

DET = 5 (элементы данных).

Уровень сложности – низкий.

5.4 Рузультат расчетов

5.4.1 Метод функциональных точек

Нормированное количество функциональных точек: 47.94
Количество функциональных точек: 47
Количество строк исходного кода: 3590

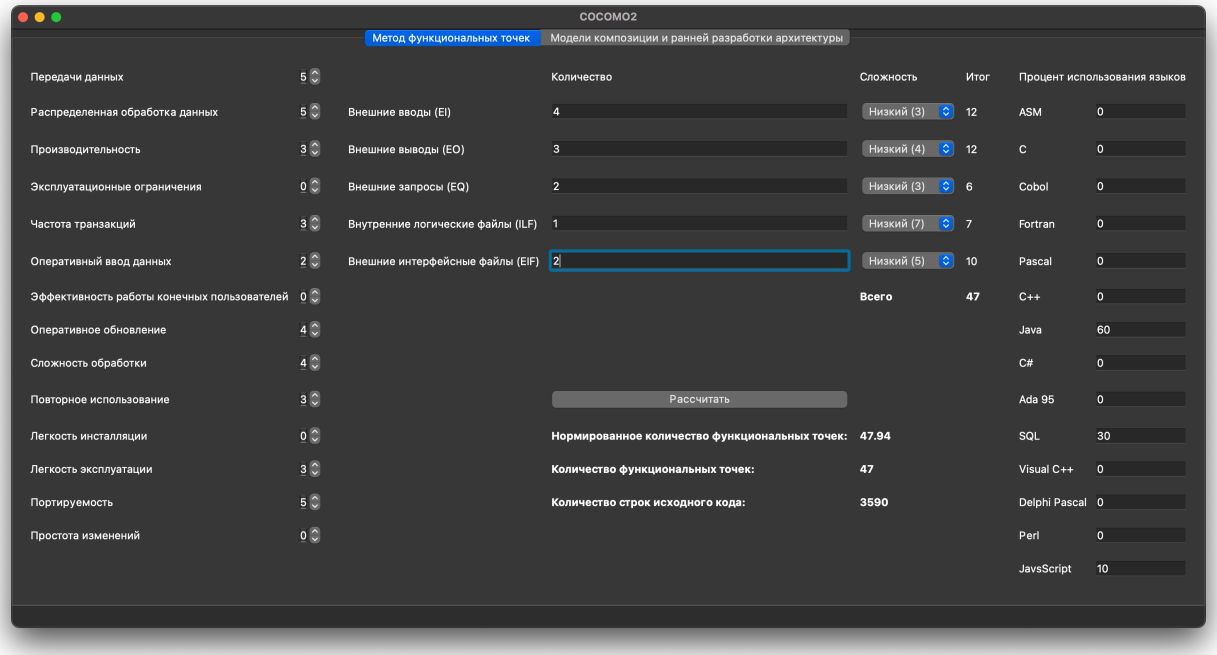

Рисунок 5.1 – Расчет проекта по методу функциональных точек

5.4.2 Оценка по модели COCOMO II

$p = 1.1964$

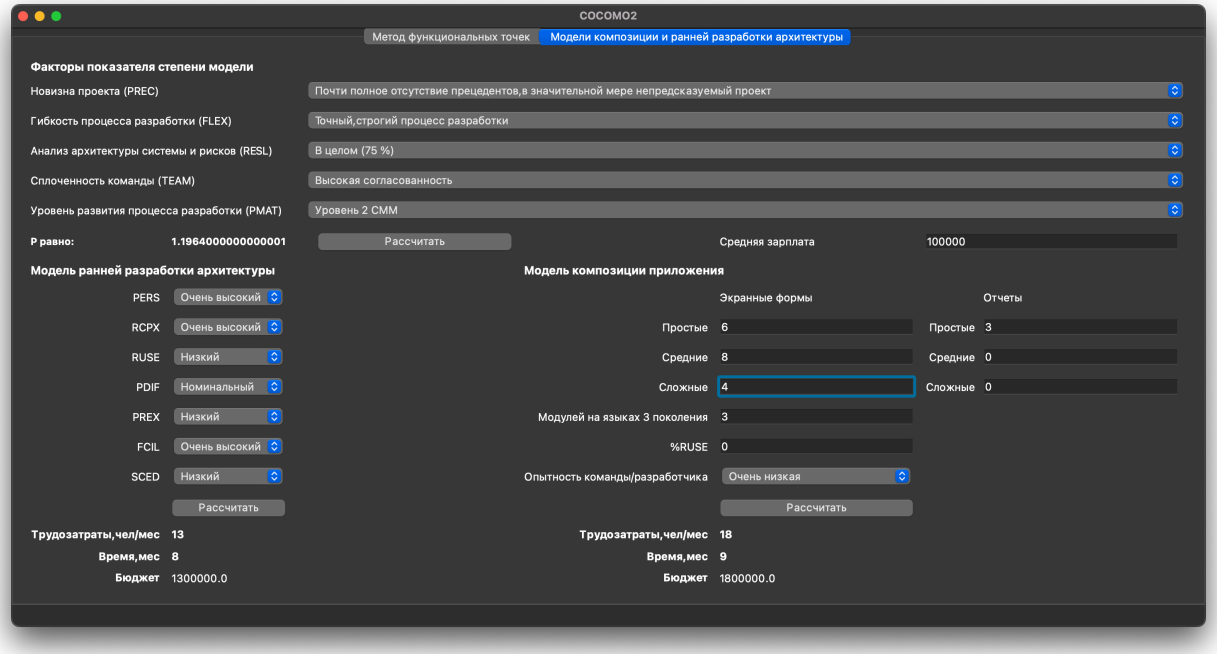

Рисунок 5.2 – Расчет проекта по оставшимся методам

5.4.3 Композиция приложения

— страница регистрации — 4 простых поля и 1 средней сложности (обращение к БД);

— страница авторизации — 2 простых поля и 1 средней сложности (обращение к БД);

— страница штрафов — 4 высокой сложности (обращение к БД ГИБДД);

— страница информации о пользователе — 6 средней сложности (запрос к БД)

Итого:

— простые поля — 6;

— поля средней сложности — 8;

— поля высокой сложности — 4;

— модули на ЯП третьего поколения — 3;

— повторное использование — 0 %;

— опыт команды — очень низкий;

5.4.4 Модель ранней разработки архитектуры

— PERS (возможности персонала) – очень высокий;

— RCPX (надежность и уровень сложности разрабатываемой системы) – очень высокий;

— RUSE (повторное использование компонентов) – низкий;

— PDIF (сложность платформы разработки) – номинальный;

— PREX (опыт персонал) – низкий;

— FCIL (средства поддержки) – очень высокий;

— SCED (график работ) – низкий;

— KSLOC = 3.6 (из метода функциональных точек).

6 Выводы

Методика COCOMO II, как и ее первая версия, позволяет досочно быстро оценить длительность и трудозатраты проекта, основываясь на субъективных данных. В условиях отсутствия объективной информации о предполагаемых трудозатратах особенно важно правильно спрогнозировать характеристики проекта.

Методика функциональных точек позволяет оценить размер программного продукта на этапе его проектирования. С помощью этого подхода можно применять методики COCOMO, которым необходимо знание о размере продукта. Хотя методика функциональных точек является неточной (как и любая методика прогнозирования), этих результатов достаточно для общего понимания объемов работ и получения приблизительных оценок параметров проекта.

В ходе выполнения лабораторной работы было выяснено, что модель композиции приложения дает менее оптимистичный прогноз, по сравнению с моделью ранней архитектуры приложения.