



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 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 СОСОМО II»

Вариант: 1

Студент ИУ7-85Б
(Группа)

(Подпись, дата)

А. В. Толмачев
(И. О. Фамилия)

Студент ИУ7-85Б
(Группа)

(Подпись, дата)

В. А. Лебедев
(И. О. Фамилия)

Преподаватель

(Подпись, дата)

М. Ю. Барышникова
(И. О. Фамилия)

2025 г.

1 Основное задание

1.1 Модель композиции в СОСОМО II

Модель композиции приложения используется на этапе создания прототипов и анализе его осуществимости. Модель композиции приложения — это модель, которая подходит для проектов, созданных с помощью современных инструментальных средств. Единицей измерения служит объектная точка.

Данная модель используется на ранней стадии конструирования ПО, когда:

- рассматривается макетирование пользовательских интерфейсов;
- оценивается производительность;
- определяется степень зрелости технологии.

Модель ориентирована на применение объектных точек. Объектная точка — средство косвенного измерения ПО. Подсчет количества объектных точек проводится с учетом количества экранов, отсчетов и компонентов, требуемых для построения приложения.

Используются следующие правила подсчета объектных точек. В зависимости от количества изображений на дисплее:

1. Простые изображения — одна объектная точка;
2. Изображения умеренной сложности — две объектные точки;
3. Очень сложные изображения — три точки.

В зависимости от количества представленных отчетов:

1. Простые отчеты — две объектные точки;
2. Умеренно-сложные отчеты — пять объектных точек;
3. Сложные отчеты — восемь точек.

Каждый модуль, написанный на языке третьего поколения, считается за десять объектных точек.

Новые объектные точки определяются по следующей формуле:

$$\text{NOP} = \text{Объектные точки} \cdot \frac{(100 - \%RUSE)}{100},$$

где %RUSE — процент повторного использования кода программы.

Трудозатраты вычисляются по следующей формуле:

$$\text{Трудозатраты} = \frac{\text{NOP}}{\text{PROD}},$$

где PROD — оценка скорости разработки.

Длительность выполнения проекта на уровне композиции приложения вычисляется по формуле:

$$\text{Время} = 3 \cdot \text{Трудозатраты}^{0.33+0.2 \cdot (p-1.01)},$$

где p — показатель степени. Значение показателя степени рассчитывается с учетом факторов, влияющих на показатель степени:

$$p = \frac{(\text{PREC} + \text{FLEX} + \text{RESL} + \text{TEAM} + \text{PMAT})}{100} + 1.01.$$

Исходя из задания пользователь описывается следующими полями:

1. Логин;
2. Пароль;
3. Номер водительского удостоверения;
4. Номер банковской карты.

Штраф описывается следующими полями:

1. Номер постановления.
2. Дата постановления.
3. Имя.
4. Фамилия.
5. Отчество нарушителя.
6. Сумма штрафа.

Исходя из условия задания были выделены следующие страницы и формы:

1. Мобильное приложение
 - Страница регистрации (1 форма: логин, пароль, номер ВУ, номер карты + кнопка «Зарегистрироваться»)
 - Страница входа (1 форма: логин, пароль + «Войти»)
 - Страница просмотра штрафов (1 форма: список штрафов + кнопка «Оплатить» + модальное окно подтверждения оплаты)
2. Веб-портал: для пользователей
 - Страница входа (1 форма: логин, пароль + «Войти»)
 - Страница регистрации (1 форма: логин, пароль, номер ВУ, номер карты + кнопка «Зарегистрироваться»)

- Страница просмотра штрафов (1 форма: список штрафов + кнопка «Оплатить» + модальное окно подтверждения оплаты)

3. Веб-портал: для администраторов

- Панель администратора (1 форма: список пользователей + кнопки «Редактировать», «Создать» + модальные окна ввода данные для создания/редактирования пользователей)
- Страница просмотра выплат (1 форма: таблица транзакций)

В таблице 1.1 представлены итоги по количеству форм той или иной сложности.

Таблица 1.1

Сложность формы	Количество форм	Примеры
Низкая	2	Вход (моб/веб)
Средняя	2	Регистрация (моб/веб)
Высокая	4	Просмотр штрафов (моб/веб), выплат, юзеров

Разрабатываемое приложение состоит из пяти модулей. Также из условия известно, что $\%RUSE = 0$, а опытность команды — низкая.

Также известно, что:

- Практически отсутствует опыт в разработке систем подобного типа, поэтому новизна проекта *PREC* — почти полное отсутствие прецедентов, в значительной мере непредсказуемый проект.
- При этом заказчик настаивает на довольно строгом процессе с периодической демонстрацией рабочих продуктов, соответствующих этапам жизненного цикла, поэтому гибкость процесса разработки *FLEX* — случайные послабления в процессе.
- Учитывая новизну проекта для команды, на этапе его подготовки был осуществлен относительно детальный анализ рисков, свойственных архитектуре разрабатываемой системы, поэтому разрешение рисков в архитектуре системы *RESL* — в целом (75%).

- К разработке проекта планируется привлечь довольно слаженную команду высокопрофессиональных разработчиков, поэтому сплоченность команды *TEAM* — повышенная согласованность.
- Организация находится чуть выше второго уровня зрелости процессов разработки, поэтому уровень зрелости процесса разработки *PMAT* — уровень 2 *CMM*.

На рисунке 1.1 представлена оценка трудозатрат и длительности разработки с использованием модели композиции приложения.

The screenshot shows the 'Cocomo II' application window. It contains two main sections for project estimation.

Top Section: Cocomo II - Функциональные точки

- Факторы, влияющие на показатель степени в модели COCOMO II:**
 - PREC: Почти полное отсутствие прецедентов, в значительной мере непредсказуемый проект
 - FLEX: Случайные послабления в процессе
 - REEL: В целом (75 %)
 - TEAM: Повышенная согласованность
 - PMAT: Уровень 2 CMM
- Модель композиции приложения:**
 - Экраны/формы:** Простые (2), Средние (2), Сложные (4)
 - Отчеты:** Простые (0), Средние (0), Сложные (0)
 - Модули:** 5
 - %RUSE:** 0.00
 - Опытность команды:** Низкая
 - Оценка стоимости человеко-месяца (руб):** 80000
- Результат:**

Трудозатраты (чел/мес)	Время (мес)	Бюджет (руб)	Команда (чел)
9.71	6.92	777142.86	2

Bottom Section: Модель ранней разработки архитектуры

- PERL:** Очень высокий
- RCRX:** Очень высокий
- RUSE:** Низкий
- PDIF:** Нормальный
- PREX:** Низкий
- FCIL:** Высокий
- SCED:** Высокий
- Зарплата (руб):** 80000
- Результат:**

Трудозатраты (чел/мес)	Время (мес)	Бюджет (руб)	Команда (чел)
2.28	4.06	182284.23	1

Рисунок 1.1

Средняя численность команды была определена как отношение трудозатрат ко времени, в результате чего было получено, что необходима команда из двух человек.

На основе экспертной оценки стоимости человеко-месяца, равной 80000 рублей, произведена предварительную оценку бюджета проекта, вычисленная как произведение трудозатрат на оценку, в результате чего бюджет составил 777142.

1.2 Метод функциональных точек

Функциональная точка — это единица измерения функциональности программного обеспечения.

Пользователи — это отправители и целевые получатели данных, ими могут быть как реальные люди, так и смежные интегрированные информационные системы.

Функциональные типы — логических группы взаимосвязанных данных, используемых и поддерживаемых приложением:

- внешний ввод (EI) — транзакция получения данных от пользователя;
- внешний вывод (EO) — транзакция перечисления данных пользователю;
- внешний запрос (EQ) — интерактивный диалог с пользователем, требующий от него каких-либо действий и не связанный с вычислением производных данных или обновлением внутренних логических файлов (базы данных);
- внутренний логический файл (ILF) — информация, которая используется во внутренних взаимодействиях системы;
- внешний интерфейсный файл (EIF) — файлы, участвующие во внешних взаимодействиях с другими системами.

Для оценки сложности функциональных типов используются следующие характеристики (их количество):

- DET (data element type) — это уникальное распознаваемое пользователем, неповторяющееся поле данных;
- RET (record element type) — идентифицируемая пользователем логическая группа данных внутри ILF или EIF;
- FTR (file type referenced) — это тип файла, на который ссылается транзакция.

Исходя из постановки задачи был выполнен анализ страниц приложения.

1.2.1 Страница регистрации в системе

На данной странице осуществляется ввод логина, пароля, номера ВУ и карты пользователя для регистрации в системе. Страница содержит четыре поля ввода и одну командную кнопку.

1. **Внешний ввод (EI):** Сохранение данных регистрации. $DET = 4$ (логин, пароль, номер ВУ, карта), $RET = 1$. Уровень низкий.
2. **Внешний запрос (EQ):** Отсутствует.
3. **Внешний вывод (EO):** Отсутствует.
4. **Внутренний логический файл (ILF):** Таблица пользователей бд с полями идентификатора, логина, пароля, номера ВУ, карты и типа. $DET = 6$, $RET = 1$. Уровень низкий.
5. **Внешний логический файл (EIF):** Отсутствует.

Итого по странице:

- $ILF = 1$, низкий уровень.
- $EI = 1$, низкий уровень.
- $EQ = 0$.

1.2.2 Страница авторизации

На данной странице осуществляется ввод логина и пароля пользователя. Страница содержит два поля ввода и одну командную кнопку.

1. **Внешний ввод (EI):** Проверка логина и пароля на сервере. $DET = 2$, $FTR = 1$. Уровень низкий.
2. **Внешний запрос (EQ):** Отсутствует.
3. **Внешний вывод (EO):** Отсутствует.
4. **Внутренний логический файл (ILF):** Таблица пользователей. $DET = 6$, $RET = 1$. Уровень низкий.
5. **Внешний интерфейсный файл (EIF):** Отсутствует.

Итого:

- $ILF = 1$ (низкий уровень);
- $EI = 1$ (низкий уровень);
- $EQ = 0$;
- $EO = 0$.

1.2.3 Страница просмотра штрафов

На этой странице отображаются неоплаченные штрафы. Страница не содержит полей ввода и есть командная кнопка «Оплатить штраф». После оплаты отображается модальное окно, которое содержит статус оплаты.

1. **Внешний ввод (EI):** Обработка оплаты штрафа. $DET = 1$, $FTR = 2$. Уровень низкий.
2. **Внешний запрос (EQ):** Отсутствует.
3. **Внешний вывод (EO):**
 - информация о штрафах. $DET = 6$, $FTR = 1$. Уровень низкий.
 - статус оплаты. $DET = 1$, $FTR = 1$. Уровень низкий.
4. **Внутренний логический файл (ILF):** Таблица оплат, которая содержит номер карты, номер счета в ГИБДД, сумму оплаты.. $DET = 3$, $RET = 2$. Уровень низкий.
5. **Внешний интерфейсный файл (EIF):** Таблица штрафов, которая содержит номер постановления, дату постановления, имя, фамилию, отчество и сумму штрафа.. $DET = 6$, $RET = 2$. Уровень низкий.

Итого:

- $ILF = 1$ (низкий уровень)
- $EIF = 1$ (низкий уровень)
- $EI = 1$ (низкий уровень)
- $EO = 2$ (оба низкого уровня)
- $EQ = 0$

1.2.4 Страница просмотра пользователей

На этой странице отображается список пользователей. Страница не содержит полей ввода и есть две командные кнопки. При нажатии на любую строку списка появляется контекстное меню с возможностью отредактировать или создать пользователя.

1. Внешний ввод (EI):

- Редактирование пользователя. $DET = 5$, $FTR = 1$. Уровень низкий.
- Создание пользователя. $DET = 5$, $FTR = 1$. Уровень низкий.

2. Внешний запрос (EQ): Отсутствует.

3. Внешний вывод (EO): Список пользователей. $DET = 5$, $FTR = 1$.
Уровень низкий.

4. Внутренний логический файл (ILF): Таблица пользователей.
 $DET = 6$, $RET = 1$. Уровень низкий.

5. Внешний интерфейсный файл (EIF): Отсутствует.

Итого:

- $ILF = 1$ (низкий)
- $EI = 2$ (оба низкого уровня)
- $EO = 1$ (низкий)
- $EQ = 0$

1.2.5 Страница просмотра выплат

На этой странице отображаются все выплаты. Страница не содержит полей ввода.

1. **Внешний ввод (EI):** Отсутствует.
2. **Внешний запрос (EQ):** Отсутствует.
3. **Внешний вывод (EO):** Информация о выплатах. $DET = 3$, $FTR = 1$.
Уровень низкий.
4. **Внутренний логический файл (ILF):** Таблица выплат. $DET = 3$,
 $RET = 2$. Уровень низкий.
5. **Внешний интерфейсный файл (EIF):** Отсутствует.

Итого:

- $ILF = 1$ (низкий)
- $EO = 1$ (низкий)
- $EI = 0$
- $EQ = 0$

Таким образом, всего функциональных типов:

- EI : 5 низкого уровня;
- EO : 4 низкого уровня;
- EQ : 0;
- ILF : 5 низкого уровня;
- EIF : 1 низкого уровня.

Для уточнения числа функциональных точек используются следующие значения характеристики продукта:

1. Обмен данными — 5

2. Распределённая обработка — 5
3. Производительность — 3
4. Эксплуатационные ограничения по аппаратным ресурсам — 0
5. Транзакционная нагрузка — 3
6. Интенсивность взаимодействия с пользователем (оперативный ввод данных) — 2
7. Эргономические характеристики, влияющие на эффективность работы конечных пользователей — 0
8. Оперативное обновление — 4
9. Сложность обработки — 4
10. Повторное использование — 3
11. Лёгкость инсталляции — 0
12. Лёгкость эксплуатации/администрирования — 3
13. Портруемость — 5
14. Гибкость — 0

Результаты расчета числа функциональных точек представлены на рисунке 1.2:

Характеристики продукта

- Обмен данными: 5
- Распределенная обработка: 5
- Производительность: 3
- Эксплуатационные ограничения по аппаратным ресурсам: 0
- Транзакционная нагрузка: 3
- Оперативный ввод данных: 2
- Эргономические характеристики, влияющие на эффективность работы конечных пользователей: 0
- Оперативное обновление: 4
- Сложность обработки: 4
- Повторное использование: 3
- Легкость установки: 0
- Легкость эксплуатации: 3
- Портруемость: 5
- Гибкость: 0

Процент использования языков программирования

ASM	0.00	Ada 95	0.00
C	0.00	Visual Basic	0.00
Cobol	0.00	Visual C++	0.00
Fortran	0.00	Delphi 5	0.00
Pascal	0.00	Perl	0.00
C++	0.00	Prolog	0.00
C#	0.00	ANSI SQL	30.00
Java	60.00	Java Script	10.00

Расчет числа функциональных точек

	Низкий	Средний	Сложный	Итого
Внешние вводы	5	0	0	15
Внешние выводы	4	0	0	16
Внешние запросы	0	0	0	0
Внутренние логические файлы	5	0	0	35
Внешние интерфейсные файлы	1	0	0	5

Результаты:

Функциональные точки	Коррекция ФТ	Строки кода
71	72.42	2991

Рассчитать

Рисунок 1.2

Таким образом, первоначальное число функциональных точек равно 71, а скорректированное — 72,42.

Процентные соотношения языков программирования:

- *SQL* — 30%;
- *JavaScript* — 10%;
- *Java* — 60%.

С учетом данного соотношения с использованием таблиц соответствия числа функциональных точек строкам кода было определено, что проект будет состоять из 2991 строк кода.

1.3 Модель ранней архитектуры в COSOMO II

Для получения приблизительных оценок проектных затрат периода выполнения проекта перед тем, как будет определена архитектура в целом, применяется модель ранней разработки архитектуры. Для такой модели характерны оценки умеренной точности и ясно понимаемые особенности проекта, требования и архитектура. В качестве единиц измерения используются функциональные точки или KSLOC.

Трудозатраты вычисляются так:

$$\text{Трудозатраты} = 2.45 \cdot \text{EArch} \cdot \text{Размер}^p,$$

где Размер = KSLOC, EArch определяется через произведение множителей трудоемкости:

$$\text{EArch} = \text{PERS} \cdot \text{RCPX} \cdot \text{RUSE} \cdot \text{PDIF} \cdot \text{PREX} \cdot \text{FCIL} \cdot \text{SCED}.$$

Время разработки получается в соответствии с формулой:

$$\text{Время} = 3 \cdot \text{Трудозатраты}^{0.33+0.2 \cdot (p-1.01)},$$

где p — показатель степени. Значение показателя степени рассчитывается с учетом факторов, влияющих на показатель степени:

$$p = \frac{(\text{PREC} + \text{FLEX} + \text{RESL} + \text{TEAM} + \text{PMAT})}{100} + 1.01.$$

Исходя из условия задания следует, что:

1. надежность и сложность продукта (RCPX) — очень высокие;
2. повторное использование компонентов (RUSE) — низкий;
3. опытность персонала (PERS) — очень высокий;
4. способности персонала (PREX) — низкий;
5. сложность платформы (PDIF) — номинальный;
6. возможности среды (FCIL) — высокий;

7. сроки (SCED) — высокий.

Число строк кода исходя из метода функциональных точек равно 2991.

На рисунке 1.3 представлена оценка трудозатрат и длительности разработки с использованием модели ранней разработки архитектуры.

Экраны/Формы	Отчеты	Модули	Оценки
Простые: 2	Простые: 0	%RUSE: 0.00	Опытность команды: Низкая
Средние: 2	Средние: 0	Оценка стоимости человеко-месяца (руб): 80000	
Сложные: 4	Сложные: 0		

Трудозатраты (чел/мес)	Время (мес)	Бюджет (руб)	Команда (чел)
9.71	6.92	777142.86	2

Трудозатраты (чел/мес)	Время (мес)	Бюджет (руб)	Команда (чел)
2.28	4.06	182284.23	1

Рисунок 1.3

Средняя численность команды была определена как отношение трудозатрат ко времени, в результате чего было получено, что необходима команда из одного человека.

На основе экспертной оценки стоимости человеко-месяца, равной 80000 рублей, произведена предварительную оценку бюджета проекта, вычисленная как произведение трудозатрат на оценку, в результате чего бюджет составил 182284.

1.4 Вывод

Модель СОСОМО II учитывает множество факторов, влияющих на экономические характеристики производства сложных программных продуктов. При этом метод является довольно универсальным и может поддерживать различные размеры, режимы и уровни качества продукта. Также возможна высокая степень достоверности калибровки с опорой на предыдущий опыт специалистов.

Благодаря этим факторам оценка, полученная с помощью данной модели, является более точной и приближенной к реальности, по сравнению с СОСОМО.

При этом результаты все еще зависят от точности оценки размера программного продукта, что значительно влияет на точность прогноза трудозатрат, длительности разработки и численности специалистов.

Модель композиции приложения позволяет оценить затраты на проект вне зависимости от оценки количества строк кода: оценивается только количество модулей и количество изображений/отчетов на дисплее.

Метод функциональных точек позволяет получить оценку количества строк кода, которую можно использовать для модели ранней разработки архитектуры.