**Лабораторная работа №4**

**Оценка трудоемкости и сроков разработки ПО**

**(COCOMO II - Constructive Cost Model)**

**Цель работы:** ознакомление с методами оценки трудоемкости и сроков разработки программного обеспечения на примере методики COCOMO II (Constructive Cost Model)

**Отчет:** отчет по лабораторной работе представляется в электронном виде должен включать:

1. Описание оцениваемого проекта.

2. Изменяемые параметры проекта и их значения.

3. Результаты оценки.

При оценке проекта необходимо понимать, что любая оценка это всегда вероятностное утверждение. Если сказать, что трудоемкость данного пакета работ составляет, например, М чел.-мес., то такая оценка ничего не скажет о вероятности того, что на реализацию этого пакета потребуется не более, чем М чел.-мес.

Оценка должна быть вероятностным утверждением, т.е. для нее существует некоторое распределение вероятности, которое может быть очень широким (высокая неопределенность) или достаточно узким (низкая неопределенность).

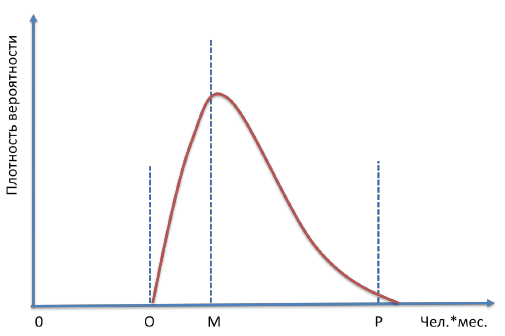


Рисунок 1. Оценка — всегда вероятностная величина

Оценка трудоемкости и сроков разработки программного обеспечения может осуществляться на основе использования собственного опыта или опыта экспертов, а также с использованием методик на основе отраслевого опыта.

Использование собственного опыта или опыта коллег, экспертов, полученного в похожих проектах, это наиболее прагматичный подход, который позволяет получить достаточно реалистичные оценки трудоемкости и срока реализации программного проекта, быстро и без больших затрат.

Если же собственный опыт аналогичных проектов отсутствует, а коллеги-эксперты недоступны, то нам не остается ничего другого, как использовать формальные методики, основанные на обобщенном отраслевом опыте. Среди них наибольшее распространение получили два подхода:

* FPA IFPUG — метод функциональных точек,
* метод COCOMO II, Constructive Cost Model.

При этом IFPUG FPA наиболее предпочтительно применять на стороне заказчика, а СОСОМО II – на стороне разработчика, так как для заказчика разница в конкретных условиях разработки не важна, а для разработчика – важна.

**Методика СОСОМО**

Методика COCOMO позволяет оценить трудоемкость и время разработки программного продукта. Впервые была опубликована Бари Боэмом в 1981 году в виде результат анализа 63 проектов компании «TRW Aerospace». Анализировались проекты объемом от 2 до 100 тысяч строк кода, на языках программирования от ассемблера до высокоуровневого языка PL/1 90 (Programming Language I, 1964), основанные на каскадной модели жизненного цикла разработки ПО.

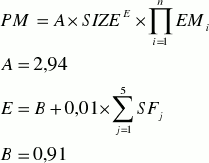
В 1997 методика была усовершенствована и получила название COCOMO II. Калибровка параметров производилась по 161 проекту разработки. В модели используется формула регрессии с параметрами, определяемыми на основе отраслевых данных и характеристик конкретного проекта.

Различаются две стадии оценки проекта: предварительная оценка на начальной фазе и детальная оценка после проработки архитектуры.

В модели СОСОМО используется ряд допущений:

* исходный код конечного продукта включает в себя все (кроме комментариев) строки кода;
* начало цикла разработки совпадает с началом разработки продукта, окончание совпадает с окончанием приемочного тестирования, завершающим стадию интеграции и тестирования (работа и время, затрачиваемые на сбор и анализ требований, оцениваются отдельно как дополнительный процент от оценки разработки в целом);
* виды деятельности включают в себя только непосредственно направленные на выполнение проекта работы, в них не входят обычные вспомогательные виды деятельности, такие, как административная поддержка, техническое обеспечение и проч.;
* человеко-месяц состоит из 152 час.;
* проект управляется надлежащим образом, в нем используются стабильные требования.

Уравнения СОСОМО для оценки номинальных значений трудоемкости (в чел.-мес) имеют следующий вид (формулы 1, 2)

 (1,2)

где *SIZE* — размер продукта в KSLOC (количество строк кода) или в количестве функциональных точек без учета поправочных коэффициентов (UFP), определенном по методике 1FPUG, с последующим преобразованием в количество строк кода;

*EMi* — (effort multiplier) множители трудоемкости;

*SFj* — (scale factor) факторы масштаба;

*n*=7 — для предварительной оценки;

*n*=17 — для детальной оценки;

Данные А и В приведены для предварительной оценки, для детальной оценки А,2.45.

Главной особенностью методики является то, что для оценки трудоемкости необходимо знать размер программного продукта в тысячах строк исходного кода (KSLOC, Kilo Source Lines Of Code). Размер программного продукта может быть, например, оценен экспертами с применением метода PERT. Если анализ продукта выполнен методом функциональных точек, то его размер может быть рассчитан с использованием собственных статистических данных фирмы или с использованием статистики по отрасли (табл. 1).

Таблица 1. Оценка количества строк, необходимых на реализацию одной не выровненной функциональной точки для некоторых распространенных языков программирования



**Факторы масштаба.**

В методике используются пять факторов масштаба *SFi*, которые определяются следующими характеристиками проекта:

1. PREC — прецедентность, наличие опыта 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).

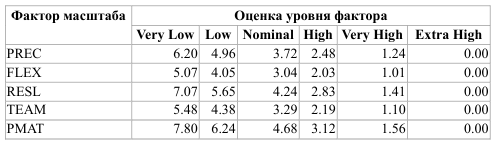
3. RESL — архитектура и разрешение рисков (Very Low — риски неизвестны/не проанализированы; Extra High — риски разрешены на 100%).

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)

Значения фактора масштаба в зависимости от оценки его уровня приведены в Таблице 2.

Таблица 2. Значение фактора масштаба,



CMM (Capability Maturity Model) — модель зрелости процессов создания ПО, или эволюционная модель развития способности компании разрабатывать качественное программное обеспечение. За основу при оценке способности организации качественно выполнять работу, которая (способность) была названа зрелостью, создатели модели взяли процессы организации. Дальше они сделали несколько нетривиальных предположений, которые впоследствии были приняты и признаны справедливыми многими ИТ-специалистами (а может быть, и большинством из них).

*Предположение 1*. Существуют качественно отличающиеся *уровни зрелости* *проектной организации*, разрабатывающей *информационные системы* (в модели СММ таких уровней пять).

*Предположение 2*. Всякая организация-разработчик заинтересована в переходе на более высокий уровень зрелости (не только для того, чтобы повысить свои шансы в борьбе за контракты Министерства обороны, но и в целях собственного совершенствования).

*Предположение 3*. Переход возможен только на следующий *по* порядку уровень. "Перескочить" через уровень нельзя (точнее, риски для организации при этом резко возрастают).

Таким образом, уровни образуют "лесенку", *по* которой поднимается организация *по* мере собственного развития. Каждый уровень характеризуется определенными составом и свойствами процессов организации.

**Уровень 1 "Начальный"**. Производственный процесс в целом характеризуется как создаваемый каждый раз под конкретный проект, а иногда даже как хаотический. Определены лишь некоторые процессы, и успех проекта зависит от усилий индивидуумов.

**Уровень 2 "Повторяемый"**. Установлены основные процессы управления проектом, позволяющие отслеживать *затраты*, следить за графиком *работ* и функциональностью создаваемого программного решения. Установлена дисциплина процесса, необходимая для повторения достигнутых ранее успехов в проектах разработки подобных приложений.

**Уровень 3 "Определенный"**. Производственный процесс документирован и стандартизован как для управленческих *работ*, так и для проектирования. Этот процесс интегрирован в стандартный производственный процесс организации. Во всех проектах используется утвержденная *адаптированная* версия стандартного производственного процесса организации.

**Уровень 4 "Управляемый"**. Собираются подробные количественные показатели производственного процесса и качества создаваемого продукта. Как производственный процесс, так и продукты оцениваются и контролируются с количественной точки зрения.

**Уровень 5 "Оптимизирующий"**. Постоянное совершенствование процесса достигается благодаря количественной обратной связи с процессом и реализации в нем передовых идей и технологий.

Пример экспоненциального коэффициента масштаба — коэффициент зрелости процессов (РМАТ— Process Maturity). Значение коэффициента РМАТ зависит в основном от уровня зрелости процессов в соответствии с моделью СММ. Процедура определения значения РМАТ основана на определении процента соответствия для каждой из 18 основных групп процессов (key process areas — КРА), определенных в СММ.

**Множители трудоемкости**

Для предварительной оценки трудоёмкости программного проекта необходимо определить уровень семи множителей трудоемкости *Mi*:

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; ExtraHigh — требуется использование в других продуктах);

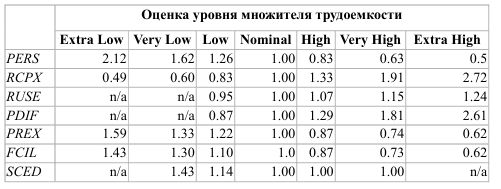
4. PDIF — сложность платформы разработки (Extra Low — специальные ограничения по памяти и быстродействию отсутствуют, платформа стабильна; Extra High — жесткие ограничения по памяти и быстродействию, платформа нестабильна);

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 определяется их числовыми значениями, которые представлены в Таблице 3.

Таблица 3. Значения множителей трудоемкости, в зависимости от оценки их уровня 

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.

###### **Оценка многокомпонентного продукта**

Для того, чтобы адекватно спланировать проект и оценить его трудоемкость, необходимо выполнить предварительное проектирование программного продукта. В результате декомпозиции мы получаем некоторое количество компонентов (N), которые составляют программный продукт.

Следует понимать, что суммарная трудоемкость проекта не равна простой сумме трудоемкостей разработки каждого из компонентов:

http://citforum.ru/SE/project/arkhipenkov_lectures/f6.gif (3)

Простая сумма не учитывает взаимосвязи компонентов и трудозатраты на их интеграцию.

Методика COCOMO II определяет следующую последовательность вычисления трудоемкости проекта при многокомпонентной разработке.

1. Суммарный размер продукта рассчитывается, как сумма размеров его компонентов:

http://citforum.ru/SE/project/arkhipenkov_lectures/f7.gif (4)

1. Базовая трудоемкость проекта рассчитывается по формуле:

http://citforum.ru/SE/project/arkhipenkov_lectures/f8.gif (5)

1. Затем рассчитывается базовая трудоемкость каждого компонента:

http://citforum.ru/SE/project/arkhipenkov_lectures/f9.gif (6)

1. На следующем шаге рассчитывается оценка трудоемкости компонентов с учетом всех множителей трудоемкости, кроме множителя *SCED.*

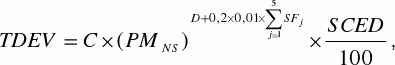
http://citforum.ru/SE/project/arkhipenkov_lectures/f10.gif (7)

1. И, наконец, итоговая трудоемкость проекта определятся по формуле:

http://citforum.ru/SE/project/arkhipenkov_lectures/f11.gif (8)

###### **Оценка длительности проекта**

Длительность проекта (time to develop) в методике COCOMO II рассчитывается по формуле:

 (9)

где *С = 3,67; D = 0,28;*

*PMNS* — трудоемкость проекта без учета множителя *SCED,* определяющего сжатие расписания;

SCED — степень сжатия расписания.

***Пример:*** *Средней сложности проект в 100 тыс. строк. Для него все коэффициенты-мультипликаторы =1,0, Е=1,15.*

*Трудоёмкость составит: РМ = 2,94х(100)\*1,15 = 586,61 чел.-мес.*

*TDEV = 3,67(586,6)\*(0,28+0,2х(1,15-0,91))=3,67(586,7)\*0,328=29,7 мес.*

*Средняя численность персонала программистов:*

*П = 586,6/29,7 = 19,75, т. е. 20 чел.*

**Программы для проведения расчетов по методике COCOMO II**

Загрузить и установить файл CII2000.exe можно по ссылке

<http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII2000.exe> с сайта <http://csse.usc.edu/csse/research/COCOMOII/cocomo_downloads.htm>

© 2000 University of Southern California, Center for Systems and Software EngineeringОнлайн-программа для проведения расчетов :

<http://csse.usc.edu/tools/COCOMO.php>

##### **Выводы**

Оценка трудоемкости должна быть вероятностным утверждением. Это означает, что для нее существует некоторое распределение вероятности, которое может быть очень широким, если неопределенность высокая, или достаточно узким, если неопределенность низкая.

Использование собственного опыта или опыта коллег, полученного в похожих проектах, это наиболее прагматичный подход, который позволяет получить достаточно реалистичные оценки трудоемкости и срока реализации программного проекта, быстро и без больших затрат.

Если собственный опыт аналогичных проектов отсутствует, а коллеги-эксперты недоступны, то необходимо использовать формальные методики, основанные на обобщенном отраслевом опыте.

Не реалистичность оценок один из серьезнейших демотивирующих факторов для участников проектной команды. Недооценка приводит к ошибкам планирования и неэффективному взаимодействию. Агрессивные сроки, постоянное давление, сверхурочные, авралы служат причиной того, что затраты на проект растут экспоненциально и неограниченно.

**Задание.**

Описать проект создания медицинского оборудования и выполнить оценку его трудоемкости и объемов работ любым из изученных методов. Написать сценарий выполнения этот проект и соотнести его оценку с фактическим результатом.

Современное медицинское оборудование базируется на широком использовании вычислительных ресурсов, позволяющих получать, накапливать и анализировать большие объемы информации, а также выдавать полученные результаты в понятном человеку виде (диаграммы, изображения, таблицы и т.п.). Поэтому современная медицинская техника представляет собой аппаратно-программные комплексы, требующие для своей работы специально разработанного программного обеспечения, к которому предъявляются жесткие требования по быстродействию, надежности и точности вычислений.

Программное обеспечение для проекта создания медицинского оборудования (эхокардиографа) включает в себя четыре функционально-обособленных модуля:

**−**встраиваемая программа для блока цифровой обработки сигналов от датчиков – реализуется на ассемблере для процессора TMS320C54x;

**−**драйвер для блока цифровой обработки сигналов от датчиков под операционные системы Windows – реализуется на языке C++ с применением вставок на ассемблере x86;

**−**клиентская программа для управления эхокардиографом под операционные системы Windows – реализуется на языке C#;

**−**серверная программа для хранения данных проведенных измерений под операционные системы Windows – реализуется на языке C#.

Приблизительный размер компонентов (в не выровненных функциональных точках) определяется на основе опыта предыдущих разработок. В работе принять следующие данные:

Встраиваемая программа для блока ЦОС (Assembler) ­­- 25

Драйвер для блока ЦОС (Assembler) - 34

Драйвер для блока ЦОС (C++) - 76

Клиентская программа (C#) - 254

Серверная программа (C#) - 392

**Выполнение работы.**

1.Опишите проект. Сделайте вывод о целесообразности расчета многокомпонентной разработки. Обоснуйте выделение основных компонентов (с учетом используемых языков программирования) (см. задание).

2. Определите количество строк, необходимых для реализации одной не выровненной функциональной точки каждого компонента, используя данные табл. 1 (для наиболее вероятного случая).

3. Определите число строк для каждого компонента суммарный размер продукта в таблице 4.

Таблица 4 – Размер компонентов программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| № | Компонент | Число точек | Число строк |
| 1 |  |  |  |
| … |  |  |  |
| Всего: | |  |  |

4. Рассчитайте значения параметра Е. Для определения значения параметра *E*по формуле (2) необходимо определиться со значениями факторов масштаба. В работе принять следующие условия:

PREC – предполагается, что предприятие имеет большой опыт разработки программного обеспечения, поэтому принимается значение «Высокий»;

FLEX – на предприятии используются различные подходы к гибкости процесса разработки в зависимости от отдела, поэтому принимается значение «Нормальный»;

RESL – на предприятии отсутствует комплексная система управления рисками, приблизительной оценкой рисков занимаются руководители проектов, основываясь на своем опыте, поэтому принимается значение «Нормальный»;

TEAM – на предприятии каждый проект закрепляется за отдельным отделом, состоящим из давно работающих совместно программистов, поэтому принимается значение «Высокий»;

PMAT – на предприятии установлен определенный, четко документированный процесс работы, не зависящий от отдельных личностей, поэтому принимается значение «Нормальный».

5. Определите базовую трудоемкость проекта. Для расчета базовой трудоемкости проекта необходимо задать значение множителя трудоемкости SCED. Исходя из того, что строгих сроков на разработку предприятию не установлено, принимается значение «Нормальный».

6. Значения базовой трудоемкости для каждого из компонентов сведите в таблицу 5. Также приведите соответствующие значения множителей трудоемкости, вычисленные значения трудоемкостей для всех компонентов и итоговое значение трудоемкости для всего проекта в целом.

При выборе значений множителей используйте следующие соображения:

PERS – предполагается, что предприятие имеет низкую текучесть кадров и достаточно высокую квалификацию персонала, что позволяет принять значение «Очень высокий»;

RCPX – как было сказано ранее, использование продукта в медицинских целях предполагает повышенную надежность продукта, поэтому принимается значение «Экстра высокий»;

RUSE – компоненты, непосредственно связанные с обработкой данных (встраиваемая программа и драйвер), являются узкоспециализированными, поэтому для них принимается значение «Низкий», для остальных компонентов – «Нормальный»;

PDIF – для компонента, реализующего встраиваемую программу, имеет смысл принять значение «Экстра высокий» (поскольку на нее накладываются жесткие ограничения по памяти и быстродействию), для компонентов, реализующих драйвер, – « Высокий», для остальных – « Нормальный»;

PREX – предыдущий опыт персонала позволяет принять значение «Высокий» для всех компонентов;

FCIL – используемое на предприятии для разработки современное оборудование и системы коммуникации позволяют назначить значение «Высокий».

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | Компонент | *PM k B* | PERS | RCPX | RUSE | PDIF | PREX | FCIL | *PM*' *k* |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Всего: | |  |  |  |  |  |  |  |  |

Таблица 5 – Трудоемкость компонентов

7. Сделайте вывод о трудоемкости разработки проекта

8. Ознакомьтесь с возможностями программных продуктов для проведения расчетов на примере:

<http://csse.usc.edu/tools/COCOMO.php>