Секція 6. Прикладні засоби програмування та програмне забезпечення

УДК 519.6

МЕТОД ВИЗНАЧЕННЯ ПРИБЛИЗНОЇ ТРИВАЛОСТІ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Мельник М.В.
Тернопільський національний економічний університет, магістр

I. Постановка проблеми

Тривалість розробки програмного забезпечення є одною з найважливіших характеристик проектів в галузі інформаційних технологій. Інформація про тривалість розробки певного функціоналу дозволяє здійснювати планування як матеріальних так і людських ресурсів необхідних для його реалізації. Без цієї інформації неможливо спрогнозувати дату закінчення проекту та його вартість.

В індустрії розробки програмного забезпечення на сьогоднішній день найпоширенішими є методи визначення тривалості розробки ПЗ на основі експертних оцінок. Дані методи мають ряд суттєвих недоліків:

- для успішного оцінювання необхідний значний досвід як в розробці ПЗ так і в оцінках тривалості розробки
- суб'єктивність експертних оцінок
- неможливість перевірки та легкого відтворення зроблених оцінок.

Ці недоліки можливо усунути за допомогою використання модельних методів визначення тривалості розробки ПЗ.

За основу базової моделі визначення тривалості розробки програмного забезпечення була вибрана модель COCOMO II, а саме її варіант, що призначений для оцінювання ранніх стадій розробки програмного забезпечення[1]. Тривалість визначається на основі інформації про трудомісткість, яка в свою чергу визначається на основі розміру програмного забезпечення. Розмір програмного забезпечення вимірюється в тисячах рядків вхідного коду і визначається на основі аналізу UML діаграм варіантів використання.

Окрім розміру модель COCOMO II для оцінювання ранніх стадій розробки програмного забезпечення також залежить від 13 інших параметрів. У випадку відсутності історичних даних про попередні проекти визначення значень цих параметрів покладається на експерта і пов'язане з певним рівнем неточності і суб'єктивності. Для того щоб краще керувати цією неточністю та ризиками пов'язаними з нею даний підхід використовує Монте-Карло симуляцію на основі діапазонів можливих значень.

II. Мета роботи

Метою дослідження є розробка методу визначення тривалості розробки програмного забезпечення використовуючи регресійну параметричну модель COCOMO II. Метод базується на визначенні функціонального розміру ПЗ на основі аналізу діаграми варіантів використання (use case diagram) та проведенні Монте-Карло симуляції.

III. Метод визначення тривалості розробки програмного забезпечення

Модель COCOMO II для оцінювання ранніх стадій розробки програмного забезпечення визначає трудомісткість розробки програмного забезпечення (людино-місяці) так:

$$PM = A \times Size^{E} \times \sum_{i=1}^{7} EM_i ,$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j , \quad (1)$$

де калібровочні константи A та B дорівнюють відповідно 2.94 та 0.91 [2]; EM_i – мультиплікативні фактори; SF_j – експоненційні фактори COCOMO, $Size$ – розмір ПЗ, має бути вираженим в одиницях KSLOC (1 KSLOC = 1000 SLOC, рядки вхідного коду).

Тривалість розробки (time to develop – TDEV) оцінюється за формулою

$$TDEV = C \times (PM)^F,$$
$$F = D + 0.2 \times (E - B), \quad (2)$$

де константи C та D дорівнюють відповідно 3,67 та 0,28[2].

Константи A та B можна калібрувати використовуючи історичні дані про виконані раніше подібні проекти в межах організації.

В [1] зазначено що кількість стрічок початкового коду можна приблизно спрогнозувати аналізуючи вимоги до програмного забезпечення методом FPA (Function Point Analysis) та перетворивши функціональний розмір в KSLOC. За стандартом ISO/IEC, функціональний розмір програмного забезпечення є кількісною мірою його функціональності та визначається як кількісна оцінка функціональних вимог користувача до ПЗ. Функціональні вимоги визначають процеси та процедури, які будуть виконуватися ПЗ. В [3] запропоновано альтернативний спосіб визначення функціонального розміру на основі аналізу UML Use Case діаграм. Для цього пропонується на основі UML стереотипів ввести додаткові позначення що встановлюватимуть зв'язок між елементами Use Case діаграми та транзакціями в FPA. Метод FPA розрізнює три типи транзакцій:

Зовнішній ввід. Процес, у якому дані перетинають межу системи, при цьому вони вводяться в неї та відбувається модифікація внутрішнього її стану або файлів даних, якими система керує. Прикладом може бути введення даних користувачем з клавіатури. Для позначення таких транзакцій пропонується використання стереотипу UML «external input».

Зовнішній вивід. Процес, у якому дані перетинають межу системи, при цьому з неї виводяться дані, отримані внаслідок обробки або проведених розрахунків. Приклад – відображення статистичних графіків на екрані або їх друк на принтері. Для позначення таких транзакцій пропонується використання стереотипу UML «external output».

Зовнішній запит. Процес, у якому дані перетинають межу системи, при цьому з неї виводяться необроблені дані. Прикладом може бути відображення тексту, раніше введеного користувачем. Для позначення таких транзакцій пропонується використання стереотипу UML «external inquiry».

Після ідентифікації транзакцій на діаграмі варіантів використання їм присвоюються відповідні вагові коефіцієнти на основі важливості тих чи інших функцій для кінцевого користувача. Для знаходження загальної функціональної складності для кожної транзакції береться її значення складності відповідно до вагового коефіцієнта та сумується[1].

Задля кращого контролю над неточностями у вхідних даних для параметричної моделі запропоновано використовувати Монте-Карло симуляцію.

Висновки

У роботі описано метод та найважливіші аспекти визначення приблизної тривалості розробки програмного забезпечення. Даний метод базується на оцінюванні розміру ПЗ за діаграмами варіантів використання та проведенні Монте-Карло симуляції. Метод дозволить менеджерам та розробникам краще розуміти процес оцінювання тривалості розробки ПЗ та фактори, що впливають на нього. Наступні роботи можуть включати детальніший опис процесу калібрації моделі на даних попередніх проектів та деталізацію підходу до Монте-Карло симуляції оцінки тривалості розробки ПЗ.

Список використаних джерел

1. CSE, 1999: Center for Software Engineering. COCOMO II Model Definition Manual. // Computer Science Department, USC Center for Software Engineering, 1999. – 37 p.
2. CSE, 1999: Center for Software Engineering. COCOMO II Reference Manual. // Computer Science Department, USC Center for Software Engineering, 1999. – 86 p.
3. Стрєлов І. А., Ігнатенко П.П. Ідентифікація та відображення функціональних елементів fpa-методу в uml-моделі створюваної системи для оцінювання її економічних характеристик // Пробл. програм. – 2005. – №1. – С. 38-51.