Електронне наукове фахове видання "Ефективна економіка" включено до переліку наукових фахових видань України з питань економіки

Ефективна ЕКОНОМІКА

Дніпропетровський державний аграрно-економічний університет

№ 6, 2011

УДК 331; 004.42

В. В. Колдовський,
кандидат економічних наук, доцент кафедри економічної кібернетики,
Українська академія банківської справи НБУ, м. Суми

ПІДХОДИ ДО ОЦІНКИ ЕКОНОМІЧНИХ ПОКАЗНИКІВ ПРОГРАМНИХ ПРОЕКТІВ НА ОСНОВІ ФУНКЦІОНАЛЬНИХ ТОЧОК ТА ЇХ ПОХІДНИХ

Досліджуються питання оцінки економічних показників проектів зі створення інформаційних систем на основі функціональних точок та їх похідних. Розглянуті перспективи використання альтернативних підходів. Запропоновано деякі рекомендації стосовно використання систем оцінки ключових економічних показників у реальних проектах.
Ключові слова: *управління програмними проектами, програмне забезпечення, трудомісткість, вартість, тривалість, функціональні точки*

Article shows how to evaluate economic performance of projects on creation of information systems based on functional points and their derivatives. Perspectives of alternative approaches reviewed. Some recommendations for using the measurement systems of key economic indicators in real projects were offered.
Keywords*: managing software projects, software complexity, cost, duration, functional point*

Вступ. Однією із найважливіших проблем управління програмними проектами є попередня оцінка таких економічних показників як вартість та тривалість реалізації проекту, які, в свою чергу, базуються на оцінці трудомісткості проекту, обсягів робіт, їх складності та ін. На відміну від сфери матеріального виробництва, де вирішення подібних задач відбувається достатньо стабільно і прогнозовано за рахунок нормування витрат і термінів виконання робіт, у сфері розробки програмного забезпечення (ПЗ), що відрізняється значним рівнем інноваційності і непередбачуваності процесів, використання подібних підходів суттєво ускладнено і потребує інших, які враховують специфіку такої діяльності.

Постановка задачі. Формування надійної системи оцінки економічних показників програмних проектів є особливо актуальною задачею у цій галузі, оскільки недоліки існуючих систем оцінок становлять цілу низку ризиків по проектам: починаючи від порушення термінів виконання і закінчуючи повним провалом з причин недооцінки обсягів і складності робіт, які підлягають виконанню. Водночас достатньо традиційним академічним підходом до вирішення даної задачі є побудова математичних моделей на основі емпіричних даних, які встановлюють зв'язок між показниками обсягу робіт по проекту (як правило – його розмір) і трудомісткістю та вартістю реалізації. Проте сама процедура оцінки розміру дуже часто виявляється найслабшим місцем подібних підходів і її вдосконалення можливе за рахунок використання незаслужено забутого підходу на основі функціональних точок, обгрунтування доцільності застосування якого і ставиться у якості задачі перед даним дослідженням.

Аналіз джерел і публікацій. Науковий пошук ефективних методів оцінки програмних проектів відбувався практично з моменту виникнення самої галузі розробки ПЗ, однак найбільш активно дослідження у цьому напрямку почали проводитися у 1970-х роках, коли стало зрозуміло, що традиційні методи, запозичені із інших галузей, діють неефективно. Справжнім поштовхом до наукових досліджень у цьому напрямку стала публікація у 1975 році книги Ф.Брукса «Міфічний людино-місяць» [1], в якій автор на досвіді проекту IBM System/360 – найдорожчому проекту США у 1960 після програми Аполон – продемонстрував недоліки проектного менеджменту, заснованого на принципах лінійних оцінок до визначення трудомісткості і тривалості програмних проектів. Однією з перших моделей оцінки економічних показників програмних проектів, яка враховувала специфіку галузі і була основана на аналізі емпіричних даних по результатам виконаних досліджень була модель SLIM, запропонована Л. Патнамом [2], що використав функцію Нордена-Рейлайха для визначення взаємозв'язку між обсягом (розміром) ПЗ, трудовитратами на реалізацію проекту та тривалістю його реалізації (1).

$$K = (S/C)^3/t^4_d, \qquad (1)$$

де K – загальні трудовитрати (люд.-рік);
S – розмір ПЗ (кількість рядків коду);
C – фактор середовища, залежить від стану технологій;
t_d – обмеження на термін виконання проекту (років).

Модель SLIM відрізнялася простотою і дозволяла отримувати з прийнятною точністю оцінки трудовитрат по проекту і, відповідно, визначати загальну вартість його виконання за умови, що відомі інші складові, необхідні для розрахунку. Водночас як отримання таких складових (обсягу ПЗ і тривалості виконання) є достатньо непростою задачею, так і точність отриманого розрахунку трудовитрат виявлялася незадовільною для проектів, які істотно відрізняються від певного «узагальненого» програмного проекту, що є основою моделі.

Невирішені раніше частини проблеми. Подальші роботи у напрямку побудови математичних моделей, які визначали залежність ключових економічних показників проекту, таких як його вартість реалізації (трудомісткість та тривалість виконання) від тих, які описували технічні характеристики проекту (обсяг робіт, складність та ін.) проводилися достатньо активно, кульмінацією подібних досліджень можна вважати достатньо популярні моделі COCOMO [3] та COCOMO II [4]. Водночас, незважаючи на те, що подібні моделі, як правило, давали вищу точність оцінок ніж SLIM, вони характеризувалися істотно вищою складністю використання, а також мали принципові недоліки останньої.

Метою статті є розгляд і формування рекомендацій по відношенню до використання підходів до оцінки економічних показників програмних проектів на основі функціональних точок та їх похідних.

Виклад основного матеріалу. Паралельно відбувався процес визначення взаємозв'язку між вимогами до програмного проекту і трудовитратами на його реалізацію, що знайшов реалізацію у групі моделей, першою з яких стала оцінка за допомогою функціональних точок (Function Points, FP) [5]. Запропонований у 1979 році співробітником IBM Аланом Альбрехтом (Allan Albrecht) підхід містив в своїй основі допущення, що визначення розміру ПЗ і обсягу робіт по його реалізації слід здійснювати на основі вимог, які ставляться до програмного проекту, виходячи із кількості та складності функцій, які реалізуються у програмному коді. Таким чином, кількість FP відображає користувацькі бізнес-функції, які реалізує саме ПЗ, а не абстрактний його розмір, що суттєво може відрізнятися навіть у ідентичних з точки зору корисності для кінцевого користувача.

Поряд із показником FP використовується і термін FPA (Function Point Analysis), який означає загальну назву для методики, що використовується для обчислення показника.

В процесі здійснення FPA виділяється дві фази: 1) фаза ідентифікації - виділення елементів функціональності на основі допустимих для аналізу вхідних даних (специфікацій, моделей, прототипів); 2) фаза вимірювання - аналіз і перетворення елементів функціональності в чисельні показники кількості FP.

На даний момент існує кілька різних методик обчислення показника FP. Найбільш авторитетними вважаються методика International Function Point Users Group (IFPUG, www.ifpug.org) і стандарт ISO 14143-1. В цілому методики дуже схожі, відрізняються в деяких деталях. Розглянемо для прикладу методику IFPUG FPA.

Обчислення показника FP здійснюється поетапно.

Перший етап. На цьому етапі здійснюється підрахунок кількості функцій по категоріям.

Розглядається п'ять основних категорій у двох секціях, для яких підраховується кількість функцій. При описі даних використовується термін «файл», під яким розуміється пов'язана група даних незалежно від типу, подання та фізичної реалізації.

Секція даних (Data Section):

- внутрішні логічні файли (Internal Logical Files, ILF) - являють собою файли, які підтримуються всередині програми. Це може бути інформація, введена користувачем і призначена для обробки ПЗ;
- зовнішні інтерфейсні файли (External Interface Files, EIF) - є файли, які підтримуються іншими додатками, однак до яких оцінюється додаток може мати доступ виключно в довідкових цілях (тобто зміна цих даних не передбачається).

Секція транзакцій (Transaction Section):

- зовнішні входи (External Inputs, EI) - являють собою транзакції, по-засобом яких здійснюється маніпуляція ILF. Як EI розглядаються будь-які операції по додаванню, зміни, або видалення даних ILF допомогою зовнішніх інтерфейсів, наприклад, інтерфейсу взаємодії з користувачем, в якому він може переглядати, вводити і змінювати дані;
- зовнішні виходи (External Outputs, EO) - являють собою транзакції, за допомогою яких дані передаються за межі програми. Як EO можуть розглядатися дані, які передаються іншим програмам, у тому числі можуть бути використані для модифікації ILF, яка інформація, яка демонструється користувачу;
- зовнішні запити (External Queries, EQ) - являють собою транзакції, які дозволяють виводити певні дані по запиту на підставі певних параметрів. Важливою вимогою до EQ є незмінність даних (ILF і EIF) в процесі здійснення запиту. Як правило, до EQ відносяться звіти, які будує додаток у відповідь на певні запити.

Другий етап. Розглядаються показники, за допомогою яких визначається складність типів даних і транзакцій.

- тип елемента «запис» (Record Element Type, RET) - підгрупа елементів даних всередині ILF. Наприклад, якщо ILF «Інформація про клієнта» може містити кілька адрес і номерів телефонів клієнта, то це буде означати наявність двох RET: адреса і номер телефону;
- тип елемента «дані» (Data Element Type, DET) - являє собою унікальне користувальницьке поле в ILF або EIF. Як DET можуть розглядатися природні первинні ключі на ERD-діаграмі;
- посилання на тип файлу (File Type Reference, FTR) - являє собою файл, який бере участь в транзакції (для ILF передбачена будь-яка операція, EIF - тільки читання).

Для оцінки складності ILF і EIF використовуються показники RET і DET. Для цього використовується спеціальна матриця, яка дозволяє зробити оцінку складності за трибальною шкалою (низька, середня, висока). Матриця оцінки складності ILF / EIF представлена в таблиці 1.

Для оцінки складності EI, EO і EQ використовуються показники DET і FTR. Матриця оцінки складності EI представлена в таблиці 2, матриця оцінки складності EO / EQ представлена в таблиці 3.

Таблиця 1. Матриця оцінки складності ILF/EIF

Число RET	**Число DET**		
	1 – 19	**20 – 50**	**51 и более**
1	Низька	Низька	Середня
2 – 5	Низька	Середня	Висока
6 та більше	Середня	Висока	Висока

Таблиця 2. Матриця оцінки складності EI

Число FTR	**Число DET**		
	1 – 4	**5 – 15**	**16 и более**
0 – 1	Низька	Низька	Середня
2	Низька	Середня	Висока
3 та більше	Середня	Висока	Висока

Таблиця 3. Матриця оцінки складності EO/EQ

Число FTR	**Число DET**		
	1 – 5	**6 – 19**	**20 и более**
0 – 1	Низька	Низька	Середня
2 – 3	Низька	Середня	Висока
4 та більше	Середня	Висока	Висока

Третій етап. На цьому етапі виробляється обчислення числа незкорегованих функціональних точок (Unadjusted FP) на підставі значень кількості функцій по категоріяя і оцінки складності, отриманих на попередніх етапах.

Обчислення проводиться відповідно до таблиць, в яких вказані коефіцієнти для перекладу значень ILF, EIF, EI, EO, EQ в кількість FP. Для кожної категорії функцій розглядається кількість функцій, відповідне одному з трьох рівнів складностей і множиться на значення коефіцієнта (таблиця 4). Підсумкова сумарна величина являє собою значення Unadj. FP.

Таблиця 4. Таблиця коефіцієнтів для перекладу ILF, EIF, EI, EO, EQ в число Unadjusted FP

Складність	**ILF**	**EIF**	**EI**	**EO**	**EQ**
Низька	7	5	3	4	3
Середня	10	7	4	5	4
Висока	15	10	6	7	6

Четвертий етап. На цьому етапі проводиться обчислення значення коригуючого чинника (Value Adjustment Factor, VAF), який використовується для коригування значення Unadj. FP. VAF може коригувати значення Unadj. FP в межах ± 35%.

У процесі обчислення VAF розглядатиметься 14 факторів середовища (General System Characteristic, GSC):

1. Канали передачі даних (Data communications) - являють собою дані або керуючу інформацію, що передаються каналами передачі даних.
2. Розподілена обробка даних (Distributed data processing) - передбачає наявність даних, які використовуються в системі, що розробляється, але зберігаються і обробляються поза нею.
3. Продуктивність (Performance) - є для користувача вимоги до продуктивності системи.
4. Інтенсивність використання платформи (Heavily used configuration) - передбачає оцінку інтенсивності використання апаратної платформи, на якій буде виконуватися система.
5. Частота транзакцій (Transaction rate) - передбачає оцінку частоти транзакцій системи (мережевий трафік, частота оновлення показників і пр.)
6. Введення даних онлайн (On-Line data entry) - передбачає оцінку в процентної співвідношенні кількості даних, які будуть вводитися в режимі онлайн.
7. Ефективність на рівні кінцевого користувача (End-user efficiency) - передбачає оцінку уваги, яка приділяється ефективності на рівні кінцевого користувача системи.
8. Оновлення в режимі онлайн (On-Line update) - передбачає оцінку числа ILF, які оновлюються в режимі онлайн.
9. Складна обробка (Complex processing) - оцінка складності обчислень, здійснюваних системою (складні алгоритми, керуюча логіка, обробка з безліччю транзакцій).
10. Повторне використання (Reusability) - висуваються вимоги до повторного (багатоцільовому) використанню розроблюваних рішень?
11. Простота установки (Installation ease) - висуваються додаткові вимоги для простої установки (перетворення) системи?
12. Простота використання (Operational ease) - висуваються додаткові вимоги для простоти використання (мінімального втручання з боку операторів) системи?
13. Використання на декількох вузлах (Multiple sites) - оцінка того, чи передбачається використання системи на різних вузлах (враховуються розбіжності в бізнес-функції, які можуть існувати на різних вузлах).

14. Полегшення змін (Facilitate change) - оцінка вимог до системи, які висуваються з метою полегшення подальших змін (підтримка модульності, розширюваності і т.п.)

Для кожного з 14-ти факторів GSC надається оцінка по шести-бальною шкалою (від 0 до 5) і формується підсумкове значення (Total Degree of Influence, TDI) як сума всіх оцінок.

Далі, на підставі TDI обчислюється VAF за такою формулою (2).

$$VAF = 0{,}65 + TDI/100. \quad (2)$$

Оскільки VAF може коригувати значення Unadj. FP в межах ± 35%, то виходить за межі цього діапазону значення VAF встановлюється рівним найближчій межі діапазону (3).

$$VAF = [0{,}65 .. 1{,}35]. \quad (3)$$

П'ятий етап. На цьому етапі обчислюється підсумкове значення FP як добуток VAF і Unadj. FP, підсумкове значення прийнято округляти до цілого числа (4).

$$FP = VAF \times Unadj.\ FP. \quad (4)$$

Далі отримане значення FP може бути перетворено в одиниці виміру обсягу ПЗ (кількість рядків коду, SLOC) або може бути зроблена оцінка продуктивності по виконанню кількості FP в день (Performance factor), на підставі якого можна отримати оцінку трудомісткості проекту в людино-днях.

Поява показника FP як принципово нового способу оцінки обсягу робіт з програмним проектам викликало цілу хвилю різних методів і модифікацій, заснованих на FP, існують десятки варіацій оригінальної методики.

Найбільш відомі похідні від FP:

- Точки властивостей (Feature Point) - запропонований в 1986 р. компанією Software Productivity Research метод, придатний для застосування в тому випадку, коли сформульовані вимоги не відображають істинної складності реалізації (що особливо характерно для системного ПЗ, критично важливих програмних комплексів тощо) , оскільки оригінальний метод FP в такому разі себе не виправдовує. Даний метод близький оригінальному FP, однак передбачає коригування одержуваної підсумкової оцінки з урахуванням алгоритмічної складності розроблюваного рішення.

- Метод Mark II - запропонований в кінці 1980-х рр. Ч. Саймонс. Основна відмінність даного методу від підходу IFPUG полягає в підходах обчислення розміру ПЗ, які більш пристосовані для великих програмних систем і дозволяють зробити метод більш придатним для оцінки складних систем. Зокрема, Mark II дозволяє добитися одного і того ж результату як при оцінці системи в цілому, так і при підсумовуванні оцінок, отриманих для складових її підсистем.

- Тривимірні функціональні точки (3D Function Point) - запропоновані софтверним підрозділом корпорації Boeing в 1991 р. В основу цього методу покладено ідею про те, що складність завдання у програмному середовищі можна уявити в трьох вимірах - дані (кількість вводів / висновків), функції (складність обчислень) і контроль (керуюча логіка). Цей метод дозволяє трохи спростити оригінальні методики обчислення FP і виходить за рамки виключно програмних проектів, оскільки дозволяє оцінювати трудомісткість вирішення завдань в різних сферах - ділової, наукової і т. д.

- Об'єктні точки (Object-Oriented Function Point) - являє собою адаптований варіант FP, що оперує термінами об'єктео-орієнтованої-технології і придатний для використання відповідно до сучасних ОО-методологіями розробки ПЗ.

Висновки. Таким чином, підхід до оцінки економічних показників програмних проектів на основі FPA оснований на вивченні вимог до програмного проекту, що, на нашу думку, є більш перспективним підходом, ніж використання підходів, які передбачають побудову регресійних математичних моделей, що встановлюють зв'язок між деякими достатньо умовними показниками обсягу коду і показниками трудомісткості, вартості та тривалості виконання проекту. Таке твердження слід пояснити наступним чином: реалізація програмного проекту характеризується високим рівнем інноваційності і невизначеності, що, в свою чергу, дозволяє вирішувати одну і ту ж задачу різними шляхами, які вимагають як різного обсягу робіт, так і, відповідно, різної трудомісткості та кінцевої вартості проекту в цілому. Подібна розбіжність водночас становить високий ризик з точки зору ефективного управління програмним проектом. Якщо оцінка виявляється завищеною, то в кінцевому випадку проект може вийти за рамки об'єктивно потрібних часових і фінансових ресурсів, щоб зрівнятися з нею, у противному випадку, якщо оцінка занижена, порушення ресурсних обмежень є неминучим і проект опиниться у критичній ситуації. Використання ж підходів, основаних на FPA містить певний «стимулюючий» фактор – оскільки оцінка обсягу робіть по проекту базується виключно на вимогах до проекту, то ризик переоцінки трудовитрат зменшується. У тих випадках, де реалізація проекту була здійснена неоптимальним шляхом, існують підстави для того, щоб вимагати дотримання початкової оцінки витрат по ньому, що корисно з точки зору захисту інтересів замовників. Також перевагою основаних на FPA підходів є достатньо висока гнучкість, що дозволяє адаптувати їх до конкретної предметної галузі, у томи числі й такої, яка не належить напряму до розробки ПЗ – прикладом подібного застосування є тривимірні функціональні точки, запропоновані корпорацією Boeing.

Література

1. The Mythical Man-Month [Текст] / Brooks F. // Addison-Wesley Publishing Company Reading, Massachusetts 1975. – 195 P.
2. Колдовський В.В. Визначення економічних параметрів інноваційних процесів на етапах життєвого циклу програмного забезпечення [Текст] // Вісник Української академії банківської справи. – 2005. – № 1. – С. 105-113.
3. COCOMO II Model Definition Manual / Boehm B., Abts C., Clark B., and others [Текст] // Los Angeles (USA): University of Southern California. – 1999. – 37 p.
4. Jensen R. A Comparison of the Jensen and COCOMO Schedule and Cost Estimation Models [Текст] // Proc. International Society of Parametric Analysis. – 1984. – P. 96-106.
5. ПО: оценка результата [Текст] / В.В. Колдовский // Компьютерное обозрение, 2006. – № 34. – С. 58–61

Стаття надійшла до редакції 22.06.2011 р.