УДК 004.891.3: 004.3

Р.А. МАЛЯРЧУК, Т.О. ГОВОРУЩЕНКО
Хмельницький національний університет

АНАЛІЗ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ МЕТОДОЛОГІЇ РОЗРОБЛЕННЯ ДЛЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ РІЗНИХ ТИПІВ

У статті виконано аналіз використовуваності сучасних методологій розроблення програмного забезпечення. Аналіз показав, що при наявній великій кількості методологій панує повний суб'єктивізм їх вибору розробниками програмного забезпечення. У роботі доведено актуальність задачі оцінювання ефективності методології розроблення для програмного забезпечення різних типів. Оцінювання ефективності методологій розроблення для програмного забезпечення різних типів може підвищити якість програмного забезпечення, а також знизити витрати на його розроблення.

Ключові слова: програмне забезпечення (ПЗ), програмний проект, методологія розроблення ПЗ.

R.A. MALYARCHUK, T.O. HOVORUSCHENKO
Khmelnitsky National University

ANALYSIS OF EFFICIENCY OF USE OF DEVELOPMENT METHODOLOGIES FOR THE DIFFERENT TYPES OF SOFTWARE

Abstract - In this paper the analysis of the subject area sources for the purpose of using of modern software development methodologies and of determining the efficiency of use of software development methodologies for different types of software was performed.

The analysis showed that the large number of methodologies exists, and the subjectivity of its choice by software developers reigns. During the evaluation of software development methodologies the developers accentuate the following problems: 1) lack of the unified knowledge base on the software development methodologies; 2) lack of the unified standard for the evaluation of software development methodologies; 3) lack of the unified criteria for evaluation of software development methodologies for different types of software. In this paper the actuality of the task of evaluation of the efficiency of the software development methodologies for different types of software was proven.

Evaluation of the effeciency of software development methodologies for different types of software can improve the software quality and reduce the cost of its development.

Keywords: software, software project, software development methodology (SDM).

Вступ

З аналізу сучасного стану галузі створення програмного забезпечення (ПЗ) випливає, що наразі програмістам доводиться використовувати комплексні засоби розроблення, які складаються з великої кількості важливих та необхідних компонентів для створення повнофункціональних програмних додатків. Одним з таких засобів є методологія розроблення ПЗ.

Методологія розроблення ПЗ – це набір методів та критеріїв оцінки, які використовуються для постановки задачі, планування, контролю і для досягнення поставленої мети [1]. Процес розроблення описується моделлю, яка визначає послідовність найбільш загальних етапів та одержуваних результатів. Методологія розроблення ПЗ – це система принципів, а також сукупність ідей, понять, методів, способів та засобів, які визначають стиль розроблення ПЗ. Методології представляють собою ядро теорії керування розробленням ПЗ [2].

Сьогодні найбільш відомими та використовуваними є наступні методології: Agile (SCRUM, XP, Dynamic System Development Method (DSDM), Kanban та інші), Microsoft Solutions Framework (MSF), Rational Unified Process (RUP), водоспадні, ітеративні та інші.

З проведеного аналізу відомих методологій розроблення [3] очевидним є існування значної кількості методологій за відсутності універсальної методології, яка підходила б для будь-якого програмного проекту, а також за відсутності чітких стандартизованих критеріїв оцінювання та вибору методології розроблення ПЗ. Тому при усій множині існуючих методологій розроблення, кількість програмних проектів з недоліками та неуспішних програмних проектів залишається досить високою (61% у 2012 році [4]), а процес розроблення ПЗ залишається недетермінованим, і результат його завжди невідомий.

Такий стан справ можна пояснити наступною причиною в контексті методологій – їх неефективністю для ПЗ певного типу. Різні типи програмних проектів вимагають різних підходів, оскільки кожна категорія проектів має різні пріоритети і цілі. Адже, зрозуміло, що неможливо використовувати один і той же підхід для розроблення Web-сторінки організації, бізнес-додатку та вбудованого програмного забезпечення для космічної ракети.

Отже, задача аналізу ефективності використання методологій розроблення для ПЗ різних типів є актуальною на етапі проектування, оскільки від ефективності використаної методології залежить майбутня успішність програмного проекту. Оцінка ефективності методологій розроблення ПЗ допоможе розробнику оцінити, чи варто йому братись за той чи інший програмний проект за наявної методології (і не витрачати часу та коштів на неуспішні програмні проекти), а замовнику дозволить зробити обгрунтований висновок про придатність або непридатність методології, а відтак і софтверної компанії, для розроблення ПЗ певного типу.

Тоді метою дослідження є аналіз джерел предметної галузі на предмет визначення ефективності

використання методології розроблення для програмного забезпечення різних типів.

Аналіз ефективності використання методології розроблення для програмного забезпечення різних типів

Оцінювання ефективності методології розроблення взагалі вимагає врахування багатьох різноманітних умов та факторів [5]: масштаб проекту; критичність проекту; кількість та розподіл повноважень учасників проекту; ступінь новизни проекту; очікувана тривалість проекту; вимоги замовника; специфіка та складність проекту. Галузеві публікації дають можливість зробити наступні висновки щодо використання методологій проектування ПЗ (табл.1).

Таблиця 1

Використання методологій розроблення для програмного забезпечення різних типів

Методологія розроблення ПЗ	**Розмір програмного проекту**	**Тип програмного проекту**	**Мета та опис програмного проекту**
1	2	3	4
Водоспадна (Waterfall)	Для великих, розрахованих на тривалий термін, проектів [5]. Для проектів з середніми та великими проектними командами (20+ чоловік) [8]. Середній (1-3 млн доларів США) або малий (від 250000 до 1 млн доларів США), програмний проект високої або середньої складності з середнім або низьким рівнем ризиків [10]. Іноді навіть дуже малі (до 250000 доларів США) програмні проекти низької складності з низьким рівнем ризиків [10]	Комерційні коробкові програмні продукти (COTS), сховища даних, проекти інтеграції/заміни, розроблення нових додатків – процедурні технології, аутсорсинг, одинчині проекти, проекти критичного застосування [8]. Проекти IT-B, IT-E [10]	Часто звертаються до менеджерів з метою пошуку простого для розуміння і управління підходу [8]
Ітеративна (Iterative)	Для великих, розрахованих на тривалий термін, проектів [5]. Дуже великі (10 млн доларів США або більше) і великі (3-10 млн доларів США) програмні проекти високої складності з високим рівнем ризиків [10]. Також середні (1-3 млн доларів США) або малі (від 250000 до 1 млн доларів США) програмні проекти середньої або низької складності з середнім або низьким рівнем ризиків [10]	Комерційні коробкові програмні продукти (COTS), сховища даних, проекти інтеграції/заміни, розроблення нових додатків – компонентні / об’єктні технології, аутсорсинг, одинчині проекти, проекти критичного застосування [8]. Проекти IT-A, IT-C [10]	Добре працюють в середовищах, які поки ще в змозі миритися з "ризикованими" практиками, пов’язаними з ітеративним розробленням [8]
Agile Data, Agile Model Driven Development	Для менш масштабних проектів, розрахованих на оперативну реалізацію [5]. Середні (1-3 млн доларів США) або невеликі (250 000 – 1 млн доларів США) програмні проекти середньої або низької складності з середнім або низьким рівнем ризиків [10]. Малі (від 250000 до 1 млн доларів США) програмні проекти низької складності з середнім або низьким рівнем ризиків [10]. Також малі (до 250 000) програмні проекти низької складності з низьким рівнем ризиків [10]. Для проектів з малими, спільно розташованими (суміщеними) проектними командами, які мають чітке уявлення про бізнес-завдання програмного проекту [10]	Сховища даних, проекти інтеграції/заміни, розроблення нових додатків – компонентні / об’єктні технології [8]. Проекти IT-C, IT-D, IT-E [10]	Для розроблення програмного забезпечення, орієнтованого на людину, що дозволяє ефективно реагувати на зміни і призводить до створення робочих систем, які відповідають потребам зацікавлених сторін [8]

Продовження табл. 1

1	2	3	4
Rational Unified Process (RUP)	Для масштабних та тривалих проектів [6]	Комерційні коробкові програмні продукти (COTS), сховища даних, проекти інтеграції/заміни, розроблення нових додатків – процедурні технології, аутсорсинг, одинчині проекти, проекти критичного застосування [8]	
Microsoft Solutions Framework (MSF)	Для великих проектів, які вимагають дотримання балансу між ресурсами, часом розроблення та можливостями [6]		
SCRUM	Для невеликих та середніх проектів, особливо якщо під час процесу розроблення очікується внесення численних змін [7]. Використовується для проектів будь-якого розміру [8]		
XP	Для невеликих проектів, в яких не виникає необхідності у створенні детальної документації та регламентації всіх кроків розроблення [5]. Для проектів з малими, спільно розташованими (суміщеними) проектними командами (4-10 чоловік) [8]	Проекти інтеграції/заміни, розроблення нових додатків – компонентні / об'єктні технології, проекти критичного застосування [8]	Вимоги не є чітко визначеними. Хороші стосунки (потенційно) існують із зацікавленими сторонами проекту [8]
Об'єктно-орієнтована	Для проектів з використанням об'єктно-орієнтованих технологій [8]. Для проектів з середніми та великими проектними командами (10+ чоловік) [8]	Розроблення нових додатків – компонентні / об'єктні технології [8]	
Орієнтована на дані	Для додатків, орієнтованих на дані [8]		
Dynamic System Development Method (DSDM)		Проекти інтеграції/заміни, розроблення нових додатків – компонентні / об'єктні технології [8]	Розроблення інтерфейсу користувача інтенсивної системи; складні бізнес-додатки [8]

У таблиці 1 маються на увазі наступні характеристики для використаних типів і розмірів програмних проектів. *Програмний проект низької складності*: проектна команда мала, розташована географічно спільно (суміщена) і значною мірою однорідна. Для проектів такого роду добре підходять команди з можливістю вербальної комунікації, коли члени проектної команди мають можливість часто спілкуватись один з одним [10]. *Програмний проект середньої складності*: є суміщені (спільно розташовані) групи всередині проектної команди, а інші члени команди знаходяться поза цією територією, або загалом проектна команда занадто велика, щоб зібратися разом в один час. Наявність таких факторів завадить команді ефективного спілкуватись у будь-який час. Проекти середньої комунікаційної складності потребують більш формальної, письмової комунікації, ніж програмні проекти низької складності, але в них ще присутнє словесне неформальне спілкування [10]. *Програмний проект високої складності*: аутсорсингове / офшорне розроблення і великі, територіально розподілені, багатокультурні проектні команди вимагають значно більшої кількості каналів формальної комунікації, формальних, письмових артефактів та їх ретельного вивчення. Ці проекти за визначенням триватимуть довше і матимуть високий ризик наявності незрозумілих вимог. Учасникам таких проектних команд рекомендується часто спілкуватись один з одним, щоб зменшити ризик використання лише явних каналів комунікації [10].

Програмні проекти IT-A – це програмні проекти, які статистично мають найбільші шанси стати провальними або проблемними (за функційними можливостями, бюджетом, часом виконання). Ці проекти будуть повільно виконуватись, тому вимоги повинні бути добре задокументовані і підписані перед

передачею їх до команди розробників. Управління та контроль повинні бути зосереджені на безперервній адаптації проекту до нових та змінюваних бізнес-потреб [10]. *Програмні проекти IT-B* – це програмні проекти, для яких передбачається короткий термін виконання (шість місяців або менше) і наперед відомі вимоги [10]. *Програмні проекти IT-C* – це програмні проекти, для успішної реалізації яких потрібно враховувати той факт, що бізнес-потреби часто змінюються [10]. *Програмні проекти IT-D* – це програмні проекти, вимоги до яких з'ясовуватимуться по мірі побудови програмної системи, з швидким зворотнім зв'язком і високим ступенем залученості кінцевого користувача. *Програмні проекти IT-E* – це малі програмні проекти, для яких збір вимог, їх підтвердження та сумарні зміни проводяться в дуже короткі терміни [10].

Рисунок 1 описує і порівнює провідні методології розроблення ПЗ [8]. В центрі уваги даного рисунку – високорівневі процеси розроблення ПЗ, а не докладні методи, а також містить поради, коли варто застосовувати ту чи іншу методологію, якщо є можливість вибору.

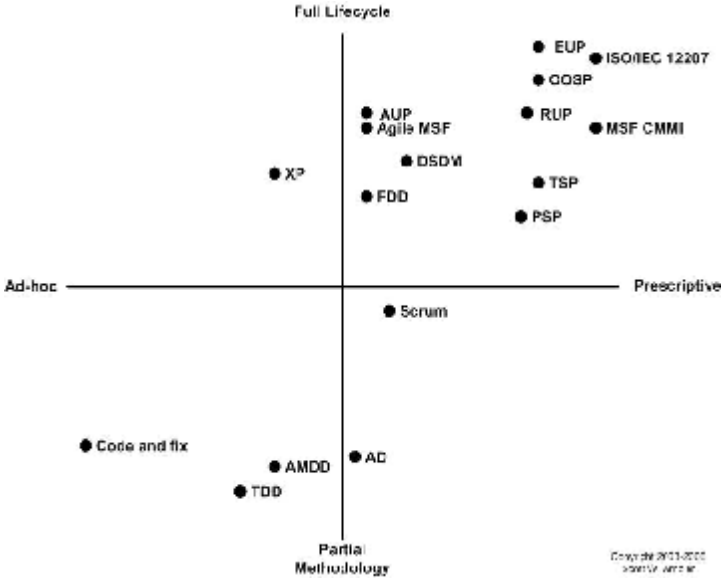

Рис. 1. Порівняння провідних методологій розроблення ПЗ [8]

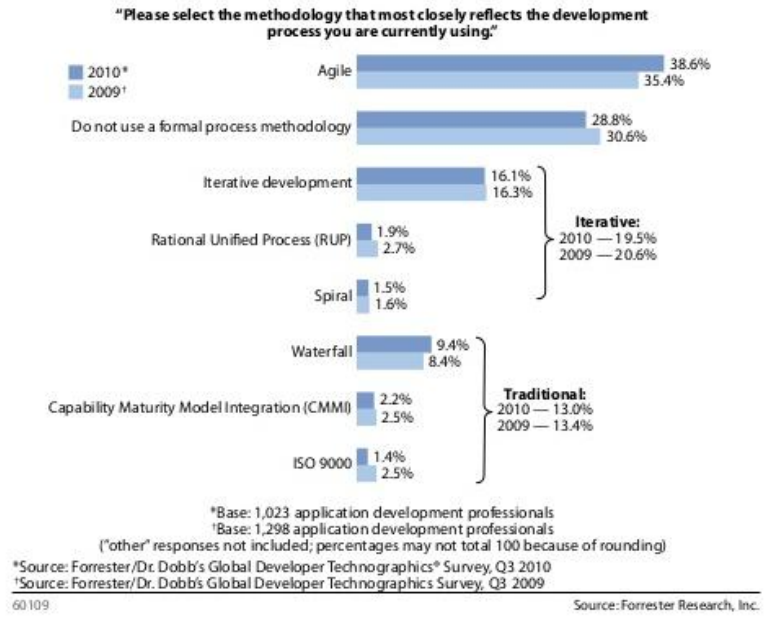

Рис. 2. Статистика використання методологій розроблення ПЗ [11]

Діаграма порівняння успішності програмних проектів, реалізованих за водоспадною та AGILE-методологіями, наведена у [9], показує, що AGILE-проекти втричі успішніші за водоспадні проекти, але з неї незрозуміло, яку саме з AGILE-методологій було використано для успішних програмних проектів, а також проекти якого типу розглядались у цьому дослідженні.

Статистика використання методологій розроблення ПЗ наведена на рисунку 2 [11].

З рисунку 2 очевидно, що у 2010 році 30,6% софтверних компаній взагалі не використовують методологій розроблення ПЗ; 38,6% софтверних компаній використовують AGILE-методології; 19,5% компаній використовують ітеративні методології розроблення; 13% софтверних компаній досі використовують традиційні (в т.ч. водоспадну) методології розроблення. Крім цього, аналіз рисунку 2 дає можливість зробити наступні висновки: 1) зростає кількість компаній, які використовують методологію розроблення ПЗ; 2) зростає кількість компаній, які використовують AGILE-методології; 3) зменшується кількість компаній, які використовують ітеративні та традиційні методології, але в той же час серед традиційних методологій зростає використовуваність саме водоспадної методології.

Наразі відомі софтверні компанії використовують наступні методології: 1) AGILE-методології використовуються компанією Google; 2) MSF-методологія – компанією Microsoft; 3) традиційні методології (зокрема, CMM) – компаніями Boeing, Northrop-Grumman, Lockheed-Martin. Отже, компанії абсолютно суб'єктивно обирають методологію розроблення – немає ані критеріїв вибору методологій, ані систем підтримки прийняття рішень щодо вибору методологій, ані рекомендацій у стандартах.

Наразі є чимало джерел, які пропонують використовувати різні методології для ПЗ різних типів, але відсутня статистика про те, яка кількість проектів певного типу, розроблених за тією чи іншою методологією, є успішними. Відтак галузеві публікації не дають можливості одразу зробити висновок про ефективність використання та підходящість конкретної методології розроблення для ПЗ певного типу.

Висновки

При оцінюванні методологій розроблення ПЗ розробники виділяють наступні проблеми: 1) відсутність єдиної бази знань з методології; 2) відсутність єдиного стандарту для оцінювання методологій; 3) відсутність єдиних критеріїв, за якими оцінюються методології для ПЗ різних типів. Оцінювання ефективності методологій розроблення для ПЗ різних типів може підвищити якість ПЗ, а також знизити витрати на його розроблення.

Зрозуміло, що кожна софтверна компанія використовує вже придбану та впроваджену методологію розроблення, для роботи з якою є достатня кількість фахівців. Дослідницька задача полягає у визначенні, на скільки вдало вдасться спроектувати та розробити ПЗ певного типу, використовуючи певну методологію розроблення, запроваджену в конкретній софтверній компанії. Отже, іншими словами, актуальною науковою задачею є визначення ефективності конкретної методології розроблення для ПЗ певного типу.

Перспективою для подальших досліджень авторів є наступні задачі:

1) провести аналіз софтверних компаній України та світу за наступними критеріями: які використовують методології; яке ПЗ розробляють; які значення (ступені виконання) основних характеристик ПЗ забезпечують; типи та кількість успішних і неуспішних (через непідходящість методології) програмних проектів;

2) обгрунтовано обрати критерій (адитивний або мультиплікативний) для визначення ефективності;

3) визначити значення основних характеристик ПЗ з точки зору можливості їх реалізації з використанням певної методології на основі аналізу п.1;

4) визначити роль та пріоритетність характеристик для ПЗ кожного типу з врахуванням цільового призначення та типу програмного проекту шляхом визначення вагових коефіцієнтів для ПЗ різних типів та різного призначення. Саме вагові коефіцієнти повинні не давати можливості компенсувати низьке значення зреалізованості важливої характеристики високим значенням зреалізованості іншої, менш важливої, характеристики. Отже, значення вагових коефіцієнтів повинні бути підібрані таким чином, щоб при неможливості (або низькій можливості) реалізації важливої характеристики конкретного ПЗ певною методологією, ефективність методології для розроблення цього ПЗ дуже стрімко зменшувалась;

5) розробити інформаційну технологію оцінювання ефективності методологій розроблення для ПЗ певного типу.

Література

1. Хаф Л. Методологии разработки программного обеспечения [Електронний ресурс] / Л. Хаф. – Режим доступу: http://compress.ru/article.aspx?id=11321

2. Методологии разработки программного обеспечения [Електронний ресурс] – Режим доступу: http://habrahabr.ru/sandbox/43802/

3. Т.О. Говорущенко. Аналіз процесу вибору технології проектування, методології та середовища розроблення програмного забезпечення / Т.О.Говорущенко, Р.А.Малярчук // Вісник Хмельницького національного університету. Технічні науки. – 2014. – №6. – С.186-196.

4. CHAOS Manifesto: Think big, act small – The Standish Group International: CHAOS Knowledge Center, 2013 – 52 p.

5. Разработка программного обеспечения: Методологии разработки программного обеспечения [Електронний ресурс]. – Режим доступу: http://bms-soft.com.ua/ru/services/razrabotka-programmnogo-obespecheniya/metodologii-razrabotki-po

6. R.C.Martin. Agile Software Development, Principles, Patterns, and Practices / R.C.Martin, J.W.Newkirk, R.S.Koss – Prentice Hall: Upper Saddle River, 2002. – 752 p.

7. Обзор методологии разработки программного обеспечения SCRUM [Електронний ресурс]. – Режим доступу: http://www.dpgrup.ru/methodology-scrum.htm

8. Choose the Right Software Method for the Job [Electronic resource]. – Access mode: http://www.agiledata.org/essays/differentStrategies.html

9. Agile Succeeds Three Times More Often Than Waterfall [Electronic resource]. – Access mode: http://www.mountaingoatsoftware.com/blog/agile-succeeds-three-times-more-often-than-waterfall

10. Project Classification // Requirements Methodology Park – Software Education Group, 2008 – 14 p.

11. D.West. Water-Scrum-Fall is the reality of AGILE for most organizations today [Electronic resource] – Access mode: http://www.slideshare.net/harsoft/water-scrumfall-isrealityofagileformost

Рецензія/Peer review : 21.9.2015 р. Надрукована/Printed :2.11.2015 р.
Рецензент: д.т.н., Поморова О.В.