УДК 519.7

МОДЕЛЮВАННЯ ЧАСУ ПРОТОТИПУВАННЯ В УПРАВЛІННІ ІТ ПРОЕКТАМИ

В. А. Ситник
Одеський національный політехничний університет

Анотація. *Запропонована методика аналогового моделювання часу створення ІТ проектів. Визначено критерії обгрунтування вибору ІТ проекту з відкритим кодом, створено життєвий цикл вибраного классу проектів. Визначено ризики відкриття коду. Сформульовано алгоритм застосування теорії подібності для визначення часу прототипування проекту. Методика теоретично обгрунтована, та дозволяє на ранніх етапах прогнозувати час та вартість майбутнього проекту. Методика перевірена на завершених проектах та показує високий відсоток достовірності. Може бути застосована при створенні нових ІТ проектів різних класів.*

Ключові слова: *системиний аналіз, математичне моделювання, теорія подібності, статистичний аналіз, життєвий цикл, класифікація, критеріальне рівняння, аналіз розмірностей.*

Вступ

Робота присвячена розробці методики управління часом прототипування програмного забезпечення, яка дозволяє на ранніх стадіях циклу життя встановлювати можливість успішного завершення проекту в рамках певного класу проектів, має порівняльну з іншими методами ефективність, і при цьому усуває існуючі недоліки аналізу успішного завершення ІТ проектів.

Об'єктом дослідження є процеси управління програмними проектами.

Предметом дослідження є моделі життєвого циклу і системи управління програмними проектами.

Метою дослідження є розробка методики підвищення ефективності управління програмними проектами.

Для досягнення поставленої мети в роботі розроблена методика оцінки часу прототипування проекту, проведені емпіричні дослідження методики для підтвердження її ефективності та можливості застосування.

Проведені дослідження базуються на методах системного аналізу, теорії подібності, математичної статистики, математичного моделювання, використання комп'ютерних систем і сучасних програмних комплексів.

Основним науковим результатом роботи є методика отримання критеріального рівняння, яке дозволяє для різних класів програмних проектів визначати на початкових етапах циклу життя можливість успішного завершення проекту (час прототипування), підвищуючи, таким чином, ефективність управління цими проектами.

Практична цінність результатів роботи і їх практичне використання визначається застосуванням розробленої методики як основи знань в управлінні програмними проектами.

1. Управління прототипуванням в ІТ проектах. Аналіз досліджень та перспективи

Аналіз останніх досліджень в галузі управління програмними проектами показав, що ефективне управління ними є основним способом їх успішного завершення в рамках обмежень часу, вартості та якості. За даними дослідницької групи Standish Group [1], з 50000 відстежених програмних проектів через неефективне управління 18% завершилися невдало, 53% потребували додаткових витрат часу і тільки 29% успішно завершилися. За іншими даними ([2]), більше 50% проектів виявляться невдалими. За результатами того ж дослідження [1], в середньому бюджет проектів перевищується на 189%, а витрачений час на 222% перевищує оцінене. При цьому реалізується в середньому всього 69% заявленої в специфікації функціональності.

Описана вище ситуація пояснюється тим, що "... розробка програм є концептуально складним заняттям" [3], труднощі створення програмних систем виникають через їх "нескорочувану сутність", до властивостей якої відносяться складність і мінливість проекту [4].

Однак, за останній час з'явилося багато нових моделей управління програмними проектами, які, не дивлячись на неможливість революційного прориву в дисципліні, дозволяють підвищити ефективність управління. Одним з таких способів стала модель розробки програмного забезпечення з відкритим кодом. Програмні проекти з відкритим кодом відомі на практиці як успішні проекти, що завершуються вчасно, з мінімальними витратами і в яких виготовляється

якісний продукт. За роботою [5] приблизно 27% всіх фірм–розробників програмного забезпечення виконують такі проекти. Крім того, існує більше 170 000 некомерційних відкритих проектів в яких беруть участь більше мільйона чоловік.

У зв'язку з такою популярністю програмних проектів з відкритим кодом і їх значенням для індустрії розробки програмного забезпечення, виникає необхідність володіти знаннями по їх управлінню. Однак відзначається ([6]), що останні дослідження були спрямовані лише на опис моделі і на дослідження конкретних випадків її використання. Внаслідок цього, питання вивчення життєвого циклу і системи управління, які забезпечують основу знань для ефективного управління проектами, залишаються відкритими. Невивченість існуючих методів виконання процесів управління, особливо управління часом і вартістю, також заважає ефективному управлінню та успішному завершенню проектів в рамках заданих обмежень часу, вартості та якості.

Все це призводить до необхідності вивчення способів підвищення ефективності управління програмними проектами з відкритим кодом шляхом розробки нових моделей оцінки часу, нової методики управління життєвим циклом проекту.

З огляду на дослідження Evans Data Corporation [7], які показують, що понад 35% розробників по всьому світу задіяні в проектах з відкритим кодом і ця цифра зростає на 2-5% на рік, а також з огляду на величезну кількість і успішність як комерційних так і некомерційних проектів, потрібно визнати важливість відкритої моделі розробки програмного забезпечення та необхідність досліджень як самої моделі так і способів підвищення ефективності управління програмними проектами, що виконуються в цій моделі.

Однак, необхідно звернути увагу на порівняно невелику частку комерційних проектів (15%), що пояснюється багатьма аналітиками відсутністю теоретичного обґрунтування основних принципів виконання таких проектів. Як зазначається в [8], "найбільша проблема індустрії відкритих початкових кодів, що перешкоджає її подальшому розвитку, сьогодні полягає в тому, що навіть прихильники відкритого коду в більшості своїй ще не розуміють, що відкрита модель є дійсно самостійною. Ніхто з прихильників відкритого коду не розуміє, який інструмент виявився у них в руках. А з цього вже витікає фізична неможливість використовувати його "ефективно", що підтверджується і численними побоюваннями, висловлюваними менеджерами і керівництвом компаній. Так, наприклад, в Sim Microsystems, незважаючи на позитивний досвід відкриття вихідного коду OpenOffice і 6-ти річний досвід його подальшої розробки у відкритій моделі, більше 2-х років приймали рішення про відкриття вихідного коду Java і переведення проекту Java на відкриту модель [9-11].

Виходячи з вищезазначеного, очевидно, що розробка методики управління створенням програмного забезпечення з відкритим вихідним кодом є важливою і своєчасною. Розв'язанням такого завдання буде формальний опис моделі відкритих початкових кодів, складових її фаз і дій. Іншими словами, необхідна розробки моделі життєвого циклу і методика системи управління. Все це надасть необхідну основу знання для організацій, які планують, або розробляють програмні проекти з відкритим кодом.

2. Методика управління часом прототипування в IT проектах

2.1. Розробка моделі життєвого циклу IT проектів

Опис моделі життєвого циклу і її складових необхідний для успішного управління проектами складається з:

- схематичного опису для визначення фаз і їх послідовності;
- діаграм розподілу робіт за фазами проекту;
- таблиць ідентифікації завдань і дій, які виконуються в життєвому циклі, що потрібні для подальшого розподілу ресурсів проекту.

Детальний формальний опис в такому вигляді визначає методологію управління проектами, що повинна виконувати всі процеси життєвого циклу відповідно до стандартів ISO / TEC 12207 [12] і ДСТУ 3918–1999 [13].

Модель життєвого циклу програмних проектів визначається сукупністю загальних і специфічних ознак.

Загальні ознаки моделі визначаються як:

- множина фаз виконання проекту;
- зміст технічних робіт по кожній фазі;
- розподіл інженерних і технічних ресурсів за фазами;
- послідовність виконання визначених фаз;
- перехідні процеси між фазами.

Специфічні ознаки, наприклад відкритої моделі, наступні:

- середовище виконання відкритого проекту значно ширше, ніж у будь–яких інших програмних проектів;
- нормальною практикою виконання проекту є постійне залучення сторонніх ресурсів, як матеріальних, так і трудових;

– множина виконавців проекту не є ні фіксованою, ні чітко визначеною.

Визначення фаз відкритої моделі і їх послідовності здійснюється шляхом аналізу ходу виконання вже завершених проектів.

Для аналізу були обрані найбільш характерні проекти, що представляють всі типи відкритих проектів:

– чисті: GNOME, KDE, Linux (ядро);
– конверсійні: Mozilla, Blender, OpenOffice;
– керовані: Qt, Eclipse, OpenSolaris.

Аналіз проводився за допомогою:

– хронологічних даних про проекти з електронної енциклопедії Wikipedia;
– інформації зі списків розсилки та веб–сайтів проектів;
– планів випуску та анонсів.

Результати аналізу показують, що за основу можна взяти хронологічні дані про управлінські дії в проекті KDE. Дані про інші проекти показують багато в чому аналогічну картину.

Інструкцією слугувала коротка характеристика дії, що класифікує його як етап (початок, або завершення) певної фази життєвого циклу. Анотація виконувалася з метою визначення фаз життєвого циклу.

Кожна фаза розробки, виходячи з даних системи управління версіями і планів випуску нових версій розподіляється на 3 основних етапи. Перший етап – виконання одночасного прототипування, кодування і документування з метою випуску нестабільної, але в повному обсязі функціональної "α" версії продукту. Наступний етап, після випуску "α" версії, включає в себе одночасне кодування, документування та модульне тестування з метою випуску більш стабільної і повнофункціональної "β" версії продукту. І, нарешті, в останній етап входить інтеграція і кваліфікаційне тестування продукту з метою випуску стабільної і повнофункціональної наступної версії продукту (з проміжними опційними випусками версій вигляду "release candidate"). За результатами проведеного вище аналізу і змістом проектної інформації, виконано схематичне відображення фаз життєвого циклу відкритої моделі і визначено зміст процесів, що виконуються в кожній з фаз.

Візуальний аналіз моделі показує, що вона має процеси, властиві каскадній і спіральній моделі. Як і в спіральній моделі, в неї включені процеси аналізу та управління ризиками і підтримки менеджменту. Передбачена розробка програмного продукту при використанні методу прототипування, або швидкої розробки додатків за допомогою застосування мов програмування і засобів розробки четвертого і вище покоління. Одночасно, кожен цикл моделі відображає базову концепцію, яка полягає в тому, що цикл являє собою набір операцій, якому відповідає така ж кількість стадій, що і в каскадній моделі.

З іншого боку, відкрита модель вносить нову концепцію обробки і управління зовнішніми змінами. Вони оголошуються невід'ємною частиною процесу виконання проекту і їх завданнями стають, в першу чергу, вирішення проблеми здійсненності проекту, а, по–друге, розв'язання проблеми ризиків, пов'язаних з персоналом, та складністю проекту. Також цикли визначення проекту і огляду вимог суміщені у відкритій моделі з метою спрощення її використання і прискорення процесу виконання проекту. Модель ділиться на квадранти, кожному з яких відповідає мета і кожен з яких складається з множини фаз для досягнення цієї мети.

Цілі, що відповідають квадрантам моделі, можуть бути визначені як:

– визначення цілей, альтернатив і обмежень. Тут визначаються цілі всього проекту, визначається робоча характеристика, перелік функцій, що виконуються, вирішальні чинники досягнення успіху, програмно–апаратний інтерфейс. Визначаються альтернативні способи реалізації продукту (частин продукту): повторне використання, розробка, кооперація, придбання. Визначаються обмеження, що накладаються на використання альтернативних варіантів: час, інтерфейс, обмеження на продукт, зовнішнє середовище, тощо;

– оцінка альтернатив, ідентифікація і розв'язання ризиків. Виконується апробація (прототипування) і оцінка альтернатив, визначених у попередньому квадранті, оцінюються і розв'язуються ризики, приймається рішення про продовження (або припинення) робіт над проектом;

– розробка продукту. Виконується, власне, розробка коду, тестування, документування, складання і випуск продукту. Тут продукт потрапляє до замовника, його (продукту) життєвий цикл трансформується в цикл з первинною метою поліпшення і доробки:

– планування наступної фази. Продукт, що надійшов до замовника, в своєму новому життєвому циклі проходить фази впровадження, супроводу та експлуатації, тоді як в життєвому циклі проекту проводиться підготовка до випуску наступної версії продукту. На підставі даних про впровадження, супровід та експлуатацію випущеного продукту, а також на підставі експертних оцінок і оглядів продукту у процесі розробки виконується планування модифікації процесу і

продукту, що впроваджуються в наступному циклі.

2.2. Процеси управління проектом

У відповідності до [14] процеси управління ІТ проектами діляться на дві категорії – процеси, орієнтовані на продукт, зосереджені на визначенні та створенні проекту, і процеси управління проектами, що описують і впорядковують роботи в проекті. Процеси, орієнтовані на продукт були визначені вище в формальному описі життєвого циклу. Процеси ж управління проектом повинні бути описані окремо, але за умови часового накладення і взаємозв'язку з процесами, орієнтованими на продукт.

При розробці програмного забезпечення, в тому числі з відкритим кодом, виділяються [15, 18] кілька основних процесів управління, що відповідають 5–ти групам процесів управління, визначених в [14]. Решта процесів, що відносяться до продукту, і процеси управління ресурсами, які виконуються протягом всього життєвого циклу, розглядати недоцільно.

1. Визначення цілі і сфери дії програмного проекту. Цей процес втілює процеси ініціалізації і планування. При виконанні проекту визначається одна або кілька цілей, досягнення яких закладається в плані менеджменту (SPMP, Software Project Management Plan) за умови обмежень в ресурсах і якості. План менеджменту і технічне завдання проекту визначають методологічну сторону середовища проекту, а технічна сторона визначається життєвим циклом.

2. Відбір і управління командою розробників. Відбір команди – це найбільш важливий процес з групи процесів виконання відкритого проекту. Зазвичай селекція команди, формування колективу розробників неабияк впливає на інші дії життєвого циклу розробки програмного забезпечення. У відкритої моделі важливим є попереднє планування і формальний опис процесу приєднання сторонніх розробників. Виконання цього процесу і менеджмент стороннього персоналу є невід'ємна частина фаз розробки в відкритому проекті.

3. Забезпечення надійності. Цей процес відноситься до великої групи процесів здійснення контролю за проектом. Процесу приділяється особлива увагу з огляду на його важливість для кінцевого користувача, і він безпосередньо впливає на успішність проекту. Процес створення надійного програмного забезпечення повинен включати в себе, у відповідності до [15], прогнозування, запобігання і усунення помилок, а також реалізацію методик забезпечення стійкості до відмов [19].

2.3. Аналогове моделювання часу прототипування в програмних проектах

Оскільки програмні продукти, що створюються, відносяться до різних "класів", узагальнення досвіду їх створення можливо тільки всередині одного класу, або всередині ще більш вузьких груп.

Найбільш досконалим і перевіреним на практиці апаратом аналогового моделювання є теорія подібності, основні положення якої були розроблені ще на початку минулого століття і найбільш повно узагальнені А.А. Гухманом [16]. Основна ідея теорії подібності полягає в тому, що в межах певного класу явищ, або процесів, виділяються групи, в яких можливо узагальнення даних одиничного досвіду. Узагальнити досвід виконання проекту (створення програмної системи) – значить дати можливість оцінити трудомісткість, вартість і тривалість виконання аналогічного проекту, тобто створити аналогічну (подібну) систему. Це можливо, якщо співвідношення між безрозмірними ознаками (критеріями подібності), що характеризують властивості системи, представити у вигляді апроксимуючої степеневої функції, що найбільш часто використовується:

$$\pi = A \cdot \pi_1^a \cdot \pi_2^b \cdot \ldots \cdot \pi_n^z,$$

де

$\pi, \pi_1, \pi_2, \ldots, \pi_n$ – безрозмірні величини, критерії подібності;

A – коефіцієнт пропорціональності між визначальними і такими, що визначаються, критеріями;

$a, b, \ldots, z$ – показники ступеня при критеріях.

Оскільки проект виконується в часі і не має просторових характеристик, то з умов подібності початкових і граничних умов такими, що застосовуються, залишаються тільки умови подібності початкових умов.

Такі початкові умови для програмних систем, як:

- процес і методи розробки продукту;
- мова програмування;
- платформа для цільової системи;
- інструменти, що використовуються,

повинні бути подібні для систем однієї групи.

Отже, умовами подібності програмних проектів (процесів їх виконання) є:

- належність до одного класу;
- подібність життєвих циклів розробки;
- подібність констант зовнішнього середовища виконання розробки;

– подібність початкових умов входження в життєвий цикл.

Методика застосування апарату теорії подібності для побудови аналогових моделей програмних систем багато в чому ідентична випадку фізичних систем. Для побудови аналогової моделі необхідно застосувати метод аналізу розмірностей, для чого виконати наступну послідовність дій.

1. Ввести множину показників, що визначають стан програмної системи.

2. Побудувати систему одиниць виміру і задати розмірності обраних показників.

3. Представити співвідношення між розмірними величинами у вигляді функції

$$a = A \cdot \nu_1^{\alpha} \nu_2^{\beta} \cdot ... \cdot \nu_k^{\ell} \nu_{k+1}^{\ell+1} \cdot ... \cdot \nu_n^{z} \qquad (1)$$

де

a – величина, що визначається;

$\nu_1, \nu_2, ..., \nu_n$ – величини, що визначають;

n – кількість розмірних величин, що визначають;

k – кількість початкових розмірностей.

4. Знайти співвідношення між безрозмірними величинами, що представляють кількісні співвідношення тієї ж системи у вигляді, визначеному π –теоремою (теоремою Бекінгема):

$$\pi = f_1(\pi_1, \pi_2, ..., \pi_{n-k}), \qquad (2)$$

де $\pi, \pi_1, \pi_2, ..., \pi_{n-k}$ – безрозмірні величини.

Для цього були здійснені наступні дії [17]:

– підставлено розмірності замість самих величин в рівняння (1);
– знайдено суму показників ступеня при однакових основних одиницях вимірювання і записано вирази з сумою в систему рівнянь;
– розв'язано отриману систему рівнянь, виразивши через довільно обрані k показників ступеня інші $n-k$;
– підставлено отриманіні показники ступеня в (1) і перетворено залежність до вигляду (2), угруповуючи величини ν_i при однакових показниках ступеня.

5. Обчислено коефіцієнти пропорційності і показники ступеня критеріїв моделі.

В критеріальному рівнянні (2) вигляд функціональної залежності, а саме коефіцієнти пропорційності і показники ступеня критеріїв моделі визначаються за даними вже виконаного проекту. Для кожного класу процесів буде спостерігатися рівність показників ступенів і коефіцієнтів пропорційності, а також для кожної групи процесів буде спостерігатись рівність критеріїв. Рівняння (2), отримане на 4–му кроці зазначеної вище методики, буде абстрактною аналоговою моделлю процесів виконання програмних проектів.

Рівняння (2) з конкретними значеннями коефіцієнтів пропорційності і показників ступеня критеріїв буде аналоговою моделлю процесів виконання програмних проектів одного класу.

Управління відкритими проектами передбачає збір особливого типу метрик. Розмір програмного коду оцінюється або в функціональних точках або в числі рядків коду SLOC (Source Lines of Code). Кількість розробників враховується двома параметрами – кількістю основних розробників d_c і обсягом залучення сторонніх розробників δ_d, виміряним кількістю розробників, залучених до проекту в одиницю часу. Якість контролюється параметрами b – кількістю внесених помилок і темпом виправлення помилок, r_b – кількістю помилок, що виправляються в одиницю часу. Продуктивність визначається групою параметрів, які враховують зовнішні і внутрішні зміни. Враховується середній обсяг внутрішніх змін (тобто внесених основними розробниками), а саме:

c_ν – кількість одиниць коду, вироблених людиною в одиницю часу, і зовнішні зміни як обсяг надходження доробок в одиницю часу r_b і середній обсяг цих доопрацювань;

p_ν – кількість одиниць коду на одне доопрацювання.

Залежність між обумовленими і визначальними показниками еволюції програмної системи апроксимується як

$$t = A \cdot s^{\alpha} \cdot d_c^{\beta} \cdot \delta_d^{c} \cdot r_p^{d} \cdot p_\nu^{e} \cdot b^{f} \cdot r_b^{s} \cdot c_\nu^{h} \qquad (3)$$

Рівняння (3) записане в термінах розмірностей величин, що входять до нього, приводить до співвідношення вигляду

$$\text{день} = SLOC^{a} \cdot \text{люд}^{b} \cdot \left(\frac{\text{люд}}{\text{день}}\right)^{c} \cdot \left(\frac{\text{дооп}}{\text{день}}\right)^{d} \times \times \left(\frac{SLOC}{\text{дооп}}\right)^{e} \cdot \text{деф}^{f} \cdot \left(\frac{\text{деф}}{\text{день}}\right)^{g} \cdot \left(\frac{SLOC}{\text{день} \cdot \text{люд}}\right)^{h} \qquad (4)$$

Тут кількість розмірних величин $n = 9$, кількість незалежних розмірностей $k = 5$. Відповідно, необхідно отримати $n - k = 4$ критерія.

Система рівнянь, складена з показників ступенів для кожної розмірності з рівняння (4) має вигляд:

$$\text{день}: 1 = -c - d - g - h$$

$$\begin{aligned} &\text{SLOC: } 0 = a + e + h \\ &\text{люд: } 0 = b + c - h \\ &\text{деф: } 0 = f + g \\ &\text{дооп: } 0 = d - e \end{aligned} \qquad (5)$$

Складаючи перше рівняння з другим і друге з третім в системі (5), а також виражаючи інші величини з останніх трьох рівнянь, отримаємо розв'язок системи:

$$\begin{aligned} d &= e \\ f &= -g \\ c &= a + f - 1 \\ b &= 1 - e - f - 2a \\ h &= -a - e \end{aligned} \qquad (6)$$

Підставляючи значення показників з (6) в (5), отримаємо

$$t = A \cdot s^a \cdot d_c^{1-2a-e-f} \cdot \delta_d^{a+f-1} \cdot r_p^e \times \times b^f \cdot p_v^e \cdot r_b^{-f} \cdot c_v^{-a-e} \qquad (7)$$

Угруповуємо в (7) величини з однаковими показниками ступеня:

$$\frac{t\delta_d}{d_c} = A\left(\frac{s\delta_d}{d_c^2 c_v}\right)^a \left(\frac{\delta_d b}{d_c r_b}\right)^f \left(\frac{r_p p_v}{d_c c_v}\right)^e, \qquad (8)$$

де виділяються чотири безрозмірних критерії:

$F_t = \frac{t \cdot \delta_d}{d_c}$ - характеристика часу відкритого проекту;

$F_{scope} = \frac{s \cdot \delta_d}{d^2 \cdot c_v}$ - характеристика масштабу проекту;

$F_{qua} = \frac{b \cdot \delta_d}{d_c \cdot r_b}$ - характеристика якості продукту;

$F_{ext} = \frac{r_p \cdot p_v}{d_c \cdot c_v}$ - характеристика обсягу зовнішніх змін.

У підсумку, абстрактна аналогова модель еволюції програмної системи загального призначення, що розробляється за відкритою моделлю, виражена критеріальним рівнянням:

$$F_t = A \cdot F_{scope}^a \cdot F_{qua}^f \cdot F_{ext}^e$$

3. Аналіз отриманих результатів

3.1. Застосування методики управління часом створення ІТ проекту

Для визначення можливості успішного завершення проекту необхідно довести можливість застосування певної моделі. Обґрунтування можливості застосування моделі здійснюється за уточненим методом аналізу характерних категорій, в порівняльні таблиці якого додається інформація про вибрану модель. При цьому виконано наступні дії:

– проведено аналіз відмінних категорій проекту:

1) категорії вимог;
2) категорії команд розробників;
3) категорії колективу користувачів;
4) категорії типу проекту і ризиків.

– розташовано за ступенем важливості категорії або питання, що відносяться до кожної категорії, щодо проекту, для якого вибирається прийнятна модель.

– проведено підрахунок показників (так/ні) кожної моделі і розв'язання суперечностей при подібних показниках за допомогою градації категорій і питань за ступенем важливості.

Після обґрунтування вибору вибраної моделі для програмного проекту, визначено ризики, пов'язані з цим вибором. У разі значних ризиків необхідно скласти план їх розв'язання, або переглянути вибір моделі.

Для програмних проектів, наприклад, з відкритим кодом виділяються наступні джерела ризиків:

– ризики середовища, пов'язані із середовищем виконання проекту:

– ризики гнучкості, пов'язані з використанням швидких методів розробки:

– ризики планування, пов'язані з використанням методів розробки з переважною часткою планування:

– ризики відкритості, пов'язані із застосуванням відкритих методів розробки:

Ризики з наведеного вище списку оцінюються і порівнюються. Завданням порівняння є з'ясування ступеню ризику використання відповідної моделі. Саме порівняння виконано на якісному рівні, і за його результатами ризикам присвоюється рейтинг – значення від 0 до 4, де

0 – мінімальний ризик;
1 – середній ризик;
2 – серйозний, але керований ризик;
3 – дуже серйозний, але керований ризик;
4 – некерований ризик.

Для оцінки ризику побудовано дерево рішень.

За методом аналітичної ієрархії сформовано матриці попарних порівнянь на кожному рівнів критеріїв та рівня альтернатив; обчислено вектори пріоритетів, індекси та відношення узгодженості оцінок ризиків.

Після узгодження оцінок ризиків відбувається синтез глобальних пріоритетів альтернатив відносно фокуса ієрархії – глобального рівня ризику щодо вибору моделі.

На етапі моделювання параметрів управління прототипуванням виконуються попередні оцінки часу виконання, відповідно до яких і уточнюється план. Оцінки часу виконуються з використанням аналогової моделі для відповідних типів проектів.

Для кожної групи подібних проектів обчислені значення критеріїв $F_t, F^a_{scope}, F^f_{qua}, F^e_{ext}$.

Значення кожного із отриманих критеріїв розглянуті як члени статистичної сукупності з розподілом Стьюдента, для якої розраховано вибіркове середнє, виправлене середнє квадратичне відхилення і оцінюється математичне сподівання за допомогою довірчого інтервалу з надійністю $\gamma = 0,95$. Такий розподіл вибрано відповідно до теорії математичної статистики для малих вибірок (з $n < 30$, де n – обсяг вибірки).

Розрахунок вибіркової середньої, виправленого середнього відхилення і меж довірчого інтервалу проводиться за формулами:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i ,$$

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2} ,$$

$$\overline{x} - t_\gamma \frac{S}{\sqrt{n}} < a < \overline{x} + t_\gamma \frac{S}{\sqrt{n}} ,$$

де

$\overline{x}$ – вибіркове середнє,

x_i, i – й елемент вибірки,

n – число елементів у вибірці,

S – виправлене середнє квадратичне відхилення.

t_γ – функція точності оцінки, що залежить від числа елементів у вибірці і надійності;

γ вибирається за таблицями розподілу Стьюдента.

Розрахунок проводиться для даних, отриманих з відкритих джерел.

3.2. Тестування методики на виконаних проектах

Методика тестувалась на IT проектах, вихідні дані яких отримані з відкритих джерел, зокрема, для інструментальних бібліотек (IT проекти з відкритим програмним кодом).

Для класу інструментальних бібліотек обчислені критерії $F_t, F^a_{scope}, F^f_{qua}, F^e_{ext}$ та їх довірчі інтервали:

- $F_t \in [0,58; 0,64] \Rightarrow F_t = 0,61$;
- $F_{scope} \in [0,73; 0,79] \Rightarrow F_{scope} = 0,76$;
- $F_{qua} \in [0,00012; 0,00018] \Rightarrow F_{qua} = 0,00015$
- $F_{ext} \in [1,23; 1,25] \Rightarrow F_{ext} = 1,24$.

Із середньовибіркових значень критеріїв складена система нелінійних алгебраїчних рівнянь:

$$F_{t1} = A \cdot F^a_{scope1} \cdot F^f_{qua1} \cdot F^e_{ext1},$$

$$F_{t2} = A \cdot F^a_{scope2} \cdot F^f_{qua2} \cdot F^e_{ext2},$$

……..,

$$F_{tm} = A \cdot F^a_{scopem} \cdot F^f_{quam} \cdot F^e_{extm}$$

Її розв'язання дає значення необхідних параметрів критеріального рівняння:

$$A = 911;\ a = 2,1;\ f = 0,68;\ e = -2,8.$$

Розкриваючи критерій часу, модель оцінки часу прототипування набирає вигляду:

$$t = 911 \cdot 0,76^{2,1} \cdot 0,00015^{0,68} \cdot 1,24^{-2,8c} \cdot \frac{d_c}{\delta_d},$$

де d_c – кількість розробників, залучених офіційно до розробки проекту, δ_d – експертна оцінка обсягу залучення сторонніх розробників.

Використовуючи цю оцінку часу, був складений план виконання проекту і проведено порівняння з відповідним параметром уже виконаного проекту. Відносна похибка оцінки часу склала 15%.

4. Висновки

Розроблена модель життєвого циклу програмних проектів з відкритим програмним кодом, яка визначає перелік і послідовність дій для виконання в проекті, визначає місце процесів управління в життєвому циклі.

Сформульована узагальнена методика побудови аналогової моделі для оцінки часу прототипування програмних проектів.

Отримана абстрактна аналогова модель еволюції програмної системи загального призначення, яка виражається критеріальним рівнянням.

Здійснена реалізація методики управління програмними проектами, яка включає:

– обґрунтування вибору моделі методом аналізу характерних категорій;

– аналіз ризику вибору моделі;

– створення аналогової моделі життєвого циклу проекту;

– визначення часу прототипування проекту.

Проведено порівняння отриманих результатів з даними із відкритих джерел, що показує високий відсоток достовірності. Зроблено висновок про можливість застосування розробленої методики для прогнозування можливості успішного завершення різних класів IT проектів на ранніх стадіях проектування.

Список використаної літератури

1. CHAOS Report [Electrical Resource]. – The Standish Group International. Inc. http://www.projectsmart.co.uk/docs/chaos–report.pdf

2. Jian, Z., Why IT Projects Fail [Text]// Computervvorld. – 2005. – Vol. 39, № 6. – P. 31–32.

3. Glass, R. L., System Development Glass Column [Text]// System Development. – 1988. – Vol. 1.№1.–P. 4–5.

4. Брукс, Ф., Мифический человеко–месяц или как создаются программные системы. [Текст] – СПб.: Символ–Плюс. 2006. – 304с.

5. Mystery Solved! Linux is Cheaper – PERIOD. [Text] / The Standish Group International, Inc. // VirtualBEACON. – 2004. № 347. – P. 1–3.

6. Ройс, У., Управление проектами по созданию программного обеспечения. Унифицированный подход. [Text] – М.: Издательство "Лори". 2002. – 424с.

7. Schindler, E., OSS/Linux Development Survey [Electrical Resource] // Evans Data Corporation Strategic Reports http://evaiisdata.coin/reports/viewRelease.php?reportID=7.

8. Routh, E. T., A Treatise on the Stability of a Given State of Motion [Text]. – London: Taylor & Francis, 1975. – 297 p.

9. Little, G. Healey, M., Worldwide Open Source Sendees 2007–2011 Forecast [Electrical Resource] http://www.idc.com/getdoc.isp?containerId=20S255.

10. NetCraft Internet Research Analysis [Electrical Resource] http://www.netcraft.coin.

11. Ghosh, R. A., Study on the economic impact of FLOSS on innovation and competitiveness of the EU ICT sector: Final Report [Text]/ UNU–MERIT. – Contract ENTR/04/112. – The Netherlands. 2006. – 2S7p.

12. ISO TEC 12207:1995. Software life cycle processes [Text]. – Geneva. Switzerland: International Organization for Standardization. – 57p.

13. ДСТУ 3918–1999 (ISO IEC 12207:1995) Державний стандарт України. Інформаційні технології. Процеси життєвого циклу програмного забезпечення. [Text] – К.: Держстандарт України, 2000. – 49 с.

14. Керівництво з питань проектного менеджменту [Text] / Під ред. С.Д. Бушуева. – К.: Видавничий дім "Деловая Украина". 2000. – 198 с.

15. Шафер, Д. Ф., Фартредт, Р. Т., Шафер, Л. И., Управление программными проектами: достижение оптимального качества при минимуме затрат. [Text] – М.: Издательский дом "Вильямс", 2003. – 1136 с.

16. Гухман, А. А., Применение теории подобия к исследованию процессов тепло–массообмена. [Text] –М.: "Высшая школа", 1974.–328с.

17. Sytnyk, V. A., Bulashov, V. V., Methodology for managing the development of it projects with open source [Text]/ 5th International conference on Eurasian scientific development in 2018: new methods and solutions». Proceedings of the Conference (September 02, 2018). Premier Publishing s.r.o. Vienna. 2018.46 p. ISBN-13 978-3-903197-73-2

18. Bushuyev, S. D., Bushuiev, D. A., Yaroshenko, R. F., Chernova, L. S., Threats management principles for development programs of high technology industries in turbulent environment [Text]// Zeszyty Naukowe. Organizacja i Zarządzanie / Politechnika Śląska, – 2017. – z. 105. – pp. 9–30.

19. Mihić, M. M., Dodevska, Z. A., Todorović, M. L., Obradović, V. L., Petrović, D. Č., Reducing Risks in Energy Innovation Projects [Text]: Complexity Theory Perspective. Sustainability. 2018; 10(9): 2968.

References

1. CHAOS Report. – The Standish Group International. Inc. http://www.projectsmart.co.uk/docs/chaos–report.pdf

2. Jian, Z., (2005) Why IT Projects Fail Computervvorld. – Vol. 39, № 6. – P. 31–32.

3. Glass, R. L., (1988) System Development Glass Column, System Development. – Vol. 1.№1, P. 4–5.

4. Brooks, F., (2006) Mythical man-month or how software systems are created. [Mificheskiy coздatcheloveko-mesyatz ili kak sozdayutsya programnye cictemy] - SPb .: Symbol – Plus. - 304s.

5. Mystery Solved! Linux is Cheaper – PERIOD. The Standish Group International, Inc., VirtualBEACON. – 2004. № 347. – P. 1–3.

6. Royce, U., (2002) Software Project Management. Unified approach [Upravlenie proektami po sozdaniyu progrramnogo obespecheniya. Unificirovannyi podhod]. M .: Publishing house "Lori". - 424c.

7. Schindler, E., OSS/Linux Development Survey // Evans Data Corporation Strategic Reports http://evaiisdata.coin/reports/viewRelease.php?reportID=7.

8. Routh, E. T., (1975) A Treatise on the Stability of a Given State of Motion. – London: Taylor & Francis – 297 p.

9. Little, G., Healey, M., Worldwide Open Source Sendees 2007–2011 Forecast http://www.idc.com/getdoc.isp?containerId=20S255.

10. NetCraft Internet Research Analysis http://www.netcraft.coin.

11. Ghosh, R. A., (2006) Study on the economic impact of FLOSS on innovation and competitiveness of the EU ICT sector: Final Report / UNU–MERIT. – Contract ENTR/04/112. – The Netherlands. – 2S7p.

12. ISO TEC 12207:1995. Software life cycle processes. – Geneva. Switzerland: International Organization for Standardization. – 57p.

13. DRSTU 3918–1999 (2000) (ISO IEC 12207:1995) State standard of Ukraine. Information Technology. Process Lifecycle Software [Derzhavny standart Ukrainy. Informaciyni technologii. Procesy zhyttevogo zyklu programnogo zabezpechennya]. K. Gosstandard of Ukraine. - 49 p.

14. S. D. Bushuyev, (2000) Manual on Project Management [Kerivnyctvo z pytan proektnogo menedzmentu], K. Publishing House "Delovaya Ukraina". 198 p.

15. Schafer, D. F., Fartredt, P. P., Shafer, L. I., (2003) Management of software projects: achieving optimal quality at minimum cost [Upravlenie programnymi proektami: dostizeniye optimalnogo kachestva pri minimum zatrat]. M .: Publishing house "Williams". 1136 p.

16. Gukhman, A. A. (1974) Application of the theory of similarity to the study of heat – mass transfer processes [Primeneniye teorii podobiya k issledovaniyu processov teplo-massoobmena]. M .: Higher School. – 328s.

17. Sytnyk, V. A., Bulashov, V. V., (2018) Methodology for managing the development of it projects with open source. 5th International conference on Eurasian scientific development in 2018: new methods and solutions». Proceedings of the Conference (September 02, 2018). Premier Publishing s.r.o. Vienna. 46 p. ISBN-13 978-3-903197-73-2

18. Bushuyev, S. D., Bushuiev, D. A., Yaroshenko R. F., Chernova L. S., (2017) Threats management principles for development programs of high technology industries in turbulent environment, Zeszyty Naukowe. Organizacja i Zarządzanie. Politechnika Śląska, – z. 105. – pp. 9–30.

19. Mihić, M. M., Dodevska, Z. A., Todorović, M. L., Obradović, V. L., Petrović D. Č., (2018) Reducing Risks in Energy Innovation Projects: Complexity Theory Perspective. Sustainability. 10(9): 2968.

MODELING OF PROTOTYPE PARAMETERS IN MANAGEMENT OF IT PROJECTS

V. A. Sytnyk
Odessa National Polytechnic University

Abstract. *The analysis of methods and techniques for managing the creation of open source software has been carried out, the shortcomings of the design of open source software products and ways to eliminate them have been identified. A model for the life cycle of open source software projects has been developed. Proposed process of analog modeling for the creation of IT projects. The methodology is based on the similarity theory, mathematical statistics, mathematical modeling, system analysis, the use of computer systems and modern software systems. To apply it, the project's belonging to a certain class is determined in advance and verified by the methods of system analysis of project risks for the selected life cycle. The rationale for the application of the model is based on an improved method for analyzing characteristic categories. Analog modeling includes determining a set of indicators that determine the state of a software system, choosing a system of units of measurement and setting dimensions, determining components and indicators of degrees of*

an equation criterion. From the equation of the criterion, time is determined for prototyping open source projects. The methodology is theoretically justified and allows you to predict the time and cost of a future project in the early stages of creating IT projects. The method was tested on completed projects and shows a high percentage of reliability. Can be used to create new IT projects of different classes. The practical value of the results of work and their practical use is determined by using the developed methodology as the basis of knowledge in the management of open source software projects.

Keywords: *Clustering, similarity theory, criterion equation, system analysis, risk analysis, dimensional analysis.*

МОДЕЛИРОВАНИЕ ВРЕМЕНИ ПРОТОТИПИРОВАНИЯ В УПРАВЛЕНИИ IT ПРОЕКТАМИ

В. А. Ситник
Одесский национальный политехнический университет

Аннотация. *Предложена методика аналогового моделирования процесса создания IT проектов. Методика теоретически обоснована, и позволяет на ранних этапах прогнозировать время и стоимость будущего проекта. Методика проверена на завершенных проектах и показывает высокий процент достоверности. Может быть применятся при создании новых IT проектов разных классов.*

Ключевые слова: *теория подобия, аппроксимация, критериальное уравненние, классификация, жизненный цикл, анализ размерностей.*

Отримано 08.02.2019

Ситник Володимир Анатолійович, кандидат фізико-математичних наук, доцент кафедри інформаційних Одеського національного політехнічного університету. Просп. Шевченка, 1, Одеса, Україна, E-mail: vladas@ua.fm, тел. +38-099-360-50-01

Volodymyr Sytnyk, candidate of physical and mathematical sciences, associate professor of information department of Odessa National Polytechnic University. Prospekt Shevchenko, 1, Odessa, Ukraine, E-mail: vladas@ua.fm, tel. +38-099-360-50-01

ORCID ID: 0000-0001-5943-4581