УДК 681.31

к.т.н. Кривоножко Г.Є. (ВІТІ)
Петров Д.В. (ВІТІ)
Жовтун А.А. (ВІТІ)

АЛГОРИТМ ВИЗНАЧЕННЯ ТРУДОВИТРАТ СУПРОВОДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА ОСНОВІ МОДЕЛІ COCOMO II

У статті розглянуто алгоритм визначення трудовитрат супроводження програмного забезпечення інформаційних систем на основі моделі COCOMO II, що дає можливість Замовнику оцінити спроможності Розробника за основними показниками.

Кривоножко Г.Е., Петров Д.В., Жовтун А.А. Алгоритм определения трудозатрат сопровождения программного обеспечения на основе модели COCOMO II. *В статье рассмотрен алгоритм определения трудозатрат на сопровождение программного обеспечения информационных систем на основе модели COCOMO II, что дает возможность Заказчику оценить способности разработчика по основным показателям.*

G. Kryvonozhko, D. Petrov, A. Zhovtun Algorithm for determining the labor cost of software maintaining based COCOMO II model. *The article considers the algorithm for determining labor costs of software information systems based on the COCOMO II model, which allows Customer ability to assess the Developer's key indicators.*

Ключові слова: *життєвий цикл програмного забезпечення, супроводження, трудовитрати, Constructive Cost Model, COCOMO II.*

Під час розробки та впровадження інформаційних систем важливою стадією є супроводження програмного забезпечення [1 – 5]. Якість програмного забезпечення недосяжна без контролю процесу його розробки та супроводження [6 – 7]. Необхідним є контроль створення програмного забезпечення, прогноз його вартості, необхідність дотримання термінів і якості результатів, прогноз вартості супроводження програмного забезпечення у визначені терміни при дотриманні показників [6].

Етапи розробки та впровадження програмного забезпечення інформаційних систем регламентовані вітчизняною галузевою та міжнародною нормативною базою [4 – 5]. Стандарт [4] є основним нормативним документом, який регламентує схему процесу життєвого циклу ПЗ. *Супровід програмного забезпечення (супроводження)* визначається як процес зміни існуючого програмного забезпечення при цьому не змінюючи його основних функцій [5]. В силу специфіки створення програмного забезпечення, процес супроводження також є досить дорогим, тривалим та важко передбачуваним [6].

У більшості компаній по створенню програмного забезпечення результативне виконання робіт у визначені терміни здійснюється лише у 25 % проектів, для решти проектів умови договорів не виконуються за різними показниками: порушення термінів виконання, збільшення бюджету тощо.

У більшості компаній по створенню програмного забезпечення інформаційних систем більше 50 % штату зайнято супроводом програм, що входять до складу систем. В таких умовах виникає потреба на початковому етапі виконання робіт визначати Розробника з достатніми спроможностями для успішної реалізації проекту (визначення трудовитрат).

Постановка задачі. Метою роботи є розробка алгоритму визначення трудовитрат на супроводження програмного забезпечення. Дослідження спрямоване на підвищення рівня нормування трудовитрат, планування та обліку витрат на супроводження програмного забезпечення.

Аналіз останніх досліджень і публікацій. Аналіз досліджень і публікацій свідчить про те, що даному питанню приділяється певна увага [1 – 3, 6 – 9].

Оцінка трудовитрат програмного забезпечення є однією з ключових фінансових проблем, які виникають в процесі створення та розвитку інформаційних технологій [6], відноситься до імовірнісних тверджень.

Виходячи з процесу проектування, поділяють існуючи моделі оцінювання трудовитрат програмного забезпечення на дві групи [10]: *неалгоритмічні*, що використовують визначені схеми або принципи (до найбільш відомих з них відносяться експертні оцінки), та *алгоритмічні*, що засновані на підрахунку кількісних характеристик програми у вигляді числа операторів або функціональних точок.

Експертні оцінки, в першу чергу, застосовуються для проектів, що вирішують інноваційні завдання або засновані на новітніх технологіях та процесах. Зважаючи на інноваційність значної частини створюваного програмного забезпечення, такі оцінки набули значного поширення у практиці розроблення програм [6].

Часто експертами виступають безпосередньо замовники та розробники, що мають певний досвід. На основі оцінок окремих експертів у відповідності до існуючих методик формується інтегрована оцінка. До загальних недоліків методів експертного оцінювання відноситься непрогнозований „людський" фактор. Ступінь повноти і достовірності інформації, поданої експертами, оцінити важко та відсутні аналітичне обґрунтування.

Метод оцінки з аналогії [11] є загальноприйнятим для будь-якого оцінювання і вважається різновидом експертної оцінки. Він використовує у якості оцінки емпіричні дані про вже завершені проекти схожого типу.

Алгоритмічні моделі базуються на математичних моделях і постійно удосконалюються з метою підвищення їх точності оцінювання. Найчастіше реалізованими і добре документованими моделями є модель Путнем (ступенева, аналітична) і модель *COCOMO* (ступенева, емпірична) [12 – 13].

Модель Путнем (*SLIM*) [6] є найбільш поширеною моделлю аналітичної групи. Вона створена для проектів об'ємом більше 70 000 рядків коду. Модель ґрунтується на твердженні, що витрати на розробку програмного забезпечення розподіляються згідно кривим Нордена-Рейлі, які є графіками функції, що представляє розподіл робочої сили у часі. Однак в моделі є ряд недоліків [6], що у ряді випадків робить оцінку з використанням даної моделі непридатною до використання. Перевагою моделі Путнем перед *COCOMO* 1.1 або *COCOMO* 2.0, є невелика кількість параметрів, необхідних для оцінки.

Модель *COCOMO* була представлена як альтернативна реалізація моделі Путнем і застосована в комплексі *SLIM Estimate* для оцінки вартості програмного забезпечення [6]. Обмеженням цієї моделі є те, що вона може бути використана, якщо передбачувані витрати на створення програмного забезпечення більше 20 людино-місяців. У цілому вважається, що для невеликих проектів застосування високих рівнів оцінювання *COCOMO* неефективне, а базовий рівень дає недостатню точність.

Для комерційних застосувань метод *COCOMO* дає, як правило, завищені оцінки, тому його застосовують більше в проектах, що відносяться до програмного забезпечення інженерних розрахунків [13]. Модель *COCOMO II* є найбільш сучасною: підтримує підходи щодо застосуванням комерційних готових продуктів, варіанти часткового повторного та модифікованого коду; керований ризиками і спільний програмний процес; враховує зрілість програмних процесів тощо. Найпопулярнішою серед алгоритмічних моделей є сімейство моделей *COCOMO* (*Constructive Cost Model*), створене у 1981 р. Модель використовує формулу регресії з параметрами, визначеними з даних, зібраних по ряду проектів. Відомі модифікації COCOMO у вигляді сімейства моделей *COCOMO* II [13], що створені у 1999 р. Основними відмінностями *COCOMO* II від *COCOMO* є використання для оцінювання складності вхідних даних у вигляді функціональних точок, оцінювання елементів повторного використання та інтеграції програмних продуктів, об'єктно-орієнтовані підходи до оцінки компонентів програмного забезпечення та ін.

Існують й інші підходи щодо встановлення показників трудовитрат, але вони ґрунтуються на базових нормах, які встановлюються на основі дослідно-статистичних або експертних норм, які коректуються на показники складності програм. Але такі норми не враховують стан культури програмування, кваліфікацію програмістів тощо.

Також звернемо увагу на існуюче протиріччя у термінології на даний час. Так, *інформаційна (автоматизована) система (англ. Information system, IC, AC, Система)* – це організаційно-технічна система, в якій реалізується технологія обробки інформації з використанням технічних і програмних засобів [14]. Між поняттями «*програма*» та «*програмне забезпечення*» є відмінності. Також є ряд розбіжностей щодо існуючих підходів до визначення та понять *комп'ютерна програма*, *база даних*. Тобто, під час проведення досліджень та розрахунків щодо трудовитрат супроводження ПЗ слід охоплювати всі суттєві ознаки об'єкта дослідження.

Наприклад, для поняття „комп'ютерна програма" такими ознаками є: 1) набір команд (інструкцій) для комп'ютера тощо; 2) функціональне призначення із зазначенням широкої апаратної сфери застосування; 3) форма вираження без детальної конкретизації тощо. *Програмне забезпечення* (*англ. Software, програмний продукт, програмний засіб, ПЗ*) – це сукупність програм систем обробки інформації та програмних документів, необхідних для їх експлуатації. На сьогодні можна сказати, що склалися такі *групи програмного забезпечення*: операційні системи та оболонки; системи програмування (транслятори, бібліотеки підпрограм тощо); інструментальні системи; інтегровані пакети програм; динамічні електронні таблиці; системи машинної графіки; системи управління базами даних; прикладне програмне забезпечення.

Тепер розрахунок трудовитрат на супроводження програмного забезпечення інформаційних систем на практиці проводиться в основному на базі експертних оцінок. Існує можливість для крупних проектів для розрахунку орієнтовних трудовитрат супроводження програмного забезпечення.

Викладемо більш детально алгоритм розрахунку трудовитрат на супроводження програмного забезпечення на основі моделі конструктивних витрат *COCOMO II*, що використовує Бейесовський аналіз для обробки статистичних даних, який у більшому ступені відповідає особливостям статистичних даних програмних проектів, що характеризуються неповнотою і неоднозначністю.

Виклад основного матеріалу дослідження. Під час розрахунків щодо визначення трудовитрат супроводження програмного забезпечення *об'єктами дослідження* є *вихідні коди, технічно-експлуатаційна та договірна документація.*

Обмеження. Слід звернути увагу, що варіант розрахунку трудовитрат на *стадії супроводження ПЗ* можливий у разі наявності *змін у вихідному коді*. У випадках, коли Замовник та Розробник розглядає супроводження як технічну підтримку, варіант розрахунку можливий на основі інших моделей та методів.

Модель *COCOMO II* існує в трьох видах, адаптована до сучасних методологій розробки програмного забезпечення, а також придатна для використання зі спіральною та ітераційною моделями життєвого циклу. Чинниками, що впливають на точність оцінки трудовитрат, при використанні засобів на основі моделі *COCOMO II*, є наступні: правильний вибір конкретної реалізації моделі; точність калібрування – відповідність установок вихідним даним. У зв'язку з цим, для застосування засобів повинен використовуватися *персонал* (*експерт, спеціаліст*), який *не має прямого відношення до процесів проектування і розробки ПЗ*. Вибір того або іншого виду моделі *COCOMO II* для оцінки трудовитрат супроводження програмного забезпечення залежить від типу проекту і стадії розробки [13]. На ранніх стадіях розробки проекту застосовують модель композиції додатків – *ACM* (*Application Composition Model*), яка придатна для використання у проектах, що розробляються з використанням сучасних інструментальних засобів, основаних на об'єктно-орієнтованих технологіях. Модель раннього проектування – *EDM* (*Early Design Model*) включає вивчення альтернативної архітектури і концепцій роботи. На цій стадії недостатньо загального опису проекту, потрібно деталі.

Придатна для приблизного оцінювання витрат на розробку проекту до того, як була визначена його архітектура. Може застосовуватися для техніко-економічного обґрунтування витрат на створення ПЗ, а також для розподілу витрат по стадіях розробки. Пост-

архітектурна модель – *PAM* (*Post Architecture Model*) пов'язана з реальною розробкою і експлуатацією програмного продукту. Ця модель працює найефективніше. Між різними моделями існує певна схожість. Кожна модель оцінювання має свої переваги та недоліки [13].

Найбільш фундаментальними розрахунками в моделі *COCOMO II* є *використання рівняння для оцінки кількості людино-місяців*, необхідних для розробки проекту. Більшість інших результатів *COCOMO II*, в тому числі оцінки вимог та технічне обслуговування, є похідними від цієї величини. Для розрахунку трудовитрат необхідно визначитись із *вхідними даними* (значення констант, експонент, факторів масштабу, множників витрат) в залежності від виду моделі *COCOMO II*.

Найбільш поширеними є наступні *одиниці оцінки розміру ПЗ* [13]:

– кількість рядків коду (*Lines Of Code, LOC*);

– функціональні точки (*Function Points, FP*);

– кількість різних елементів у складі управлінської специфікації;

– обсяг документації тощо.

Кількість рядків коду (*Lines Of Code, LOC; Source Lines of Code, SLOC*) є найпростішою і найпоширенішою серед зазначених одиниць виміру. У загальному випадку *LOC* означає кількість рядків коду на відповідній мові програмування, які мають бути написані для того, щоб проект був виконаний [13].

Оскільки різні мови програмування мають різні можливості і різну продуктивність по вирішенню задач програмування, то одиниці *LOC* мають бути приведені до певної співрозмірної величини.

Приведення здійснюється з використанням таблиць перетворень, які періодично поновлюються, щоб враховувати еволюцію мов програмування [13]. Слід звернути увагу, одиниця розміру *LOC не відображає функціональні властивості коду*.

Рекомендації щодо оцінки факторів адаптованого програмного забезпечення для різних категорій коду (новий, повторно використаний, автоматично переведений код тощо), використовуючи *COCOMO II*, детально викладено в [13].

Супроводження програмного забезпечення включає в себе додавання нових можливостей і фіксацію або адаптацію існуючих можливостей, виключає зміни відновлення основного продукту більше 50 % від існуючого програмного забезпечення і розвиток значної (більше 20 % змін) взаємодії систем, що вимагають невеликого доопрацювання існуючої системи. Існують припущення, зроблені в моделі *COCOMO II* [13], що в цілому вартість супроводження програмного забезпечення має атрибути параметрів вартості як під час розробки програмного забезпечення. Супроводження включає в себе перепроектування та перекодування невеликими частинами оригінального продукту, перепроектування та розробку інтерфейсів, і невелику модифікацію структури продукту. Супровід може бути класифікований як поновлення або доробка продукту.

У *COCOMO II* трудовитрати виражаються в *людин-/місяцях* (*PM*). Людино-місяць – кількість часу, яке одна людина витрачає, коли працює над розробкою програмного забезпечення протягом одного місяця. *COCOMO II* розглядає кількість людино/годин на людино-місяць, *PH/PM*, як регульований фактор з номінальною вартістю 152 годин на людино-місяць (це число не включає, зазвичай, свята, канікули, вихідні та вільний час).

Для розрахунку трудовитрат супроводження (Рівняння 1) вхідними даними є розмір розробки програмного забезпечення *Size*, константа *A*, експонента *E* і ряд значень факторів масштабу *SM* та множників витрат *EM*, кількість яких залежить від вибраної моделі [13].

$$PM = A \times Size_M^E \times \prod_{i=1}^{n} EM_i \,, \qquad (1)$$

де $A = 2{,}94$ (для COCOMO II.2000),

розмір $(Size)_M$ виражається в тисячах рядків вихідного коду (*kilolines of code, KSLOC*) або некоректованих функціональних точках (*UFP*),

E (*cost driver, параметр вартості*) – величина, яка оцінює різні тимчасові, якісні та ресурсні аспекти розробки ПЗ, параметр, який може бути відкалібрований,

EM – (*Effort Multiplier, множник витрат*) – значення, пов'язане з певним рейтингом параметру вартості.

Найбільш значним внеском у моделі *COCOMO II* є *розмір супроводження* $(Size)_M$, який, зазвичай, розраховується за Рівнянням 2, коли відомі розмір основного коду і відсоток змін в основному коді.

$$(Size)_M = [(BaseCodeSize) \times MCF] \times MAF, \quad (2)$$

де $BaseCodeSize$ – розмір основного коду,
MCF (*Maintenance Change Factor, коефіцієнт зміни супроводу*) – відсоток змін в основному коді,
MAF (*Maintenance Adjustment Factor, поправочний коефіцієнт супроводу*) – використовується для обліку програмного забезпечення і розуміння незрозумілих ефектів.

MCF представляє відношення у Рівнянні 3.

$$MCF = \frac{SizeAdded+SizeModified}{BaseCodeSize}, \quad (3)$$

де $SizeAdded$ – розмір повторно використаного коду,
$SizeModified$ – розмір модифікованого коду.

Відома простіша версія, що може використовуватися, коли частину коду було додано або змінено до існуючого основного коду впродовж інтервалу супроводу. Видалений код не враховується (Рівняння 4).

$$(Size)_M = (SizeAdded + SizeModified) \times MAF. \quad (4)$$

Розмір може відноситися до тисяч рядків вихідного коду (*KSLOC*), функціональних точок тощо. Поправочний коефіцієнт супроводу *MAF* (Рівнянням 5) використовується, щоб скоректувати ефективний розмір супроводу для розуміння програмного забезпечення та ефекту незнання програмного забезпечення при повторному використанні. *COCOMO II* використовує коефіцієнти розуміння програмного забезпечення (*Software Understanding – SU*) і незнання програмного забезпечення (*Programmer Unfamiliarity with Software – UNFM*) відповідно до п. 2.4 [13].

$$MAF = 1 + \left(\frac{SU}{100} \times UNFM\right). \quad (5)$$

Відповідно до [13] *COCOMO II* при розрахунках необхідною умовою є застосування масштабних коефіцієнтів.

Експонента *E* є сукупністю п'яти *масштабних коефіцієнтів* (*SF*), що використовуються для розробки програм різного розміру. Кожний з факторів масштабу має діапазон рівнів рейтингу від дуже низького (*Very Low, VL*) до надвисокого (*Extra High, XH*). Кожен рівень рейтингу має вагу. Конкретне значення ваги називається масштабним коефіцієнтом (*SF*). Масштабні коефіцієнти проекту, обраним масштабом факторів рейтингів підсумовуються і використовуються для визначення показника масштабу *E*.

$$E = B + 0{,}01 \times \sum_{j=1}^{5} SF_j, \quad (6)$$

де *B=0,91* (для COCOMO II.2000),

SF (*Scale Factor, фактор масштабу, масштабний коефіцієнт*) – значення певного рейтингу масштабного коефіцієнта, що визначаються характеристиками проекту.

Для розрахунку трудовитрат супроводження використовується наступні фактори масштабу: *PREC – Precedentedness scale factor* (фактор масштабу „Прецедентність") – наявність досвіду аналогічних розробок (Very Low – досвід в продукті і платформі відсутні; Extra High – продукт і платформа повністю знайомі); *FLEX – Development Flexibility* (фактор масштабу „Гнучкість розвитку") – гнучкість процесу розробки (Very Low – процес строго детермінований; Extra High – визначені тільки спільні цілі); *RESL – Architecture and Risk Resolution scale factor* (фактор масштабу „Архітектура і дозвіл ризиків") (Very Low – ризики невідомі / не проаналізовані; Extra High – ризики дозволені на 100 %); *TEAM – Execution Time Constraint Cost Driver* (обмеження з доступності програмного середовища) – враховує часові ресурси, які використовуються ПЗ при виконанні поставленого завдання; *PMAT – Process Maturity scale factor* (фактор масштабу „Зрілість процесів") – технологічна зрілість розробки. Детально таблиці розрахунку значень факторів масштабу наведено в [13].

Параметри вартості (cost drivers) використовуються для опису характеристик розробки програмного забезпечення, які впливають на трудовитрати для завершення проекту. Всі параметри вартості *COCOMO II* мають якісні рівні рейтингу, які висловлюють вплив параметра на трудовитрат щодо розвитку.

Ці рейтинги можуть варіюватися від наднизької (*Extra Low, XL*) до надвисокої (*Extra High, XH*). Кожен рівень рейтингу будь якого мультиплікативного параметру вартості має значення, що називається *множник трудовитрат* (*effort multiplier, EM*). Ця схема переводить рейтинг параметра вартості в кількісну величину для використання в моделі. Параметр вартості – це суб'єктивна величина, яка оцінює різні тимчасові, якісні та ресурсні аспекти розробки ПЗ. Кожен з параметрів може бути відкалібрований. Калібрування параметрів вартості – це коригування значень параметрів, яка впливає на значення трудовитрат, і отже на час і вартість, при оцінці програмного проекту.

Рейтинг параметрів вартості заснований на вагомих аргументах. Модель раннього проектування та постархітектурна модель відрізняються кількістю мультиплікативних параметрів вартості. Є сім мультиплікативних параметрів вартості для моделі раннього проектування і сімнадцять мультиплікативних параметрів вартості для пост-архітектурної моделі. Масштабні коефіцієнти в експоненті *E* використовуються тільки *на рівні проекту*. Крім того, один із множників трудовитрат, що використовується в продукті, необхідний графік розробки (*Required Development Schedule, SCED*) використовується тільки на рівні проекту. Інші мультиплікативні параметри вартості, кожний з яких представлений в продукті як множники зусиль, і розмір застосовуються до окремих компонентів проекту. Модель може бути використана для оцінки зусиль для проекту, який має тільки один компонент або декількох компонентів.

Існують спеціальні рекомендації з використання *COCOMO II* в супроводі програмного забезпечення, деякі з них взяті з [12].

Параметр вартості *SCED* (необхідний графік розробки) не використовується в оцінці зусиль на супровід. Це тому, що цикл супроводу зазвичай має фіксовану тривалість.

Параметр вартості *RUSE* (можливість використання продукту в подальших розробках) не використовується в оцінці зусиль на супровід.

Це відбувається тому, що додаткові трудовитрат, необхідні для підтримки повторного використання компонент, приблизно врівноважуються зниженням витрат на супровід завдяки ретельному проектуванню дизайну, документації та тестування.

Параметр вартості *RELY* (необхідна надійність програмного забезпечення) має різний набір множників витрат на супровід. Для супроводу параметр вартості *RELY* залежить від значення необхідної надійності, при якому продукт був розроблений. Якщо продукт був розроблений з низькою надійністю, це потребує більше зусиль, аби виправити приховані недоліки. Якщо продукт був розроблений з дуже високою надійністю, трудовитрат, необхідне для підтримки цього рівня надійності, буде вище номінального значення.

Для розрахунку трудовитрат супроводження використовується наступні параметри вартості: *RELY – Required Software Reliability Cost Driver* (необхідна надійність ПЗ) – враховує міру виконання програмою задуманої дії протягом певного часу; *DATA – Database Size Cost Driver* (розмір тестових даних) – враховує вплив обсягу тестових даних на розробку продукту; CPLX – *Product Complexity Cost Driver* (складність продукту) – включає п'ять типів операцій: управління, рахункові, пристрій-залежні, управління даними, управління призначеним для користувача інтерфейсом. Рівень складності – це суб'єктивне середньо зважене значення рівнів типів операцій; *DOCU – Documentation Match to Life-cycle Needs Cost Driver* (повнота документації) – враховує ступінь відповідності документації проекту його життєвому циклу; *TIME – Execution Time Constraint Cost Driver* (обмеження з доступності програмного середовища) – враховує часові ресурси, використовувані ПЗ при виконанні поставленого завдання; *STOR – Main Storage Constraint Cost Driver* (обмеження пам'яті) – враховує відсоток використання сховищ даних; *PVOL – Platform Volatility Cost Driver* (розробка платформи) – враховує термін життя платформи (комплекс апаратного і програмного забезпечення, який потрібно для функціонування ПЗ, що розробляється); *ACAP – Analyst Capability Cost Driver* (кваліфікація аналітиків) – враховує аналіз, здатність проектувати, ефективність і комунікативні здібності групи фахівців, які розробляють вимоги і специфікації проекту. Параметр не повинен оцінювати рівень кваліфікації окремо взятого фахівця; *PCAP – Programmer Capability Cost Driver* (кваліфікація програмістів) – враховує рівень програмістів в колективі.

При виборі значення для цього параметра слід особливо звернути увагу на комунікативні та професійні здібності програмістів і на командну роботу в цілому; *PCON – Personnel Continuity Cost Driver* (проектна команда) – враховує текучість кадрів у колективі; *APEX – Applications Experience Cost Driver* (досвід роботи над застосуваннями) – враховує досвід колективу при роботі над застосуваннями певного типу; *PLEX – Platform Experience Cost Driver* (знання платформи) – враховує вміння використовувати особливості платформ, такі як графічний інтерфейс, бази даних, мережевий інтерфейс, розподілені системи; *LTEX – Language and Tool Experience Cost Driver* (знання мови і середовища розробки) – враховує досвід програмістів (мови, середовища та інструментів); *TOOL – Use of Software Tools Cost Driver* (середовище розробки) – враховує рівень використання інструментів розробки; *SITE – Multisite Development Cost Driver* (розподілена розробка) – враховує територіальну віддаленість (від офісу до міжнародних офісів) членів команди розробників і використовувані ними засоби комунікації (від телефону до відео конференц-зв'язку). Детально таблиці розрахунку значень параметрів вартості наведено в [13].

Таким чином, звернемо увагу, що масштабуючий показник *E* застосовується для числа змінених *KSLOC* (доданих та змінених, невидалених), для загальної кількості успадкованих *KSLOC* системи. Ефективний розмір супроводу $(Size)_M$ коригується Поправочним коефіцієнтом супроводу (*MAF*) для обліку ефектів успадкованої системи.

Формула оцінки трудовитрат на супроводження така ж, як в *COCOMO II* для Пост-архітектурної моделі розвитку (за винятком *SCED* та *RUSE*):

$$PM_M = A \times (SIZE_M)^E \times \prod_{i=1}^{15} EM_i \quad (7)$$

Підхід *COCOMO II* відрізняється від *COCOMO 81* стосовно оцінки витрат на супровід, дозволяючи використовувати будь-яку тривалість супроводу, *TM*. Середня чисельність персоналу супроводу (*FSPM*) може бути отримана за Рівнянням 8:

$$FSPM = PM_M/TM \quad (8)$$

Аналіз вітчизняної та зарубіжної літератури, а також діючих нормативних документів для визначення трудовитрат програмного забезпечення показує, що традиційні підходи, що засновані на підрахунку кількісних характеристик програми та застосуванні визначених

Розробником та Замовником схем або принципів, не є універсальними і не враховують всіх чинників та особливостей досліджуваної задачі.

Таким чином, можливо навести загальний алгоритм розрахунку трудовитрат супроводження ПЗ за моделлю *COCOMO II* (за результатом усвідомлення засад та підходів, які застосовуються у методиці), визначити послідовність дій експерта, спеціаліста (алгоритм дій) та обмеження.

Слід зазначити, що перед початком дій експерт (спеціаліст), який проводить оцінку та калібрування показників для даного проекту, визначається із обмеженнями (виходячи із засад методики *COCOMO II*), саме експерт має отримати *відомості*:

1. Концепція, стратегія, бізнес-план проекту КП;

Довідково. Наявність даних документів є необхідною, але не критичною для проведення дослідження. У випадку їх відсутності проведення дослідження вбачається можливим.

2. Вихідні коди (тексти) комп'ютерної програми у повному обсязі.

Довідково. Наявність даних є необхідною та критичною для проведення дослідження. У випадку їх відсутності проведення дослідження вбачається неможливим. У випадку дослідження багатомодульної/багатокомпонентної системи (комплексу), відомості мають бути представлені поелементно.

3. Об'єктні коди комп'ютерної програми у повному обсязі.

Довідково. У випадку наявності інсталяційного пакету КП на дослідження мають бути представлена коробкова версія із комплектом технічно-експлуатаційної документації.

Таким чином, *експерт* під час проведення дослідження:

– отримує та вивчає об'єкти та документацію, надані на дослідження;
– усвідомлює завдання;
– у разі неповного комплекту наданих на дослідження об'єктів та документації, направляє клопотання про надання додаткових матеріалів;
– проводить дослідження;
– проводить експерименти (інструментальне та програмне тестування);
– здійснює оцінку отриманих результатів, формулює висновки *(слід звернути увагу, що відповідно до [13] у разі нечіткого визначення стану масштабуючих факторів та множників витрат виставляється рівень оцінки номінального значення за замовченням);*
– оформлює Висновок;
– здійснює перевірку на обґрунтованість, повноту тощо.

Практика показує, що процес розробки програмного забезпечення досить складно піддається нормуванню з використанням традиційних формальних моделей, що при оцінюванні трудовитрат часто вимагає додаткового застосування експертних чи інтервальних підходів [6].

Висновки та перспективи подальших досліджень. Таким чином, в даній статті проаналізовано існуючі варіанти розрахунку трудовитрат супроводження програмного забезпечення інформаційних систем, обґрунтовано недоліки та переваги існуючих методів розрахунку, викладено алгоритм послідовності дій розрахунку на основі моделі конструктивних витрат *COCOMO II.*

Дану модель можливо застосовувати також для розрахунку трудовитрат під час розробки, модернізації, впровадження програмного забезпечення інформаційних (автоматизованих) систем. З боку Замовника є можливість орієнтованого розрахунку трудовитрат для результативного виконання робіт Розробником у визначені терміни при визначених показниках.

Перспективою подальших досліджень є задача розробки алгоритмів та методики розрахунку на основі конструктивної моделі витрат *COCOMO II* для визначення часових витрат на повний життєвий цикл програмного забезпечення (розробка, супроводження, модернізація тощо). Ступінь впровадження результатів знаходиться на початковому рівні.

ЛІТЕРАТУРА

1. Шафер Д., Фатрелл Р., Шафер Л. Управление программными проектами: достижение оптимального качества при минимуме затрат.: Пер. с англ. – М.: „Вильямс”, 2003. – 1136 с.

2. Колдовський В.В. Управління інноваціями на етапах життєвого циклу програмного забезпечення / дисертація – Суми: Українська академія банківської справи. – 2005. – 184 с.

3. Сопровождение. Технология разработки программного обеспечения: [Електронний ресурс]. – Режим доступу: http://project.dovidnyk.info/index.php/home/tehnologiyarazrabotkiprogrammnogoobespecheniya/27-soprovozhdenie.

4. ДСТУ 3918-1999 (ISO/IEC 12207:1995). Інформаційні технології: Процеси життєвого циклу програмного забезпечення. – Введ. 01-07-2000. – К., 2000. – 49 с.

5. ДСТУ ISO/IEC 14764:2002. Інформаційні технології. Супровід програмного забезпечення. (ISO/IEC 14764:1999, IDT). – Введ. 01-07-2004. – К., 2004. – 36 с.

6. В.О. Гороховатський, В.Ю. Дубницький, А.М. Кобилін, В.О. Лукін, О.В. Москаленко Визначення трудомісткості при розробленні програмних комплексів // Системи обробки інформації, 2014. – Випуск 2 (118). – С. 92 – 98.

7. Мандрикова Л.В. Методы оценки стоимости и затрат на создание программного продукта, основанные на нечеткой логике / Л.В. Мандрикова, Ю.С. Манжос, П.А. Лучшев // Радіоелектронні і комп’ютерні системи. – 2008. – № 2 (29). – С. 115 – 118.

8. Алиев Х.Р. Модель планирования и управления разработкой сложных программных систем на основе комбинированной методики оценки трудозатрат // Автореферат. – Санкт-Петербург. – 2010. – 25 с. [Електрон. ресурси] / Режим доступу: // http://www.spbu.ru/files/upload/disser/eco/2010/Aliev.pdf.

9. Технико-экономическое обоснование стоимости программных систем / Методическое пособие / В.Т. Калайда. – Томск: ТУСУР, 2009. – 50 с.

10. Boehm B.W. Software engineering economics / B.W. Boehm. – Prentice–Hall, 1981. – 320 p.

11. Shepperd M. Estimating software project effort using analogy / M. Shepperd, C. Schofield // IEEE Trans Software Eng. – 1997. – P. 736 – 743.

12. Boehm B.W. The COCOMO 2.0 Software Cost Estimation Model / B.W. Boehm. – American Programmer, 2000. – 586 p.

13. COCOMO II.2000.0: CSE, 1999: Center for Software Engineering. COCOMO II Reference Manual. // Computer Science Department, USC Center for Software Engineering, 1999. – 86 p.: [Електронний ресурс]. – Режим доступу: http://csse.usc.edu//TECHRPTS/2000/usccse2000- 500/usccse2000-500.pdf.