УДК 004.8

Т. О. Савчук[1], Н. В. Приймак[1]

РОЗРОБКА ІНФОРМАЦІЙНОЇ МОДЕЛІ ПРОЦЕСУ ПОШУКУ АСОЦІАТИВНИХ ПРАВИЛ ПРИ РОЗРОБЦІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

[1]Вінницький національний технічний університет

Анотація. У роботі сформовано задачу пошуку асоціативних правил при розробці програмного забезпечення, а також запропоновано інформаційну модель процесу пошуку асоціативних правил при розробці програмного забезпечення. При цьому, розглянуто моделі розробки програмного забезпечення, їх переваги та недоліки; вказано основні етапи даного процесу; обгрунтовано доцільність пошуку асоціативних правил при розробці програмного забезпечення, з метою знаходження залежностей, що можуть бути використані для визначення часу необхідного для виправлення завдання певним розробником. Менеджери проекту можуть використовувати дану інформацію для планування та управління процесом розробки програмного забезпечення. Доведено доцільність використання запропонованої інформаційної моделі процесу пошуку асоціативних правил при розробці програмного забезпечення. Результати проведеного дослідження показали, що інформативність асоціативних правил зросла, а тривалість їх пошук скоротилася. Розроблена інформаційна модель процесу пошуку асоціативних правил при розробці програмного забезпечення може бути використана при розробці відповідної інформаційної технології.

Ключові слова: програмне забезпечення; інформаційна модель; асоціативні правила; Data Mining.

Аннотация. В работе сформирована задача поиска ассоциативных правил при разработке программного обеспечения, а также предложена информационная модель процесса поиска ассоциативных правил при разработке программного обеспечения. При этом, рассмотрены модели разработки программного обеспечения, их преимущества и недостатки; указаны основные этапы данного процесса; обоснована целесообразность поиска ассоциативных правил при разработке программного обеспечения, с целью нахождения зависимостей, которые могут быть использованы для определения времени, необходимого для исправления задания определенным разработчиком. Менеджеры проекта могут использовать данную информацию для планирования и управления процессом разработки программного обеспечения. Доказана целесообразность использования предложенной информационной модели процесса поиска ассоциативных правил при разработке программного обеспечения. Результаты проведенного исследования показали, что информативность ассоциативных правил выросла, а продолжительность их поиск сократилась. Разработанная информационная модель процесса поиска ассоциативных правил при разработке программного может быть использована при разработке соответствующей информационной технологии.

Ключевые слова: программное обеспечение; информационная модель; ассоциативные правила; Data Mining.

Abstract. The problem of associative rules search during software development is formed in the work, and the informational model of the process of associative rules search during software development is proposed. In this paper, the models of software development, their advantages and disadvantages are considered; the main stages of this process are specified; the expediency of associative rules search during the software development was substantiated in order to find dependencies that can be used to determine the time required by a particular developer to correct a task. Project managers can use this information to plan and manage the software development process. The expediency of using the proposed information model of associative rules search during software development is proved. The results of the research showed that the informative of the associative rules has increased and the search time has decreased. The developed information model of associative rules search during software development can be used in the development of relevant information technology.

Keywords: software; information model; associative rules; Data Mining.
DOI: https://doi.org/10.31649/1999-9941-2018-42-2-43-48.

Вступ

Програмне забезпечення (ПЗ) – це набір комп'ютерних програм, представлених в цифровому вигляді, що використовуються для вирішення задач певного класу [1]. В свою чергу комп'ютерна програма – це упорядкована послідовність інструкцій, створених для досягнення поставлених цілей [1]. Процес розробки програмного забезпечення – це спланований та визначений процес створення комп'ютерних програм [2].

Актуальність

Процес розробки програмного забезпечення може бути здійснений з використанням різноманітних технологій розробки, в основу яких покладено такі моделі [2]:

1. Waterfall Model (каскадна модель) – модель, під час використання якої відбувається послідовна реалізація усіх стадій розробки, кожна з яких має повністю завершитися перед початком наступної. Дана модель зручна при керуванні невеликим проектом, тому що розробка ПЗ проходить швидко, вартість і терміни закінчення розробки заздалегідь визначені. Недоліком даної моделі є те, що вона може бути застосована лише при розробці ПЗ у якого технічне завдання не зазнаватиме змін у процесі розробки, що майже неможливо під час розробки програм.

2. V – подібна модель застосовується при розробці ПЗ для якого важливе безперебійне функціонування, наприклад програми координування безперервних технологічних процесів на виробництвах та атомних реакторах. Особливістю даної моделі є те, що тестування програми проводиться одночасно з відповідною стадією розробки: під час аналізу вимог, здійснюється перевірка на відсутність логічних помилок в діях користувачів, розглядаються всі можливі способи використання даного ПЗ, а під час кодування пишуться модульні тести. Недоліком є використання ресурсів на паралельне тестування.

3. Інкрементна модель передбачає кілька циклів розробки програмного забезпечення, що утворює життєвий цикл «мульти-каскад». Процедура розробки ПЗ згідно даної моделі передбачає випуск на першому етапі продукту в базовій функціональності, а потім послідовне додавання нових функцій, так званих «інкрементів». Серед недоліків даної моделі можна виділити: необхідність планування та дизайну ПЗ, що розробляється, чіткого і повного визначення усієї програми.

4. Ітеративна модель передбачає створення частини функціоналу, що стає базою для визначення подальших вимог. Для успішного використання ітеративної моделі розробки програмного забезпечення потрібно здійснювати перевірку вимог до кожної версії програмного забезпечення в рамках кожного циклу моделі. Недоліками даної моделі є необхідність активного управління процесом розробки ПЗ, неможливість визначення точної дати завершення розробки, необхідність передбачення та аналізу можливих ризиків.

5. Agile model (гнучка модель розробки) характеризується тим, що після кожної ітерації процесу розробки ПЗ, замовник може спостерігати результат і розуміти, задовольняє він його чи ні. Серед недоліків даної моделі виділяють відсутність сформульованого очікуваного результату та складність оцінки трудовитрат і вартості розробки.

Порівняльний аналіз моделей процесу розробки програмного забезпечення наведено у табл. 1.

Таблиця 1 – Моделі процесу розробки програмного забезпечення

Назва моделі	Переваги	Недоліки
Каскадна модель	Зручна для створення невеликого проекту	Можна застосовувати лише при розробці ПЗ із технічним завданням, яке не змінюватиметься у процесі розробки
V – подібна модель	Кожен етап розробки ПЗ тестується відповідними засобами	Необхідно використовувати ресурси на паралельне тестування
Інкрементна модель	Можна побачити завершений базовий функціонал програмного продукту після першої ітерації розробки	Необхідно здійснювати планування та дизайн ПЗ, чітко і повно визначити програму, перш ніж вона буде розроблена
Ітеративна модель	Під час кожної ітерації розробки ПЗ, здійснюється перевірка його вимог та версій	Необхідно активно управляти процесом розробки ПЗ; неможливо визначити точну дату завершення розробки; необхідно провести аналізі можливих ризиків
Agile model	Після кожної ітерації розробки, замовник може спостерігати результат і розуміти, задовольняє він його чи ні	Відсутність сформульованого очікуваного результату; складність в оцінці трудовитрат і вартості розробки

Під час розробки ПЗ накопичується велика кількість інформації, результати аналізу якої, можна використовувати для управління та удосконалення процесу розробки ПЗ. Для аналізу збереженої інформації доцільно застосовувати технології Data mining, що розроблені з метою отримання корисних (з точки зору експерта) даних, залежностей, шаблонів тощо зі збереженої інформації.

Тому, актуальною є задача розробки інформаційної моделі пошуку асоціативних правил під час розробки програмного забезпечення. Знайдені асоціативні залежності можна використовувати при плануванні, передбаченні та розумінні процесу розробки програми, а також для підтримки та управління даного процесу.

Мета

Метою даного дослідження є розробка інформаційної моделі процесу пошуку асоціативних правил при розробці програмного забезпечення, використання якої дозволить скоротити час пошуку асоціативних правил (АП) та підвищити їх інформативність.

Задачі

1. Сформувати задачу пошуку асоціативних правил при розробці програмного забезпечення
2. Розробити інформаційну модель процесу пошуку асоціативних правил під час розробки програмного забезпечення

Розв'язання задач

Нехай Dev_j – це j-ий розробник, $Task_i$ – це i-те завдання, що має ряд характеристик та буде ви-

рішене j-м розробником, а t_{i_j} – час, необхідний на розв'язання i-го завдання j-им розробником.

Задачу пошуку асоціативних правил при розробці програмного забезпечення можна описати наступним чином:

$$Task_i \cup Dev_j \rightarrow t_{i_j},\ Task_i \cup Dev_j \cap t_{i_j} \rightarrow \varnothing$$

Тобто, необхідно знайти асоціативні правила виду: «якщо завдання $Task_i$ буде розв'язуватися розробником Dev_j, то йому потрібно буде t_{i_j} часу».

Визначимо інформаційну модель процесу пошуку асоціативних правил при розробці програмного забезпечення, що відображає вхідні та вихідні значення даного процесу, важливі параметри і величини, а також їхні зв'язки.

Відповідну інформаційну модель можна подати у вигляді кортежа $IMARM$:

$$IMARM = \langle Task, Dev, minsupp, minconf, method, AR, newTask, PredictTask \rangle, \quad (1)$$

де $Task$ – множина завдань, що були виконані під час розробки ПЗ; Dev – множина розробників, які можуть виконати поставлене завдання; $minsupp$ – мінімальне значення підтримки АП; $minconf$ – мінімальне значення достовірності АП; $method$ – метод, за допомогою якого буде здійснюватися пошук АП; AR – множина знайдених асоціативних правил; $newTask$ – множина завдань, для яких необхідно визначити тривалість їх реалізації певним розробником; $Predict\ Task$ – множина завдань з часом, необхідним на їх виконання певним розробником.

Нехай i-те завдання $Task_i$ описується:

$$Task_i = \langle Type, Priority, Severity, Component, t_{i_j}, Dev_j \rangle, \quad (2)$$

де $i = \overline{1,n}$, n – кількість завдань; $Type$ – це тип i-го завдання, що має наступну множину значень {покращення, особливість, дефект}. В залежності від типу завдання, воно виконується на конкретному етапі розробки ПЗ. $Priority$ – це пріоритет i-го завдання, що має наступну множину значень {високий, середній, низький} та описує важливість і порядок, в якому потрібно виконати завдання у порівнянні з іншими. $Severity$ – це важливість i-го завдання, що має наступну множину значень {блокуюча, критична, важлива, помірна, незначна}. Дана характеристика визначає рівень впливу даного завдання на функціонування ПЗ в цілому. Даний показник важливий для менеджерів проектів, оскільки він є основним при вирішенні бізнес-питань. $Component$ – це компонент розроблюваного ПЗ, множина значень якого залежить від конкретного ПЗ, тобто це частина програмного забезпечення, для якої буде виконуватися завдання певного типу. t_{i_j} – це час, необхідний для виконання i-го завдання j-м розробником. Значення даної величини залежить від конкретного завдання та розробника, що буде його виконувати. Менеджери проекту повинні знати величну даної характеристики для планування процесу розробки ПЗ та ресурсів, необхідних для нього.

Знайдені асоціативні правила при розробці ПЗ утворюють множину AR. Нехай d-те асоціативне правило описується:

$$AR_d = \{ Task_i \cup Dev_j \rightarrow t_{i_j},\ Task_i \cup Dev_j \cap t_{i_j} \rightarrow \varnothing \}, \quad (3)$$

де $d = \overline{1,c}$, d – кількість знайдених асоціативних правил.

Множину $newTask$, утворюють завдання, які потрібно виконати, під час розробки ПЗ і для яких не визначено тривалість їх виконання. Нехай f-те завдання $newTask_f$ описується:

$$newTask_f = \langle Type, Priority, Severity, Component \rangle, \quad (4)$$

де $f = \overline{1,g}$, g – кількість завдань.

Множина $Predict\ Task$, складається із завдань, для яких визначено тривалість їх виконання. Нехай l-те завдань з часом, необхідним на його виконання певним розробником $Predict\ Task_l$ описується:

$$PredictTask_l = \left\langle Type, Priority, Severity, Component, Dev_j, t_{l_j} \right\rangle. \quad (5)$$

де $l = \overline{1,k}$, k – кількість завдань; t_{l_j} – це тривалість виконання l-го завдання j-им розробником.

Запропонована інформаційна модель процесу пошуку асоціативних правил при розробці програмного забезпечення розділена на два окремих блоки (рис. 1):

1. Блок пошуку асоціативних правил – де описано процес пошуку асоціативних правил.

Результатом виконання даного процесу є множина асоціативних правил AR, що використовуються для визначення часу, необхідного на виконання завдання конкретним розробником у наступному блоці.

Для пошуку асоціативних правил при розробці ПЗ використовується удосконалений алгоритм Frequent Pattern Growth (FPG) [3, 4], що був модифікований за рахунок класифікації завдань на три групи в залежності від складності їх виконання. Така модифікація дозволяє пришвидшити пошук АП, оскільки він здійснюватиметься паралельно в створених групах, та підвищити їх інформативність.

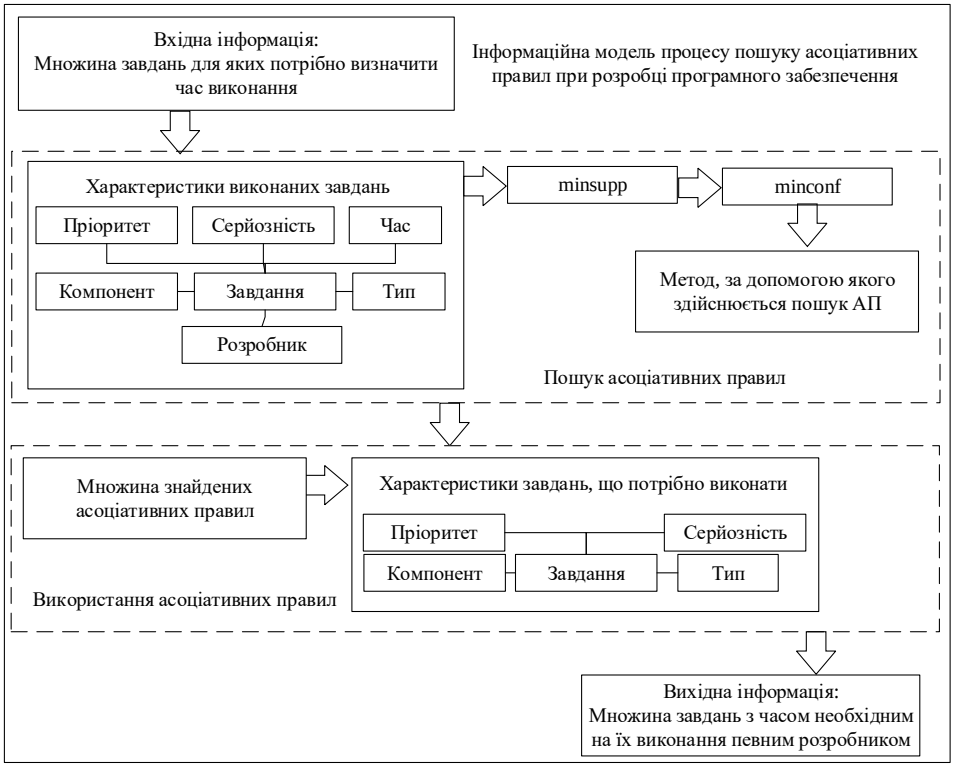

Рисунок 1 – Інформаційна модель процесу пошуку АП під час розробки програмного забезпечення

Значення мінімальної підтримки $minsupp$ і мінімальної достовірності $minconf$ є обов'язковими при пошуку асоціативних правил і задається експертом [5]. Неправильно визначена величина даних параметрів, може призвести до невірної генерації АП: при встановленій високій мінімальній підтримці може бути згенеровано лише кілька правил, але багато часу буде витрачено на сканування бази даних. Вибір низької мінімальної підтримки може призвести до численних надлишкових правил. Тому виникає необхідність у виборі оптимального значення для даних показників.

Значення підтримки асоціативного правила $X \to Y$ дорівнює значенню кількості транзакцій в яких X і Y зустрічаються разом [6].

Для визначення значення мінімальної підтримки $minsupp$ для пошуку АП під час розробки ПЗ за-

стосуємо функцію, відображену у виразі 1.6 [7]. Ідея використання цієї функції для пошуку частих предметних наборів була вперше запропонована в [8], що дозволяє встановити високе значення мінімальної підтримки, коли даних в БД небагато, а потім зменшувати його, у випадку збільшення кількості даних:

$$minsupp = (e^{(-\ ax-\ b)}) + c, \tag{6}$$

де x – це кількість записів в БД; a, b, c – додатні константи. Константа c – це найменше значення, якому може дорівнювати достовірності асоціативного правила. Константи a та b впливають на те, як різко повинна зменшуватися $minsupp(x)$, коли x збільшиться.

2. У наступному блоці інформаційної моделі процесу пошуку АП під час розробки ПЗ відображено процес використання асоціативних правил, з метою визначення часу, необхідного на виконання завдання конкретним розробником.

Результатом даного процесу є множина завдань $Predict\ Task_l$ з часом, необхідним на їх реалізацію певним розробником.

Оскільки процес розробки ПЗ розглядається з точки зору менеджера проекту, то необхідно вирішити задачу прийняття рішень щодо відбору завдань, які можна реалізувати у визначені терміни та з використанням вказаного бюджету.

Такий план можна представити кортежом $PLAN$ вираз 1.7:

$$PLAN = \langle PredictTask, Budget, Duration, Fault \rangle, \tag{7}$$

де $Budget$ – бюджет, виділений на розробку ПЗ. Це максимальна сума коштів, що може бути витрачена на розробку даного програмного продукту чи його складової. Величина даної характеристики відрізняється в залежності від проекту. $Duration$ – запланована тривалість розробки ПЗ. Величина даної характеристики також залежить від проекту. $Fault$ – допустиме відхилення тривалості та вартості розробки, що встановлюється менеджером проекту, в залежності від проекту.

Ефективність використання розробленої інформаційної моделі процесу пошуку асоціативних правил при розробці програмного забезпечення було підтверджено результатами проведеного експерименту.

Експеримент проводився з наступною тестовою інформацією:

- кількість завдань – до 100 елементів, до 500 елементів, до 1000 елементів, до 1500 елементів;
- кількість знайдених асоціативних правил;
- час, витрачений на їх пошук в секундах.

Результати експерименту з використанням запропонованої інформаційної моделі та без неї представлені на рис. 2:

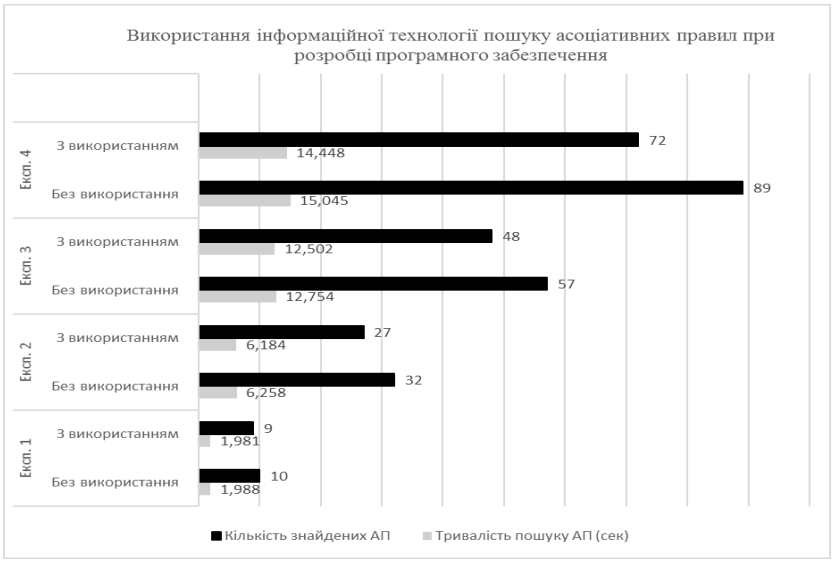

Рисунок 2 – Результати використання інформаційної моделі процесу пошуку АП під час розробки ПЗ

1. Зі збільшенням кількості завдань, серед яких здійснювався пошук АП, збільшується час, необхідний для їх пошуку; але різниця між часом, необхідним на пошук асоціативних правил з використанням розробленої інформаційної моделі та без неї, збільшується зі збільшенням загальної кількості тестових даних.

2. Швидкість пошуку АП з використанням запропонованої інформаційної моделі зросла наступним чином: кількість елементів до 100 – зросла на 0,35%, кількість елементів до 500 – на 1,2%, кількість елементів до 1000 – на 2%, кількість елементів до 1500 – на 3,96%.

3. Інформативність та цінність знайдених асоціативних правил при використанні запропонованої інформаційної моделі вища, оскільки кількість таких АП менша ніж без використання відповідної інформаційної моделі.

Висновки

1. Сформовано задачу пошуку асоціативних правил під час розробки програмного забезпечення,

2. Розроблено інформаційну модель даного процесу, що може бути використана при розробці відповідної інформаційної технології. Використання запропонованої інформаційної моделі дозволило скоротити час пошуку асоціативних правил та підвищити їх інформативність, що підтверджено отриманими результатами проведених експериментів.

Список літератури

[1] В. К. Батоврин, Толковый словарь по системной и программной инженерии: М: ДМК Пресс, 2012.

[2] Application Development, 2014. [Online]. Available: http://www.bestpricecomputers.co.uk/glossary/application-development.htm.

[3] Т. О. Савчук, Н. В. Приймак, «Використання fpg-алгоритму для пошуку асоціативних правил при прийнятті рішень в управлінні процесами», на XLVI НТК професорсько-викладацького складу, співробітників та студентів університету, Вінниця, 2017.

[4] Т. О. Савчук, Н. В. Приймак, «Обгрунтування вибору методу генерації частих предметних наборів для пошуку асоціативних правил при розробці прогрмного забезпечення», на XI міжнародній конференції «Інтернет-освіта-наука-2018», Вінниця, 2018, с.45-46.

[5] Ассоциативные правила, 2016 [Електронний ресурс]. Режим доступу: https://www.researchgate.net/publication/315629798_AssociativnyepravilSravnitelnyjnalinstrumentaria.

[6] J. Manimaran, "Analysing the quality of Association Rules by Computing an Interestingness Measures", Indian Journal of Science and Technology, Vol 8(15), pp. 1-12, July. 2015. DOI: 10.17485/ijst/2015/v8i15/76693

[7] P. Fournier-Viger, How to auto-adjust the minimum support threshold according to the data size, 2014. [Online]. Available: http://data-mining.philippe-fournier-viger.com/how-to-auto-adjust-the-minimum-support-threshold-according-to-the-data-size/.

[8] P. Fournier-Viger, "Un modèle hybride pour le support à l'apprentissage dans les domaines procéduraux et mal-définis". Ph.D. Thesis, University of Quebec in Montreal, Montreal, 2010.

Стаття надійшла: 28.08.18.

Відомості про авторів

Савчук Тамара Олександрівна – PhD, професор кафедри комп'ютерних наук Вінницького національного технічного університету.

Приймак Наталія Василівна – аспірант кафедри комп'ютерних наук Вінницького національного технічного університету.

T. O. Savchuk[1], N. V. Pryimak[1]

DEVELOPMENT OF THE INFORMATION MODEL OF THE ASSOCIATIVE RULES SEARCH DURING THE SOFTWARE DEVELOPMENT

[1]Vinnytsia National Technical University