УДК 004.413

ВИКОРИСТАННЯ МЕРЕЖ ПЕТРІ В УПРАВЛІННІ ЖИТТЄВИМ ЦИКЛОМ БЕЗПЕКО-ОРІЄНТОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Юлія Назар
Олександр Придатко, кандидат технічних наук, доцент
Львівський державний університет безпеки життєдіяльності

У даній роботі аргументовано невідповідність та недієвість сучасних методів управління життєвим циклом програмного забезпечення у контексті розробки безпеко-орієнтованих сервісів. На основі чого побудовано модель процесу розробки програмних систем безпеко-орієнтованого спрямування із використанням мереж Петрі. Розроблена модель дозволяє побачити загальну послідовність процесів управління життєвим циклом програмного забезпечення та їх взаємозв'язки.

Ключові слова: безпеко-орієнтована система, мережі Петрі, життєвий цикл програмного забезпечення

THE USE OF PETRI NETS IN THE LIFE CYCLE MANAGEMENT OF SECURITY-ORIENTED SOFTWARE

Yuliia Nazar
Oleksandr Prydatko, Candidate of Technical Sciences, Associate Professor
Lviv State University of Life Safety

This paper argues the inconsistency and ineffectiveness of modern methods of software life cycle management in the context of developing security-oriented services. On this basis, a model of the process of developing security-oriented software systems using Petri nets is built. The developed model allows to present the general sequence of software life cycle management processes and their interrelationships.

Keywords: safety-oriented system, Petri nets, software life cycle

Відомо, що оцінювання тривалості розробки програмного забезпечення у динамічних умовах є вкрай невизначеним, проте дуже важливим етапом у процесі розробки нового програного продукту. Змінні вимоги, обмежені часові та матеріальні ресурси, управління ризиками та якістю – всі ці умови мають свій вплив на термін впровадження та випуск нового програмного забезпечення. Станом на сьогоднішній день існує два класичних підходи (методології) в управлінні процесом розробки програмного забезпечення – гнучкий та каскадний, кожен із них володіє своїми принципами та обмеженнями. Зокрема при гнучкій розробці фіксованими є команда розробників та час виконання, а вимоги – змінні.

При каскадній – навпаки, фіксованим є чіткий перелік робіт (вимог до продукту), а команда і час можуть змінюватись [1].

Проте, під час розробки безпеко-орієнтованих сервісів, якою займаються працівники Державної служби України з надзвичайних ситуацій, було виявлено проблему невідповідності та недієвості даних методів в управлінні ЖЦ СПЗ, оскільки в даному випадку динамічність процесів розробки характеризується не лише обсягом і змістом робіт, а також часовим ресурсом. Саме тому існує потреба у розробці нових методів та підходів до оцінки тривалості розробки безпеко-орієнтованих сервісів, адаптованих під специфіку роботи Державної служби України із надзвичайних ситуацій та корелюють із принципами гнучкої методології управління життєвим циклом програмного забезпечення [3].

Метою роботи є розробка адаптивного методу короткострокового планування життєвого циклу програмних систем безпеко-орієнтованого спрямування у динамічному оточенні із використанням мережі Петрі.

Процес планування розробки програмного забезпечення складається із визначення вимог до продукту у вигляді користувацьких історій. В Agile-методології не прийнято розбивати вимоги на технічні завдання, оскільки такий підхід не дозволяє цілісно поглянути на виконання певного функціоналу програми. Користувацька історія – це короткий та простий опис характеристик продукту з точки зору користувача, який прагне нових можливостей. Вона включає в себе весь спектр розробки: від проектування дизайну і до тестування. Такий підхід дозволить всій команді приймати участь у процесі оцінки та розробки даної функціональності. Після визначення, користувацька історія подрібнюється на дрібні під задачі (функції), які будуть виконувати безпосередньо різні учасники команди. Від так, зрозуміло, що існує велика ймовірність, що декілька робіт можуть виконуватись паралельно (що дозволить значно зекономити час), а також можуть бути роботи, виконання яких залежать від попередніх. Саме тому, визначення критичного шляху, який забиратиме найбільше часу на виконання, надзвичайно важливе. Графічний опис процесу планування розробки програмного забезпечення зображений на рисунку 1.

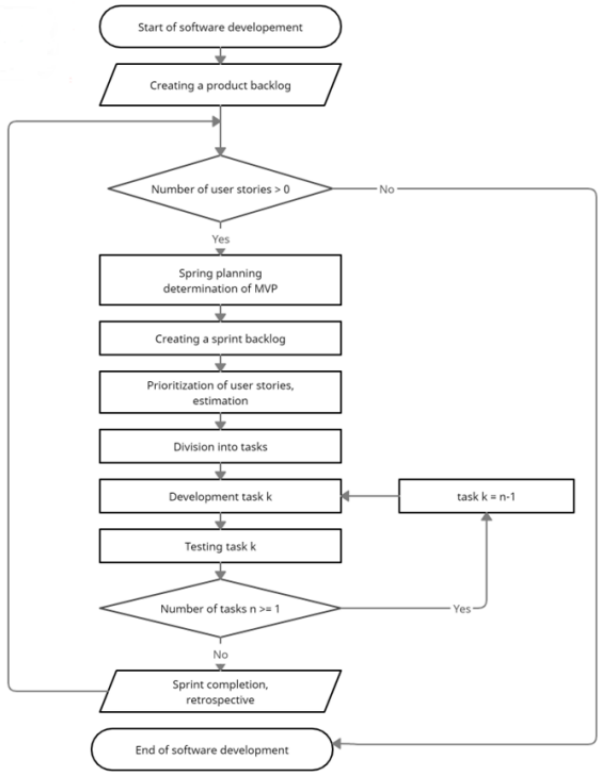

Рисунок 1 – Процес планування розробки програмного продукту [4].

З даної блок-схеми можна зробити висновок, що процес розробки програмного забезпечення у динамічних умовах являє собою високонавантажену розподілену паралельну систему. Для формування чіткої уяви про організаційно-технічні процеси, які виникають в подібних системах необхідно провести моделювання паралельних процесів. Чудовим інструментарієм для виконання такого роду завдань є апарат моделювання мережами Петрі. Мережа Петрі дозволяє відобразити певні стани модельованої системи, де для переходу між різними станами системи необхідне обов'язкове виконання умов (вони закладаються в мережу завчасно).

Представимо процес розробки спеціалізованих програмних систем як дискретну систему, переходами між станами якої є реалізація окремих частин функціоналу програмного продукту (рисунок 2).

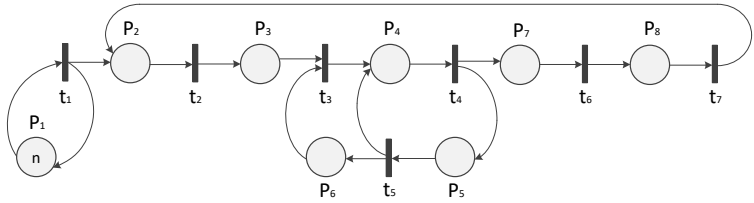

Рисунок 2 – Модель процесу розробки програмних систем безпеко-орієнтованого спрямування (масштабована до спринта) [4]

Опишемо представлену на рисунку 2 модель. Позиція P1 передбачає формування беклогу продукту, який передбачає перелік усіх завдань для успішної реалізації проекту. Перехід t1 свідчить про те, що команда розробників отримала визначений перелік завдань з беклогу. Позиція P2 передбачає планування окремого спринта, умовою для виконання якої є відібрані завдання на спринт t2. Позиція P3 орієнтована на формування беклогу спринта із одночасним розподілом завдань між учасниками проектної команди. Успішно розподілені завдання позиціонуються переходом t3. Безпосереднє виконання завдань спринта щодо розробки програмної системи реалізується в межах позиції P4. Виконані завдання в межах переходу t4 можуть мати розгалуження на тестування складових P5 та спрінт рев'ю P7. Протестований фрагмент системи t5 , за умови якісного виконання завдань та відсутності порушень вимог, повертається до позиції P4. За умови виявлення невідповідності, модель системи передбачає перехід на позицію P6, умовою якої є виправлення помилок та доопрацювання вимог. Позиція спірнт рев'ю P7 завершується успішним прийняття виконаних завдань t6, а після завершення ретроспективи P8 завершення спринта фіксується переходом t7. Після виконання описаного переліку умов та переходів дискретно-циклічна система передбачає планування та початок нового спринта із переліком нових завдань в позиції P2. Обсяг та зміст завдань обираються в результаті спрацювання позиції P1 та переходу t1, які також володіють характеристикою циклічності для визначення нового обсягу робіт на спринт.

Здійснити дослідження змін у різних станах системи (описаної мережею) можливо шляхом комбінування матриць. Перша матриця «позитивна» вказує які саме процеси є похідними після виконання певної події (переходу). Такі процеси передують наступній позиції – умові виконання переходу.

$$R^{+} = \begin{pmatrix} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 \\ P_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ P_3 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ P_5 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ P_6 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ P_7 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ P_8 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \quad (1)$$

$$R^{-} = \begin{pmatrix} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 \\ P_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ P_3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ P_5 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ P_6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ P_7 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ P_8 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (2)$$

Друга «негативна» матриця вказує на те, яка саме умова (позиція) передує визначеному переходу.

Для подальшого моделювання станів системи необхідно знайти різницю матриць. Побудова матриці R здійснюється у результаті віднімання від R+ матриці R-.

Результат різниці матриць представлено виразом:

$$R = \begin{pmatrix} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 \\ P_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_2 & 1 & -1 & 0 & 0 & 0 & 0 & 1 \\ P_3 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ P_4 & 0 & 0 & 1 & -1 & 1 & 0 & 0 \\ P_5 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ P_6 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ P_7 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ P_8 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix}. \quad 3)$$

Матриця (3) дає підґрунтя для подальших досліджень працездатності системи в різних її станах. Для цього необхідно визначати добуток матриці та вектору, який описує стан системи у визначений проміжок часу. Проте при моделюванні працездатності системи слід враховувати умови виконання паралельних переходів.

За результатами даної роботи можна зробити висновок, про невідповідність сучасних методів та підходів управління життєвим циклом програмного забезпечення до розробки безпеко-орієнтованих сервісів. Оскільки окрім змінних вимог принципово важливим є час виконання. Розроблена мережа Петрі дозволяє побачити загальну послідовність процесів та їх взаємозв'язки, що, власне є предметом подальших досліджень.

Список літератури

1. Кордунова Ю. С., Смотр О. О., Кокотко І. Я., Малець Р. Б. Аналіз традиційного та гнучкого підходів до створення програмного забезпечення в динамічних умовах. Управління розвитком складних систем. Київ, 2021. № 47. С. 71 – 77, https://doi.org/10.32347/2412-9933.2021.47.71-77

2. Hovorushchenko T., Herts A.; Hnatchuk Y. Concept of Intelligent Decision Support System in the Legal Regulation of the Surrogate Motherhood. International Workshop on Informatics & Data-Driven Medicine, 2019, pp: 57-68.

3. Kordunova Y., Prydatko O., Smotr O., Golovatyi R. Expert Decision Support System Modeling in Lifecycle Management of Specialized Software. Lecture Notes on Data Engineering and Communications Technologies, Springer, Switzerland. Vol. 149, 2022, pp. 367-383, https://doi.org/10.1007/978-3-031-16203-9_22

4. Kordunova Yu., Prydatko O., Smotr O., Kokotko I. The network graph traversal method for solving the problem of short-term planning of safety-oriented services development. Monografia powstała w ramach Projektu dofinansowanego przez Ministra Edukacji i Nauki ze środków budżetu państwa w ramach programu „Doskonała Nauka", Warszawa 2022. p. 172 – 181.

References

1. Kordunova, Yu., Smotr, O., Kokotko, I. & Malets, R. (2021). Analysis of the traditional and flexible approaches to creating software in dynamic conditions. Management of Development of Complex Systems, 47, 71 77, https://doi.org/10.32347/2412-9933.2021.47.71-77 [in Ukrainian]

2. Hovorushchenko T., Herts A., Hnatchuk Y. Concept of Intelligent Decision Support System in the Legal Regulation of the Surrogate Motherhood. International Workshop on Informatics & Data-Driven Medicine, 2019, pp: 57-68.

3. Kordunova, Y., Prydatko, O., Smotr, O. & Golovatyi, R. (2023). Expert Decision Support System Modeling in Lifecycle Management of Specialized Software. Lecture Notes in Data Engineering, Computational Intelligence, and Decision Making. ISDMCI 2022. Lecture Notes on Data Engineering and Communications Technologies, 149, https://doi.org/10.1007/978-3-031-16203-9_22

4. Kordunova Yu., Prydatko O., Smotr O., Kokotko I. The network graph traversal method for solving the problem of short-term planning of safety-oriented services development. Monografia powstała w ramach Projektu dofinansowanego przez Ministra Edukacji i Nauki ze środków budżetu państwa w ramach programu „Doskonała Nauka", Warszawa 2022. p. 172 – 181.