УДК 004.7 **doi: 10.26906/SUNZ.2019.1.116**

Д. О. Ільїн, С. Г. Семенов

Національний технічний університет «Харківський політехнічний інститут», Харків, Україна

АНАЛІЗ ТА ПОРІВНЯЛЬНЕ ДОСЛІДЖЕННЯ АРХІТЕКТУРНИХ РІШЕНЬ ПРОГРАМНИХ РОБОТІВ ДЛЯ ПРОЦЕСІВ З ВЕЛИКОЮ КІЛЬКІСТЮ ТРАНЗАКЦІЙ

Аналіз нової області в сфері інформаційних технологій - RPA показав одну з актуальних завдань генерації рішення автоматизації бізнес процесів. Це визначення архітектурного рішення, яке визначає функціонування робота. Аналіз вище наведених прикладів показав переваги і недоліки кожної з схем рішень в ситуації автоматизації процесу з великою кількістю транзакцій. Розкрита сутність та актуальність теми RPA, а також існуючих архітектурних рішень програмних роботів та представлені переваги та недоліки в існуючих рішеннях в ситуації процесів з великою кількістю транзакцій. В висновку запропоноване можливе вирішення проблеми з виділенням переваг та недоліків. Якщо розглядати шаблон Multiple Instances Requiring Synchronization, то архітектурне рішення підпроцесу можна описати подібно вище наведеної машині станів, таким чином взявши властивість стійкості до збоїв, але при цьому робота кожної транзакції буде незалежна один від одного, тобто паралельне виконання. Таким чином, з'являється можливість зменшити час виконання процесу і зникає дублювання виконання елементів архітектурного рішення всього процесу. Але варто враховувати, що кожен екземпляр B (описаний як машина станів) повинен бути незалежний один від одного (тобто не перебувати в послідовності). Недоліком природно є збільшення ресурсів на виконання, тобто збільшення кількості локальних машин, на кожній з яких виконується одна транзакція. Тому такий підхід доречний для великих процесів з великою кількістю транзакцій.

Ключові слова: RPA, програмні роботи, архітектурні рішення, патерни архітектурних рішень.

Вступ

Щоб звільнити співробітників для виконання більш цінною роботи, IT-директора все частіше звертаються до методів роботизованої автоматизації процесів. Але, щоб така автоматизація приносила реальну користь бізнесу, необхідно грамотне проектування, планування і керівництво. Сьогодні IT-директора активно беруть на озброєння нову методику, що отримала назву «роботизованою автоматизації процесів» (robotic process automation, RPA). Вони розраховують з її допомогою оптимізувати діяльність компанії і зменшити витрати. RPA автоматизує повсякденні бізнес-процеси, описувані правилами, дозволяючи співробітникам приділяти більше часу обслуговування клієнтів і іншим більш цінним завданням [1]. Експерти вважають RPA тимчасовим рішенням на шляху до автоматизації, що реалізується засобами штучного інтелекту.

Розробка програмного робота для автоматизації процесів вимагає проходження наступних етапів: підготовка RPA (налаштування оточення розробки, оцінка застосування можливостей RPA), проектування рішення автоматизації процесів (формалізація процесу автоматизації, проектування архітектурного рішення, проектування сценаріїв тестів), побудова RPA рішення (побудова архітектурного рішення, тестування на етапах реалізації, підготовка тестування RPA рішення та тестових даних), тестування, доробка, технічна підтримка [2].

Згідно з вищепереліченими етапами в середовищі розробників особлива увага приділяється генерації архітектурного рішення (один з етапів побудови RPA рішення) [3]. Особливо важливою ця задача представляється для процесів з великою кількістю транзакцій

Метою даної статті є аналіз і порівняльні дослідження існуючих архітектурних рішень (патернів) для процесів з великою кількістю транзакцій, і виявлення переваг і недоліків кожної з них.

Основна частина

Проведені дослідження показали, що машина станів або кінцевий автомат (FSM) або автомат з кінцевим станом (FSA, множина: автомати) є математичною моделлю обчислення (в даній статті розглядається шаблон кінцевого автомата, розроблений американською компанією UiPath). Це абстрактна машина (рис. 1), яка може перебувати точно одному з кінцевого числа станів в будь-який момент часу.

FSM може переходити від одного стану до іншого у відповідь на деякі зовнішні входи, зміна від одного стан до іншого називається переходом. FSM визначається списком його станів, його початковий стан і умови для кожного переходу.

Основні правила при використанні кінцевого автомата:

- оскільки система може перебувати тільки в одному стані за раз, хоча б один перехід за умовою від поточного стану до іншого повинен стати справжнім або шляхом створення умови в коді, що працюють всередині стану, або зовнішня умова або їх комбінація;

- умови переходу з кожного стану повинні бути винятковими (два переходи не можуть бути істинними одночасно, що дозволяє мати кілька виходів і поточного).

- інше погоджене правило полягає в тому, що області дії переходу. Вся обробка повинна виконуватися всередині стану.

Проблеми, які можна вирішити за допомогою цього шаблону:

а) зберігання та читання цих змін проекту;

б) поділ запуску, виконання та завершення використання IT-ресурсів:

1) для всіх повторних транзакцій перезапуск використання IT-ресурсу;

в) запровадження надійної обробки виключень і повторів спроб транзакцій:

1) захоплення винятків за його типом;

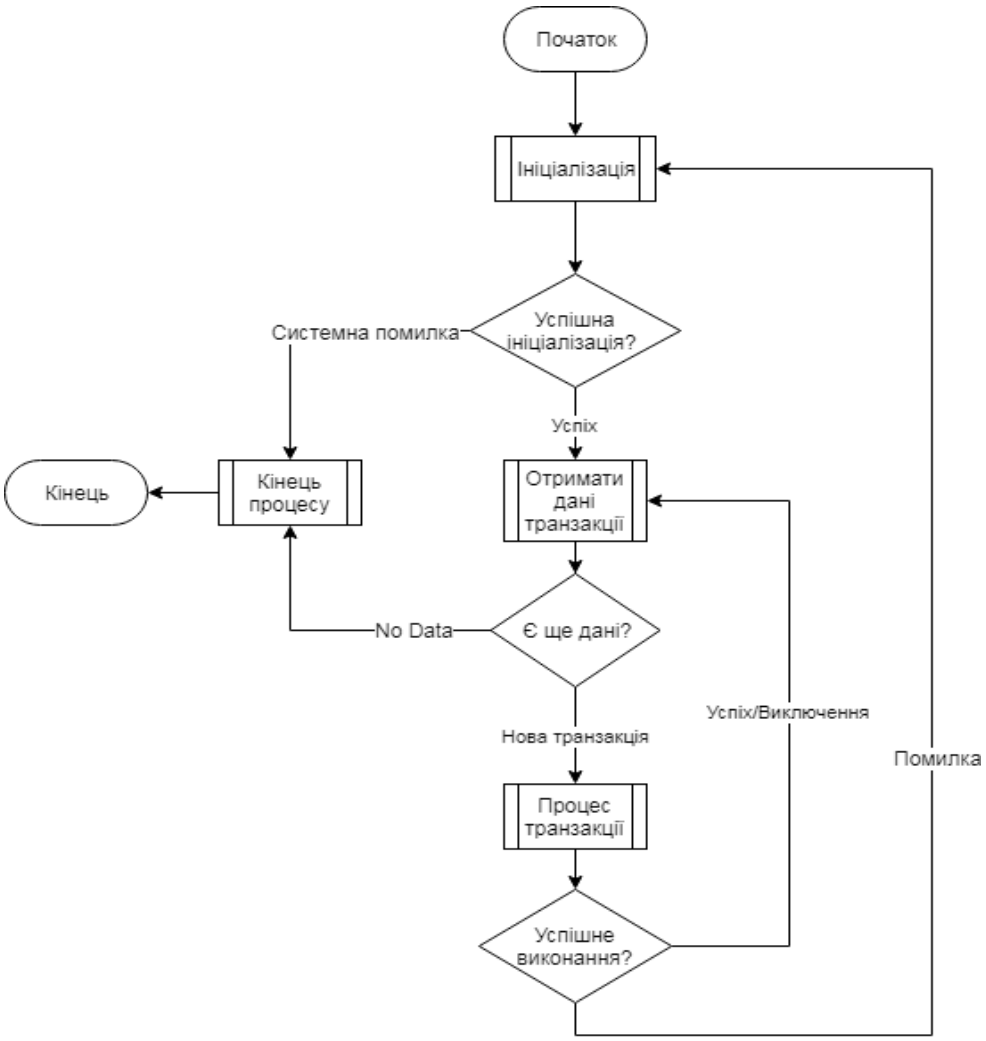

Рис. 1. Машина станів

2) використання типу виключення для повторення транзакцій, які не виконувалися із завершенням роботи робота;

г) захоплення і передача в журнал всіх видів винятків і відповідної інформації про транзакції [4].

Очевидною перевагою є поновлення виконання процесу з будь-якого його етапу виконання, документування ходу виконання процесу і облік всіх можливих реакцій на появу непередбачених збоїв. Недоліком є те, що, якщо процес містить в собі підпроцеси які виконують не зв'язні один з одним транзакції (не пов'язані за своїм основним призначенням, але це не відноситься до використання загальних IT-ресурсів).

На перший погляд очевидне рішення - це проектування двох (або більше, в залежності від кількості підпроцесів) роботів з однаковими архітектурними рішеннями, таким чином позбавляючи процес від одночасного використання IT-ресурсів, підвищуючи стійкість до збоїв в роботі [5, 6]. Однак виникає ряд зауважень: можлива ситуація необхідного використання загальних IT-ресурсів підпроцесами основного процесу, через що елементи архітектурного рішення дублюються і як наслідок збільшується час виконання завдань роботом. Але з іншого боку їх об'єднання в одне архітектурне рішення збільшує навантаження у використанні IT-ресурсів.

Безліч процесів мають проблему, пов'язану з явищем, яке називають множинне дублювання послідовного виконання спільних елементів під процесів [7–10]. З теоретичної точки зору концепція щодо цього проста і відповідає більш ніж одному логічному знаку в певному місці моделі роботи процесу, наприклад, в мережі Петрі.

З практичної точки зору це означає, що одна дія на графіку робочого процесу може запускати одночасно кілька активних екземплярів. Основна проблема з реалізацією цього шаблону архітектурного рішення Multiple Instances Requiring Synchronization (екземпляри вимагають синхронізації), наданий Технічним університетом Ейндховена (під керівництвом професора Віла ван дер Аальста) і Технологічним університетом Квінсленда (під керівництвом професора Артура тер Гофстеде), полягає в тому, що через конструктивні обмеження більшість механізмів робочого процесу не дозволяють одночасно активувати більше одного екземпляра однієї і тієї ж дії [11, 12].

У шаблоні Multiple Instances Requiring Synchronization одна транзакція активується кілька разів одночасно. Кількість екземплярів транзакцій можуть бути невідомо під час розробки. Після завершення всіх екземплярів цієї активності необхідно запустити іншу дію (рис. 2) [13].

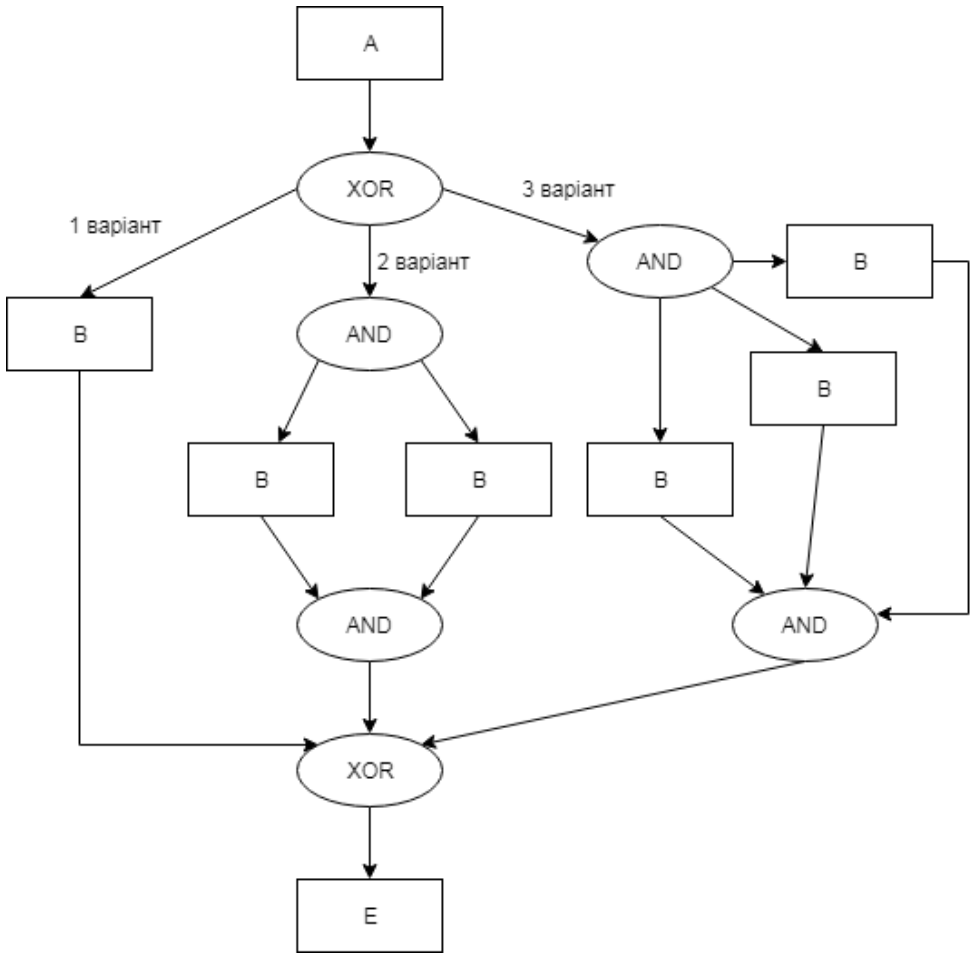

Рис. 2. Шаблон Multiple Instances Requiring Synchronization
з трьома варіантами одночасного виконання транзакцій

Дане архітектурне рішення пропонує спосіб зменшення навантаження – розпаралелювання виконання транзакцій, але не враховує відмовостійкість як це є в машині станів.

З точки зору реалізації UiPath в інтеграції з його сервером з управління програмними роботами Orchestrator дозволяють розподіляти транзакції згідно доступному кількості локальних машин [14]. На початковому етапі виконання процесу необхідно визначити кількість доступних локальних машин, на кожній з яких «знаходиться» робот, який керує процесом виконання архітектурного рішення. Після чого на кожну машину формується черга з транзакцій, які повинні їх виконати локальна машина.

Висновки

Аналіз нової області в сфері інформаційних технологій - RPA показав одну з актуальних завдань генерації рішення автоматизації бізнес процесів. Це визначення архітектурного рішення, яке визначає функціонування робота. Аналіз вище наведених прикладів показав переваги і недоліки кожної з схем рішень в ситуації автоматизації процесу з великою кількістю транакцій.

Таким чином, якщо розглядати шаблон Multiple Instances Requiring Synchronization, то архітектурне рішення підпроцесу (підпроцес B на схемі рис. 2) можна описати подібно вище наведеної машині станів, таким чином взявши властивість стійкості до збоїв, але при цьому робота кожної транзакції буде незалежна один від одного, тобто паралельне виконання.

Таким чином з'являється можливість зменшити час виконання процесу і зникає дублювання виконання елементів архітектурного рішення всього процесу. Але варто враховувати, що кожен екземпляр B (описаний як машина станів) повинен бути незалежний один від одного (тобто не перебувати в послідовності).

Недоліком природно є збільшення ресурсів на виконання, тобто збільшення кількості локальних машин, на кожній з яких виконується одна транзакція. Тому такий підхід доречний для великих процесів з великою кількістю транзакцій.

СПИСОК ЛІТЕРАТУРИ

1. UiPath Automation Handbook // UiPath. - 4 - 5 с.
2. UiPath. Business Analyst Training. Business Analyst's role - RPA Journey // UiPath. - 5с.
3. UiPath. Solution Architect. Preparation - Project Governance // UiPath. - 26с.
4. Mihai Dunareanu / UiPath REFramework Manua / Mihai Dunareanu // UiPath. - 7с.

5. Amin Salih Mohammed (2018), “modification of load balancing method in networks with wimax technology”, Qalaai Zanist Journal, Vol. 3, No. 2, pp. 791-802.
6. Kuchuk G., Kovalenko A., Kharchenko V., Shamraev A., "Resource-oriented approaches to implementation of traffic control technologies in safety-critical I&C systems" in book: Green IT Engineering: Components Network and Systems Implementation, Springer International Publishing, vol. 105, pp. 313-338, 2017.
7. Kuchuk G.A. An Approach To Development Of Complex Metric For Multiservice Network Security Assessment / G.A. Kuchuk, A.A. Kovalenko, A.A. Mozhaev // Statistical Methods Of Signal and Data Processing (SMSDP – 2010): Proc. Int. Conf., October 13-14, 2010.– Kiev: NAU, RED, IEEE Ukraine section joint SP, 2010. – P. 158 – 160.
8. Kuchuk G. Approaches to selection of combinatorial algorithm for optimization in network traffic control of safety-critical systems / G. Kuchuk, V. Kharchenko, A. Kovalenko, E.Ruchkov // East-West Design & Test Symposium (EWDTS). – 2016. –P. 1-6. doi : https://doi.org/10.1109/EWDTS.2016.7807655.
9. Amin Salih Mohammed, D Yuvaraj, M. Sivaram Murugan, V. Porkodi, “Detection and removal of black hole attack in mobile ad hoc networks using grp protocol”, International Journal of Advanced Computer Research, vol. 9, no. 6, pp. 1-6, 2018, DOI: http://doi.org/10.26483/ijarcs.v9i6.6335
10. Saravana Balaji B., A. Salih Mohammed, Chiai Al-Atroshi, “Adaptability of SOA in IoT Services – An Empirical Survey”, International Journal of Computer Applications, vol. 182(31), pp. 25-28, 2018, DOI: http://doi.org/10.5120/ijca2018918249
11. WMP van der Aalst. Chapter 10: Three Good reasons for Using a Petri-net-based Workflow Management System. In T. Wakayama et al., Editor, Information and Process Integration in Enterprises: Rethinking documents, The Kluwer International Series in Engineering and Computer Science, pages 161 {182. Kluwer Academic Publishers, Norwell, 1998.
12. WMP van der Aalst, AP Barros, AHM ter Hofstede, and B. Kiepuszewski. Workflow Patterns. Unpublished (46 pages), 2000..
13. WMP van der Aalst, AP Barros, AHM ter Hofstede, B. Kiepuszewski / Advanced Workflow Patterns / WMP van der Aalst, AP Barros, AHM ter Hofstede, B. Kiepuszewski // Eindhoven: Eindhoven University of Technology. - 7 - 8 с.
14. .Emplotics / Функціональні можливості UiPath / Опис UiPath - https://www.emplotics.com/rpa/описание-uipath.html.

Рецензент: д-р техн. наук, проф. К. С. Козелкова,
Державний університет телекомунікацій, Київ
Received (Надійшла) 24.10.2018
Accepted for publication (Прийнята до друку) 09.01.2019

Анализ и сравнительные исследования архитектурных решений программных роботов для процессов с большим количеством транзакций

Д. А. Ильин, С. Г. Семенов

Анализ новой области в сфере информационных технологий - RPA показал одну из актуальных задач генерации решение автоматизации бизнес процессов. Это определение архитектурного решения, которое определяет функционирование работа. Анализ приведенных выше примеров показал преимущества и недостатки каждой из схем решений в ситуации автоматизации процесса с большим количеством транзакций. Раскрыта сущность и актуальность темы RPA, а также существующих архитектурных решений программных роботов и представлены преимущества и недостатки в существующих решениях в ситуации процессов с большим количеством транзакций. В заключении предложено возможное решение проблемы с выделением преимуществ и недостатков. Если рассматривать шаблон Multiple Instances Requiring Synchronization, то архитектурное решение подпроцесса можно описать подобно выше приведенной машине состояний, таким образом взяв свойство устойчивости к сбоям, но при этом работа каждой транзакции будет независимая друг от друга, то есть параллельное выполнение. Таким образом, появляется возможность уменьшить время выполнения процесса и исчезает дублирование выполнения элементов архитектурного решения всего процесса. Но стоит учитывать, что каждый экземпляр B (описан как машина состояний) должен быть независим друг от друга (т.е. не находиться в последовательности). Недостатком естественно является увеличение ресурсов на выполнение, то есть увеличение количества локальных машин, на каждой из которых выполняется одна транзакция. Поэтому такой подход уместен для больших процессов с большим количеством транзакций.

Ключевые слова: RPA, программные работы, архитектурные решения, паттерны архитектурных решений..

Analysis and comparative studies of architectural solutions of program robots for processes with a large number of transactions

D. Ilin, S. Semenov

Analysis of the new area in the field of information technology - RPA has shown one of the most urgent tasks of generating business automation solutions. This is the definition of the architectural solution that determines the functioning of the robot. The analysis of the above examples showed the advantages and disadvantages of each of the decision schemes in a situation of automation of the process with a large number of transactions. The essence and relevance of the RPA topic, as well as existing architectural solutions of software robots, are revealed, and the advantages and disadvantages of existing solutions in situations involving processes with a large number of transactions are presented. The conclusion suggests a possible solution to the problem with the allocation of advantages and disadvantages. If you consider the Multiple Instances Requiring Synchronization template, then the architectural subprocess solution can be described just like the above machine states, thus taking the property of stability to failures, but the work of each transaction will be independent of each other, that is, parallel execution. Thus, it is possible to reduce the time of execution of the process and disappear duplication of the implementation of elements of the architectural decision of the entire process. But it should be borne in mind that each instance B (described as a state machine) must be independent of each other (that is, it does not have to be in sequence). A disadvantage naturally is an increase in resources for execution, that is, an increase in the number of local machines, each of which executes one transaction. Therefore, this approach is appropriate for large processes with a large number of transactions.

Keywords: RPA, software robots, architectural solutions, patterns of architectural solutions.