УДК 004.41
О. Гузеляк, Ю. Шевчук, Б. Береженко, І. Боднарчук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ПРОГРАМНА АРХІТЕКТУРА В РОЗПОДІЛЕНИХ КОМАНДАХ ГНУЧКИХ ПРОЄКТІВ

UDC 004.41
O. Huzeliak, Yu. Shevchuk, B. Berezhenko, I. Bodnarchuk

SOFTWARE ARCHITECTURE DEGIGN IN DISTRIBUTED TEAMS OF AGILE PROJECTS

Базуючись на принципах Agile Маніфесту [1] практики Agile розглядають первинну розробку програмної архітектури (ПА), як витрати ресурсів, які можуть не окупитись. В Agile традиційно приділяли основну увагу при плануванні ітерацій потребам замовника і сценаріям використання (так званим user stories), залишаючи поза увагою архітектурне проектування.

Однак, при застосуванні Agile до великих проектів ці погляди змінились. Так в роботі [2] показано, що ігнорування архітектурного проектування може спричинити неоптимальні рішення, а в роботі [3] автор стверджував, шо невідповідність розробки великих проектів архітектурному проекту може привести до їх провалу. В роботах [3], [4] підкреслена ключова роль архітектурного проекту при застосуванні Agile до розробки великих і складних проектів, а також фактичну неможливість їх виконання без використання архітектури. Це відбувається через втрату зв'язку беклогів (специфікацій вимог) з усією системою, що робить неможливим передбачити реальні наслідки своїх рішень при плануванні кожної ітерації.

Для вирішення цих проблем були впроваджені в ітерації гнучких методів елементи архітектурного проектування та почали реалізовувати зв’язок архітектурного проекту всієї програмної системи з проектами локальних команд [6]. При реалізації великих проектів в рамках гнучких методів створювалась «координуюча» команда, яка вела архітектурний проект всієї програмної системи і здійснювала координацію внесення змін в архітектури компонентів, які розробляли локальні команди [6]. Для цього були введені нові ролі в координуючій команді. Це архітектор системи, архітектор бізнес-процесів та інші. Процеси архітектурного проектування в локальній команді може вести або архітектор, або один з членів команди, суміщаючи ці обов’язки з роллю розробника чи іншою роллю на проєкті [7]. Для забезпечення ефективності реалізації взаємодії процесів на локальному і глобальному рівнях необхідно було розробити відповідні моделі цих процесів та створити CASE-засіб, який би автоматизував процеси створення та обміну архітектурною інформацією. В [8] було запропоновано двоетапну схему такої взаємодії (див. рис. 1). На першому етапі створюються альтернативні варіанти архітектури з врахуванням вимог за технологією, яка залежить від прийнятого стилю документування. Для оцінювання альтернатив обчислюються їх відносні оцінки по кожному з критеріїв якості з використанням модифікованого методу аналізу ієрархій (MAI). Вибір кращого варіанта архітектури виконується методом аналізу компромісів або на основі значення інтегрального показника якості [9].

На другому етапі вибраний варіант архітектури впроваджується в гнучкому методі розробки. В разі виявлення нових вимог до програмної системи (ПС) або зміни існуючих вносяться відповідні зміни в архітектуру. Зміни виконуються шляхом корегування коду відповідного патерну (шаблону) проєктування або його заміною на альтернативний. Для дослідження та оптимізації впливу цих змін на якість проводиться корекція значень критеріїв якості. Процедура корекції розроблена на основі застосування методу «заміщення – компенсації», в якому збільшення деяких критеріїв якості відбувається за рахунок зменшення інших, що дозволяє оптимізувати ці зміни і не виходити за межі бюджету проекту [10].

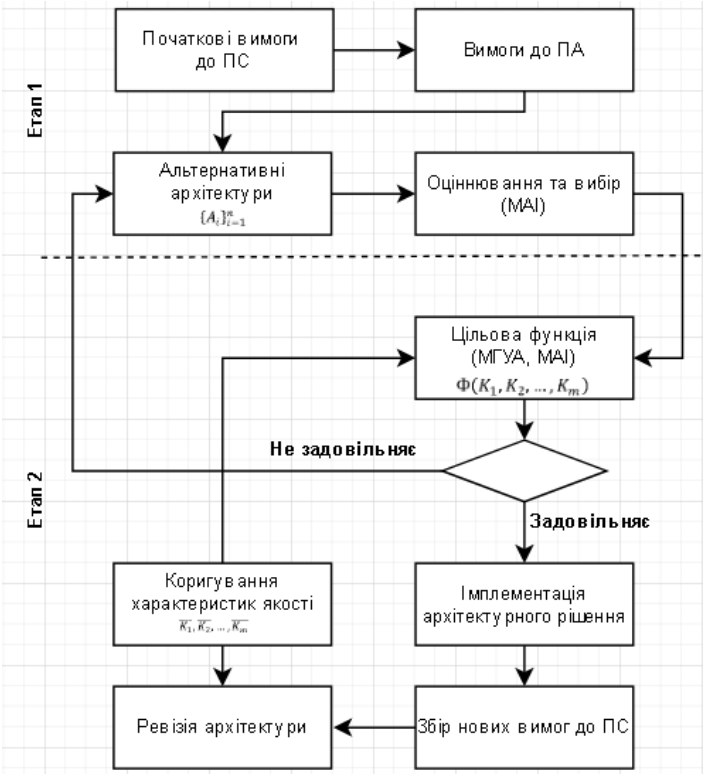

Рисунок 1. Схема проєктування архітектури в розподілених командах для гнучких проєктів

Перший і другий етапи об’єднуються процедурою обчислення цільової функції якості архітектури. Цільова функція визначається методом групового урахування аргументів в поєднанні з MAI [8]. Запропонована процедура дозволить, крім координації процесів, також забезпечити якість проєкту. Для адаптації процесів архітектурного проектування до вимог Agile необхідно максимально зменшити час на процеси архітектурного проектування. Для цього пропонується створити систему, яка б могла підтримувати спільне використання командами архітектурної інформації. Ці процедури повинні виконуватись командами в ітераціях Agile, а також координуючою командою для архітектури всієї системи. Архітектурний проект приймається і створюється на фазі аналізу вимог перед спринтами розробки. Проте в ході розробки може виникнути потреба змінювати архітектуру, і командою SCRUM або архітектором приймаються певні нові рішення. Основною задачею тут є оперативне оцінювання можливих архітектурних рішень та вибору найбільш прийнятної для задоволення вимог всіх зацікавлених сторін. Тут може бути використаний MAI, а для коригування архітектури в ітераціях той же MAI або метод групового урахування аргументів (МГУА), які набагато ефективніші ніж метод сценаріїв ATAM, що розглядався в [14]. Також необхідно автоматизувати процедури формування альтернативних архітектур на основі архітектурно значимих вимог та при внесені змін в програмний код і здійсненні реінжинірингу. Ці процедури, а також метод оцінювання MAI та процедури зберігання архітектурної інформації реалізовані в засобі «Архітектор» [15], [16]. Тому систему можна будувати на його основі.

Отже, хоча Agile підхід має багато переваг він все ж таки не є універсальним рішенням, і компанії часто використовують поєднання гнучких та планово орієнтовних методів, що називається гібридним підходом. Він є результатом компромісу використання переваг ітеративного, логічного та спільного підходів із підтримкою певного рівня планування та структури. При цьому гібридний підхід має ряд переваг, таких як: належна підтримка бізнес-проектів, допомагає досягти хорошої якості продукції, адекватно відповідає зовнішнім вимогам (наприклад, стандартам) та допомагає пришвидшити розробку й скоротити час виходу продукту на ринок, а також дає можливість постійно вдосконалювати якість завдяки гібридним конструкціям, покращити задоволеність команди. Гібридний підхід має також

недоліки: проблеми забезпечення продуктивності проекту та затримки релізів. Мінімізацію недоліків реалізують застосуванням ширшого впровадження архітектурного проектування в Agile, а також застосуванням засобів автоматизації цих процесів.

Література

1. Agile alliance: manifesto for agile software development. URL: http:// agilemanifesto.org [retrieved: 10.10.2022]
2. Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. Information and software technology, 50 (9–10), 833–859.
3. Bowers, J., May, J., Melander, E., Baarman, M., & Ayoob, A. (2002, August). Tailoring XP for large system mission critical software development. In Conference on Extreme Programming and Agile Methods (pp. 100–111). Springer, Berlin, Heidelberg.
4. Ghanam Y. Andreichuk D., Maurer F. Reactive variability management in agile Software development. In: Agile conference. Orlando, FL: Computer Society; 2010, p. 27–34.
5. Abrahamsson, P., Babar, M. A., & Kruchten, P. (2010). Agility and architecture: Can they coexist? IEEE Software, 27 (2), 16–22.
6. Dingsøyr, T., Moe, N. B., & Seim, E. A. (2018). Coordinating knowledge work in multiteam programs: findings from a large-scale agile development program. Project Management Journal, 49 (6), 64–77.
7. Eloranta, V. P., & Koskimies, K. (2013). Software architecture practices in Agile enterprises. In Aligning Enterprise, System, and Software Architectures (p. 230–249). IGI Global.
8. Bodnarchuk, I., Lisovyi, V., Kharchenko, O., & Galai, I. (2018, September). Adaptive Method for Assessment and Selection of Software Architecture in Flexible Techniques of Design. In 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 1, p. 292–297). IEEE.
9. Kharchenko, A., Bodnarchuk, I., Halay, I., & Yatcyshyn, V. (2016). An Optimal Trade-off Solution of the Software Architecture Choice Problem. Journal of Information and Computing Science. 11 (4). P. 281–290.
10. I. Bodnarchuk, et al. «Multicriteria choice of software architecture using dynamic correction of quality attributes.» International Conference on Computer Science, Engineering and Education Applications. Springer, Cham, 2019, p. 419–427.
11. Galster M., & Avgeriou P. (2014). Supporting variability through agility to achieve adaptable architectures. In Agile Software Architecture (p. 139–159). Morgan Kaufmann.
12. KharchenkoA., Halay I., & Bodnarchuk I. (2016, September). Multicriteria architecture choice of software system under design and reengineering. In 2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT) (p. 4–8). IEEE.
13. Харченко О. Г., Боднарчук,І. О., Райчев І. Е., & Галай І. О. (2015). Інструментальний засіб порівняльного оцінювання і багатокритеріального вибору архітектури програмних систем. Інженерія програмного забезпечення. Київ: НАУ, 2015. № 1. С. 10–24.