НЕДОЛІКИ ДІАГРАМИ ПОСЛІДОВНОСТІ ТА ЇЇ АЛЬТЕРНАТИВА

Старіков Назарій Валерійович
здобувач вищої освіти факультету комп'ютерних та інформаційних технологій
Луцький національний технічний університет, Україна

Науковий керівник: Самчук Людмила Михайлівна
канд. техн. наук, доцент кафедри прикладної механіки та мехатроніки
Луцький національний технічний університет, Україна

Існує велика кількість видів UMLдіаграм, всі з яких різняться за рівнем інформативності, корисності та інтуїтивності розуміння. Різноманітні особисті спостереження показують, що діаграма послідовності є однією з найкорисніших, а тобто однією з найбільш пріоритетною, адже завдяки неї простіше створювати інші діаграми UML. Діаграма послідовності описує поведінкові особливості програмного забезпечення з урахуванням часових характеристик та взаємодії сутностей, як наслідок класів, між собою. Саме тому створивши діаграму послідовності відкриваються широкі можливості для швидкого створення інших графічних моделей. Дана робота пропонує альтернативну та узагальнену модель діаграми послідовностей, яка надає краще розуміння процесів відображених на діаграмі та носить назву TM(Thinging Machine).

UML діаграми взаємопов'язані між собою та повинні відповідати одна одній, адже різні елементи програмного продукту мають взаємодіяти між собою безпомилково. Труднощі моделювання полягають у великому числі взаємодіючих елементів різних діаграм. У більше ніж 80% студентів виникають труднощі з розробкою програмного продукту або навіть з частиною такої розробки. Значна частина з таких студентів проявляє труднощі при ідентифікації взаємодіючих об'єктів та передачі даних з відповідними аргументами. Саме тому виникла потреба у створенні необхідного способу графічного відображення, яке стало би заміною звичним діаграмам послідовності.

Згідно роботи [1], незважаючи на явну інтуїтивність притаманну діаграмам послідовності, не завжди очевидно як зробити правильну діаграму послідовності проаналізувавши основні вимоги до програмного продукту. Був запропонований новий підхід відомий під назвою STAIRS. Він розглядає процес розробки як процес навчання через опис. Під час процесу навчання цей опис змінюється від нечіткого ескізу до деталізованого.

У дослідженні [2]також описується SD (Sequence Diagram) як той вид UML діаграм, який легкий у використанні та розумінні, що зумовлює їхнє розповсюдження у багатьох галузях. Але на думку автора даного дослідження, даний тип діаграм надає лише частковий вид на систему та принципи її роботи. На його суб'єктивну думку, семантика діаграм UML 2 не позбавлена недоліків, так як є недостатньо чіткою та ясною у розумінні. Тим самим автором піднімалися інші теми також, проте найбільше уваги привертала семантика, адже це основа правил створення нової діаграми як окремого графічного зображення. Використовуючи мову програмування Python та певні їх бібліотеки автор роботи запропонував створювати точне відображення базоване на hMS Ссемантиці.

Користувач вставляє власну картку(1), яка направляється до банкомату (2), аби її дані були оброблені(3). Банкомат отримує номер карти(4), який направляє в банківську систему(5). Те, що надсилається, можна неправильно зрозуміти як фізичну картку, яка називається повідомленням. Правильне розуміння полягає у

тому, що банкомат відправляє вбудований номер картки для звірення. Ця неоднозначність полягає в усуненні того, що відправляється, завдяки явно визначеному процесу отримання номеру з фізичної карти. У банкоматі стрілка направлена від користувача до банкомату – це фізична картка, а стрілка направлена від банкомату до банку – це номер. І ці дві стрілки не перетинаються. Номер карти піддається обробці(6) у банківській системі як дійсний(7) або недійсний(8). Якщо номер карти є дійсним, тоді відповідне повідомлення про успіх операції(9) відправляється до банкомату. Банкомат обробляє повідомлення(10) та потребує PIN код(11), що потребується від користувача(12). Якщо номер карти недійсний, тоді з'являється інше відповідне повідомлення, яке переходить до банкомату, щоб спричинити відмову картки(14). Користувач прочитує повідомлення сприяючи створенню записуPINкоду(16). Повідомлення переходить до банкомату(17), а потім відправляється до банківської системи(18), де PINкод проходить перевірку(19). Якщо PINкод недійсний(20), відповідне повідомлення про недійсність PINкоду з'являється на екрані банкомату(13), спричиняючи відхилення карти(14). Якщо PINдійсний(21), тоді в банкомат надходить відповідне повідомлення(22). Банкомат питає у користувача про кількість, яка надходить до користувача(23).

Користувач вводить необхідну суму(24), число якої отримується банкоматом, а відповідні дані надходять до банківської системи(25). Ці дані направляються до підсистеми облікового запису. Підсистема облікового запису отримує PINкод (28) і шукає відповідний обліковий запис або суму у власній БД. Таким чином, відповідний баланс отримується (31) та порівнюється(32) з необхідною сумою. Якщо сума на рахунку недостатня, тоді необхідне повідомлення відправляється в банкомат(33), який як наслідок відправляє його користувачу(34 і 35). Якщо сума достатня, відображається необхідне повідомлення(36), яке переходить до банкомату, а затим до користувача(37)(Рисунок 1).

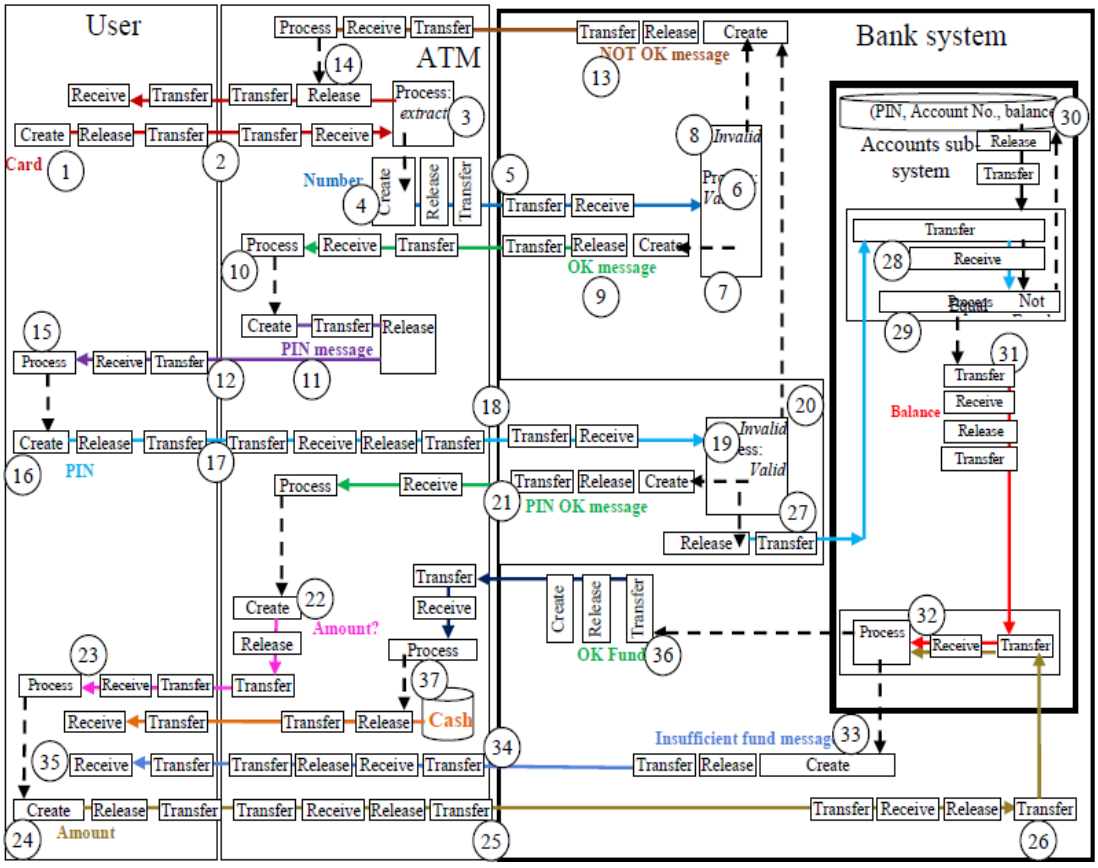

Рис. 1. **Діаграма послідовності, робота банкомату**

В роботі, пропонується альтернативна узагальнена модель до діаграми послідовності. Це мотивується труднощами згаданими у літературі заважаючи на розвиток відповідного семантичного уточнення необхідної поведінки SD. До того ж, дослідження показали, що студенти мають труднощі пов'язані з ідентифікацією дійсних взаємодіючих об'єктів та конструювання повідомлень з відповідними аргументами. ТМ модель більш систематична, з більш ясним розділенням статичних та динамічних аспектів. Наше твердження обґрунтовано вищевказаними дослідними випадками. Відповідно, здається, що прийняття цього нового підходу вимагає вивчення того, як інтегрувати ТМ модель у апарат діаграм UML.

Список використаних джерел:

1. Ø.Haugen, K. E. Husa, R. K. Runde, and K. Stølen, —Why timed sequence diagrams require three-event semantics, in Scenarios: Models, Transformations and Tools. Lecture Notes in Computer Science, vol. 3466, S. Leue and T. J. Systä, Eds. Berlin: Springer, 2005, pp. 1–25. DOI: 10.1007/11495628_1pp 1-25.
2. S.Busard, C. Ponsard, and C. Pecheur, —Verification of scenario-based behavioural models using Capella and PyNuSMV,|| Proc. 9th Int. Conf. Model-Driven Eng. Software Dev., pp. 337–343, 2021. DOI: 10.5220/0010346103370343