11.2 Засоби архітектурного моделювання складних систем

На найвищому рівні опису модель структури системи разом з моделлю основних принципів її функціонування називають *архітектурою*. Поширені системи архітектурного моделювання відрізняються, в основному, особливостями спрямування на моделі структури або функцій (процесів), моделі програмних, апаратних засобів чи персоналу, моделі для окремих предметних галузей.

Методи моделювання програмованих виробничих систем останнім часом набувають усе більшої актуальності. Для цього використовуються як засоби моделювання, що вбудовані у САПР, наприклад, в системах Factory-Link (US DATA Co., США), InTouch (Wonderware, США), Genesis (Iconics, США), RealFlex (BJ Software Systems, США), Sitex (Jade Software, Великобританія), FIX (Intellution, США), Trace-Mode (AdAstra, Росія), IGSS (Seven Technologie, Данія), Image (Технолинк, Росія), так і спеціалізовані засоби структурно-логічного моделювання програмованих систем, зокрема UML (уніфікована мова моделювання).

11.2.1 Уніфікована мова системного моделювання UML

Структурний аналіз і метод Джексона розробки систем (JSD) як універсальні методи моделювання і проектування складних систем вперше були запропоновані ще у 70-х роках двадцятого століття. У 80-90-х роках XX ст. до них додалися об'єктно-орієнтовані методи, запропоновані Г.Бучем і Д.Рамбо. Ці методи інтегровані в єдиний уніфікований метод на основі мови *UML* (*Unified Modeling Language*). Мова UML є досить строгим і потужним засобом моделювання, який може бути ефективно використаний для побудови концептуальних, логічних і графічних моделей складних систем різного цільового призначення.

UML створена для візуального моделювання систем на основі об'єктно-орієнтованого підходу. Ця мова створювалася для оптимізації процесу розроблення програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту. Вона дозволяє будувати моделі для усіх фаз життєвого циклу програмного забезпечення.

Наразі зареєстровано як міжнародний стандарт ISO/IEC 19501:2005 «Information technology – Open Distributed Processing – Unified Modeling Language (UML)».

Словник мови UML містить три види блоків:

- сутності;
- відношення;
- діаграми.

Сутності в UML – це абстракції, які є основними елементами моделі. Відношення пов'язують сутності; діаграми наочно зображують відношення і взаємодію сутностей.

В UML є чотири типи сутностей:

- структурні;
- поведінкові;
- групувальні;
- анотаційні.

Структурні сутності – показують складові частини моделі, які відповідають концептуальним або фізичним елементам системи. Існують 7 видів структурних сутностей: *Клас, Інтерфейс, Кооперація, Прецедент, Активний клас, Компонент, Вузол.*

Поведінкові сутності описують поведінку системи у часі і просторі. Існує 2 основних типи поведінкових сутностей: *Взаємодія* і *Автомат.*

Групувальні сутності є частинами моделі UML з функціями організації. Це блоки, на які можна розкласти модель. Є тільки 1 первинна групувальна сутність – *пакет.*

Анотаційні сутності – пояснювальні частини моделі UML. Це коментарі для додаткового опису, роз'яснення або примітки до елемента моделі. Є 1 базовий тип анотаційних елементів – *примітка.*

У мові UML визначені 4 типи відношень:

- залежність;
- асоціація;
- узагальнення;
- реалізація.

Діаграма (diagram) – графічне зображення сукупності елементів моделі у формі зв'язного графу, вершинам і ребрам (дугам) якого приписується певний зміст.

У мові UML визначені такі канонічні діаграми:

- варіантів використання (use case diagram, діаграма сценаріїв, діаграма прецедентів);
- класів (class diagram);
- кооперації (collaboration diagram, діаграми співробітництва);
- послідовності (sequence diagram);
- станів (statechart diagram);
- діяльності (activity diagram);
- компонентів (component diagram);
- розгортання (deployment diagram).

Кожна з цих діаграм деталізує і конкретизує різні уявлення про модель складної системи в термінах мови UML. При цьому ***діаграма варіантів використання є найбільш загальною концептуальною моделлю складної системи, яка є початковою для побудови решти діаграм***. Приклад діаграми варіантів використання наведений на рис. 11.1.

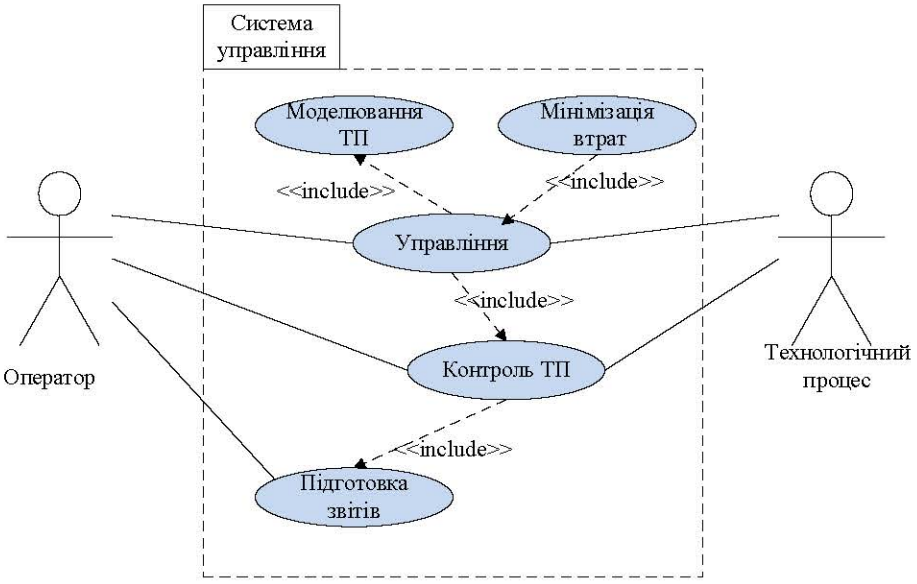

Рисунок 11.1 – Діаграма варіантів використання системи управління

Є різні точки зору на побудову діаграм класів залежно від цілей їх застосування:

– концептуальна точка зору – діаграма класів описує модель предметної області, в ній присутні лише класи прикладних об'єктів;

– точка зору специфікації – діаграма класів застосовується при проектуванні інформаційних систем;

– точка зору реалізації – діаграма класів містить класи, що використовуються безпосередньо в програмному коді (при використанні об'єктно-орієнтованих мов програмування).

Взаємозв'язок – це особливий тип логічних відносин між сутностями, показаних на діаграмах класів і об'єктів. У UML наявні нижченаведені види відносин:*Асоціація* показує, що об'єкти однієї сутності (класу) пов'язані з об'єктами іншої сутності.

Агрегація – проста асоціація між двома класами відображає структурні відношення між рівноправними сутностями, коли обидва класи знаходяться на одному концептуальному рівні і жоден не важливіший за іншого.

Композиція – більш строгий варіант агрегації. Якщо контейнер буде знищений, то весь його вміст також буде знищено. Графічно подається як і агрегація, але з зафарбованим ромбиком.

Узагальнення (спадкування) показує, що один з двох зв'язаних класів (підтип) є окремим випадком (формою) іншого (надтипу), який називається узагальненням першого. На практиці це означає, що будь-який екземпляр підтипу є також екземпляром надтипу.

Потужність відносини (мультиплікатор) означає число зв'язків між кожним екземпляром класу (об'єктом) на початку лінії з екземпляром класу в кінці.

Діаграми класів (class diagram) при моделюванні об'єктно-орієнтованих систем зустрічаються частіше інших. На таких діаграмах відображається безліч класів, інтерфейсів, кооперацій і відношень між ними. Діаграма класів служить для подання статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. Крім того, діаграми класів складають основу ще двох діаграм – компонентів і розгортання.

Діаграма класів (рис. 11.2) може відбивати різні взаємозв'язки між такими окремими сутностями предметної області, як об'єкти і підсистеми, а також описує їхню внутрішню структуру і типи відношень. На даній діаграмі не вказується інформація про тимчасові аспекти функціонування системи.

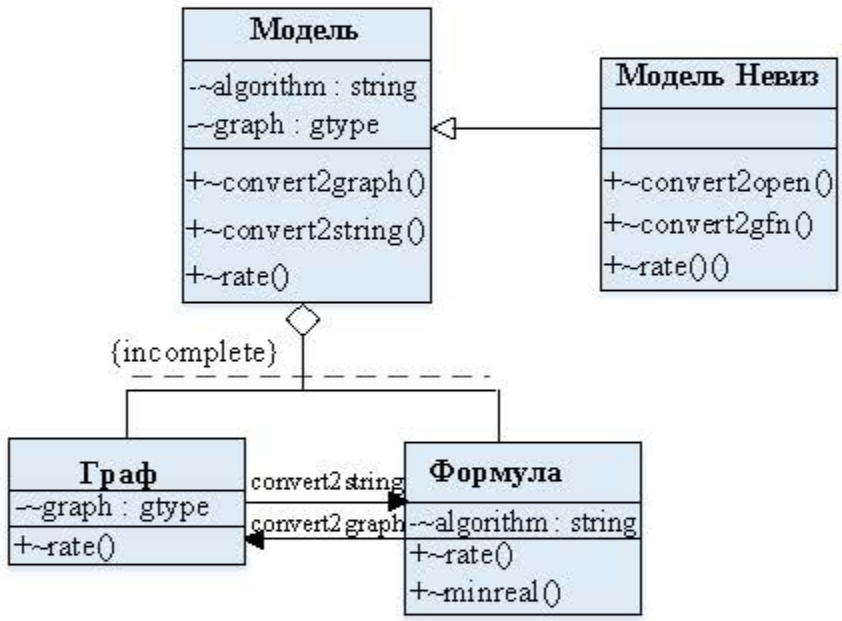

Рисунок 11.2 – Діаграма класів

Діаграма послідовності (sequence diagram) – діаграма, на якій показано взаємодію об'єктів (обмін між ними сигналами і повідомленнями), впорядковане за часом, з відображенням тривалості обробки і порядку їх появи.

Основними елементами діаграми послідовності (рис. 11.3) є позначення об'єктів (прямокутники з назвами об'єктів), вертикальні «лінії життя» (lifeline), що відображають плин часу, прямокутники, що відображають діяльність об'єкта або виконання ним певної функції (прямокутники на пунктирній «лінії життя»), і стрілки, що показують обмін сигналами або повідомленнями між об'єктами.

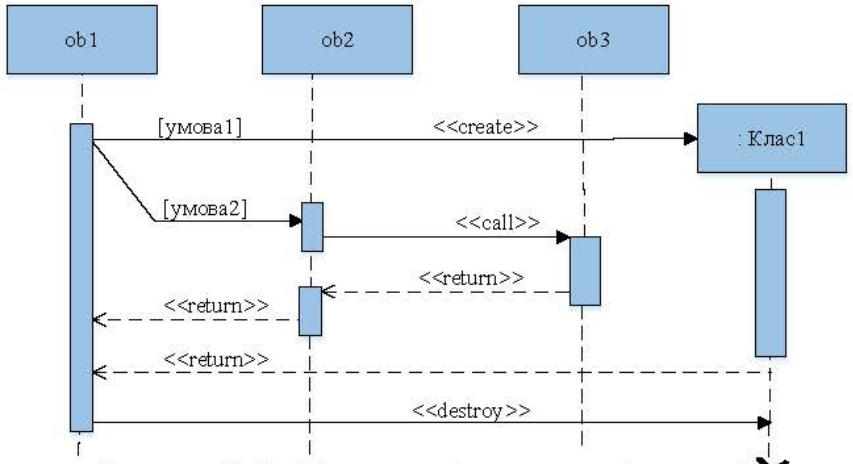

Рисунок 11.3 – Приклад діаграми послідовності

Діаграми станів і діяльності (activity diagram) призначені для моделювання поведінки системи. Діаграми діяльності (рис. 11.4) використовуються при моделюванні бізнес-процесів, технологічних процесів, послідовних і паралельних обчислень.

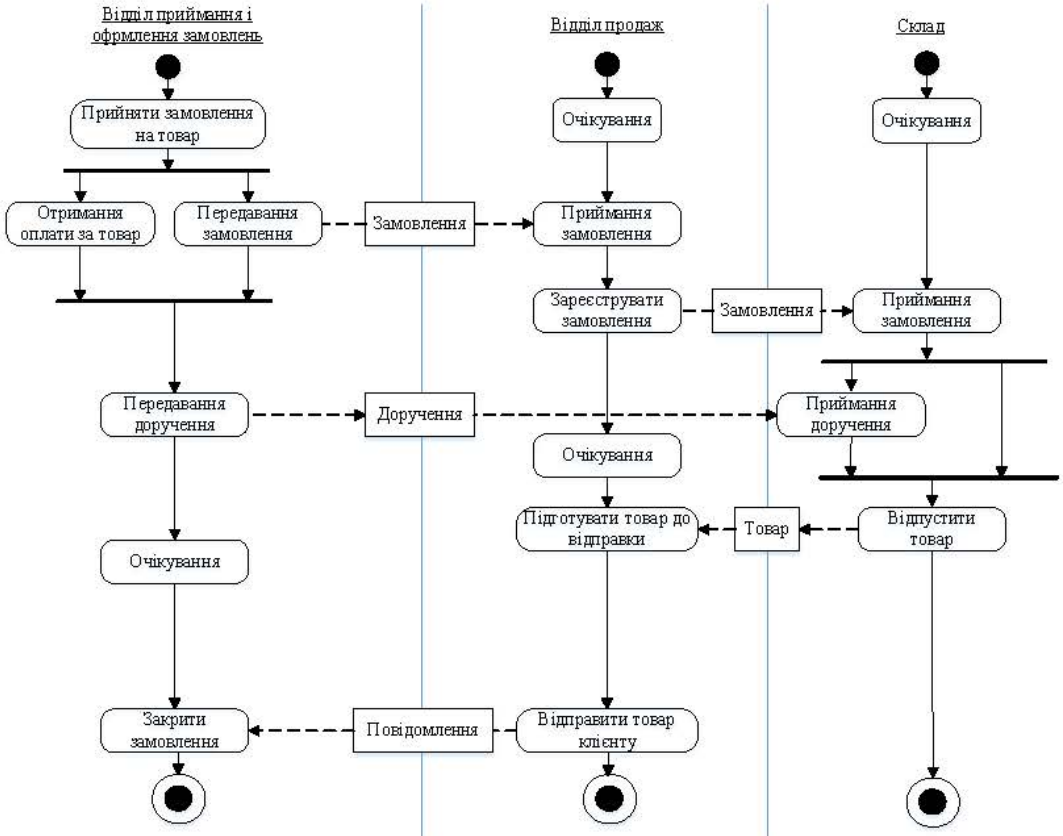

Рисунок 11.4 – Фрагмент діаграми діяльності компанії

При цьому кожна дія розділяється на фундаментальні процеси. На діаграмі діяльності управління здійснюється через:

– потоки управління;
– визначені потоки даних.

Діаграма діяльності близька за змістом до блок-схеми алгоритму, однак має ширші можливості опису систем. Зокрема, на діаграмах діяльності показують паралельне функціонування окремих об'єктів системи (діяльність кожного об'єкта показується на окремій «доріжці») і обмін даними (сигналами, впливами) між ними.

Діаграма розгортання (deployment diagram, рис. 11.5) в UML моделює фізичне розгортання артефактів на вузлах. Наприклад, щоб описати веб-сайт діаграма розгортання повинна показувати, які апаратні компоненти («вузли») існують (наприклад, веб-сервер, сервер бази даних, сервер-додаток), які програмні компоненти («артефакти») працюють на кожному вузлі (наприклад, веб-додаток, база даних), і як різні частини цього комплексу з'єднуються один з одним (наприклад, JDBC, REST, RMI).

Рисунок 11.5 – Діаграма розгортання для системи мобільного доступу до корпоративної бази даних

Вузли подаються у вигляді прямокутних паралелепіпедів з артефактами, розташованими в них, зображеними у вигляді прямокутників. Вузли можуть мати підвузли, які подаються як вкладені прямокутні паралелепіпеди. Один вузол

діаграми розгортання може концептуально подавати множину таких фізичних вузлів, як кластер серверів баз даних.

Подібно до діаграми послідовності, *діаграми кооперації* (collaboration diagram, рис. 11.6) відображають потік подій в конкретному сценарії варіанта використання. Головна особливість діаграм кооперації полягає в можливості графічно подати не тільки послідовність взаємодії, але й усі структурні відносини між об'єктами, які беруть участь у цій взаємодії.

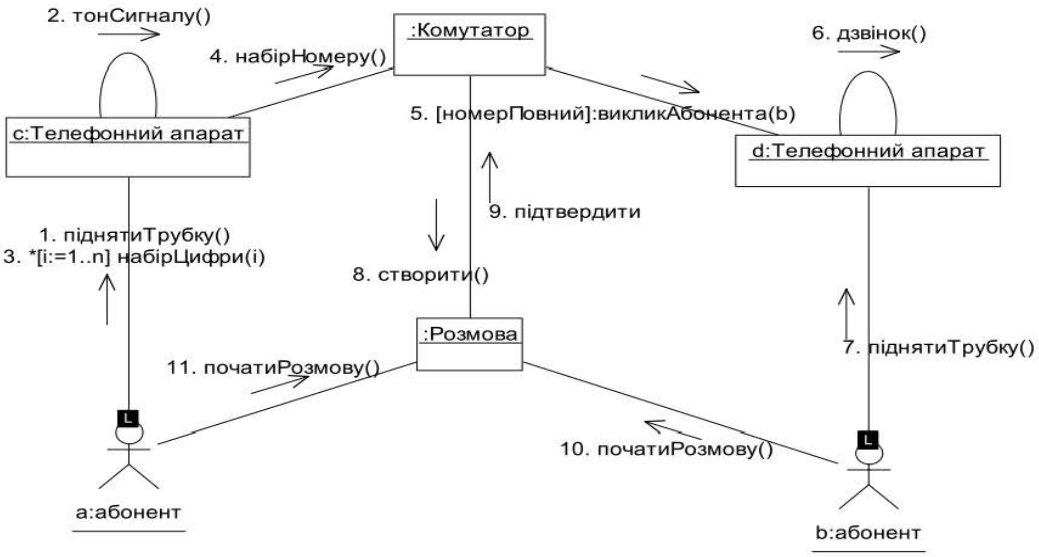

Рисунок 11.6 – Діаграма кооперації моделі початку телефонної

Перш за все, на діаграмі кооперації у вигляді прямокутників зображуються об'єкти, які беруть участь у взаємодії, що містять ім'я об'єкта, його клас і, можливо, значення атрибутів. Далі, як і на діаграмі класів, вказуються асоціації між об'єктами у вигляді різних з'єднувальних ліній. При цьому можна явно вказати імена асоціації та ролей, які грають об'єкти в даній асоціації. Додатково можуть бути зображені динамічні зв'язки – потоки повідомлень. Вони подаються також у вигляді з'єднувальних ліній між об'єктами, над якими розташовується стрілка з вказанням напрямку, імені повідомлення і порядкового номера в загальній послідовності ініціалізації повідомлень.

Усі розглянуті раніше діаграми відображали концептуальні аспекти побудови моделі системи і належали до логічного рівня подання. *Діаграма компонентів* (component diagram, рис. 11.7) описує особливості фізичного подання системи. Діаграма компонентів дозволяє визначити архітектуру розроблюваної системи, встановивши залежності між програмними компонентами, в ролі яких може виступати вихідний, бінарний і виконавчий коди. Основними графічними елементами діаграми компонентів є компоненти, інтерфейси і залежності між ними. Діаграма компонентів розробляється для цілей:

– візуалізації загальної структури вихідного коду програмної системи;

– специфікації виконуваного варіанта програмної системи;

– забезпечення багаторазового використання окремих фрагментів програмного коду;

– подання концептуальної і фізичної схем баз даних.

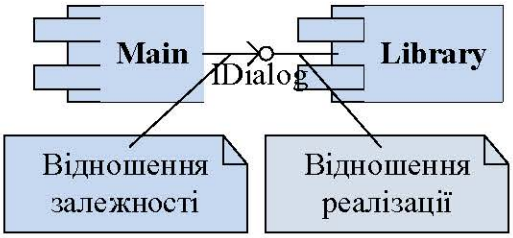

Рисунок 11.7 – Фрагмент діаграми компонентів

Діаграма компонентів забезпечує узгоджений перехід від логічного подання до конкретної реалізації проекту у формі програмного коду. Одні компоненти можуть існувати тільки на етапі компіляції програмного коду, інші – на етапі його виконання. Діаграма компонентів відображає загальні залежності між компонентами, розглядаючи останні як класифікатори.

Наприклад, зображений фрагмент діаграми компонентів (рис. 11.7) відображає інформацію про те, що компонент Main залежить від імпортованого інтерфейсу IDialog, який реалізується компонентом Library. У цьому випадку для компонента Library інтерфейс IDialog є експортованим.

11.2.2 Мова моделювання бізнес-процесів BPML

У 2006 р. міжнародний консорціум з розробки описів процесів у комп'ютерних технологіях (Object Management Group, Inc. – OMG) опублікував остаточну версію мови моделювання бізнес-процесів BPML (Business Process Modeling Language), на основі якої наразі створені і використовуються декілька похідних засобів моделювання, зокрема BPMN (Business Process Model and Notation).

Специфікація BPMN описує умовні позначення для відображення бізнес-процесів у вигляді діаграм бізнес-процесів. BPMN орієнтована як на технічних фахівців, так і на бізнес-користувачів. Для цього мова використовує базовий набір інтуїтивно зрозумілих елементів, які дозволяють визначати складні семантичні конструкції. Крім того, специфікація BPMN визначає, як діаграми, що описують бізнес-процес, можуть бути трансформовані у виконувані моделі мовою BPML. Специфікація BPMN 2.0 також може виконуватись та переноситись (тобто процес, нарисований в одному редакторі від одного виробника,

може бути виконаний на движку бізнес-процесів зовсім іншого виробника, за умови, якщо вони підтримують BPMN 2.0).

Основна мета BPMN – створення стандартного набору умовних позначень, зрозумілих усім бізнес-користувачам. Бізнес-користувачі – це також бізнес-аналітики, які створюють і поліпшують процеси, технічні розробники, відповідальні за реалізацію процесів, і менеджери, що стежать за процесами і керують ними. Отже, BPMN призначена бути сполучною ланкою між фазою дизайну бізнес-процесу і фазою його реалізації.

Використання BPMN

Моделювання бізнес-процесів використовується для донесення широкого спектра інформації до різних категорій користувачів. Діаграми бізнес-процесів дозволяють описувати наскрізні бізнес-процеси, але в той же час допомагають швидко розуміти процес і легко орієнтуватися в його логіці. У наскрізний BPMN-моделі можна виділити три типи підмоделей:

- окремі (внутрішні) бізнес-процеси;
- абстрактні (відкриті) бізнес-процеси;
- процеси взаємодії (глобальні).

Моделювання в BPMN здійснюється за допомогою діаграм з невеликим числом графічних елементів. Це допомагає користувачам швидко розуміти логіку процесу. Виділяють чотири основні категорії елементів:

- об'єкти потоку управління: події, дії та логічні оператори;
- з'єднувальні об'єкти: потік управління, потік повідомлень та асоціації;
- ролі: пули і доріжки;
- артефакти: дані, групи і текстові анотації.

Елементи цих чотирьох категорій дозволяють будувати найпростіші діаграми бізнес-процесів. Об'єкти потоку управління поділяються на три основні типи: події (events), дії (activities) і логічні оператори (gateways).

Елементи потоку є ключовими для формування моделі процесу. Події використовуються для позначення початку і завершення процесу. Крім того,можуть бути проміжні події (Intermediate). Як правило, подія має причину (так званий тригер) і зображується в BPMN у вигляді кола з вільним центром, призначеним для відображення різних тригерів або результатів («Повідомлення», «Таймер», «Помилка», «Скасування», «Компенсація», «Правило », «Зв'язок», «Множинний», «Завершення»). Для кожного тригера є відповідне умовне позначення

Події зображуються колом і означають певну подію в світі. Події ініціюють дії або є їх результатами. За розташуванням в процесі події можуть бути класифіковані на початкові (start), проміжні (intermediate) і завершальні (end) (рис. 11.8).

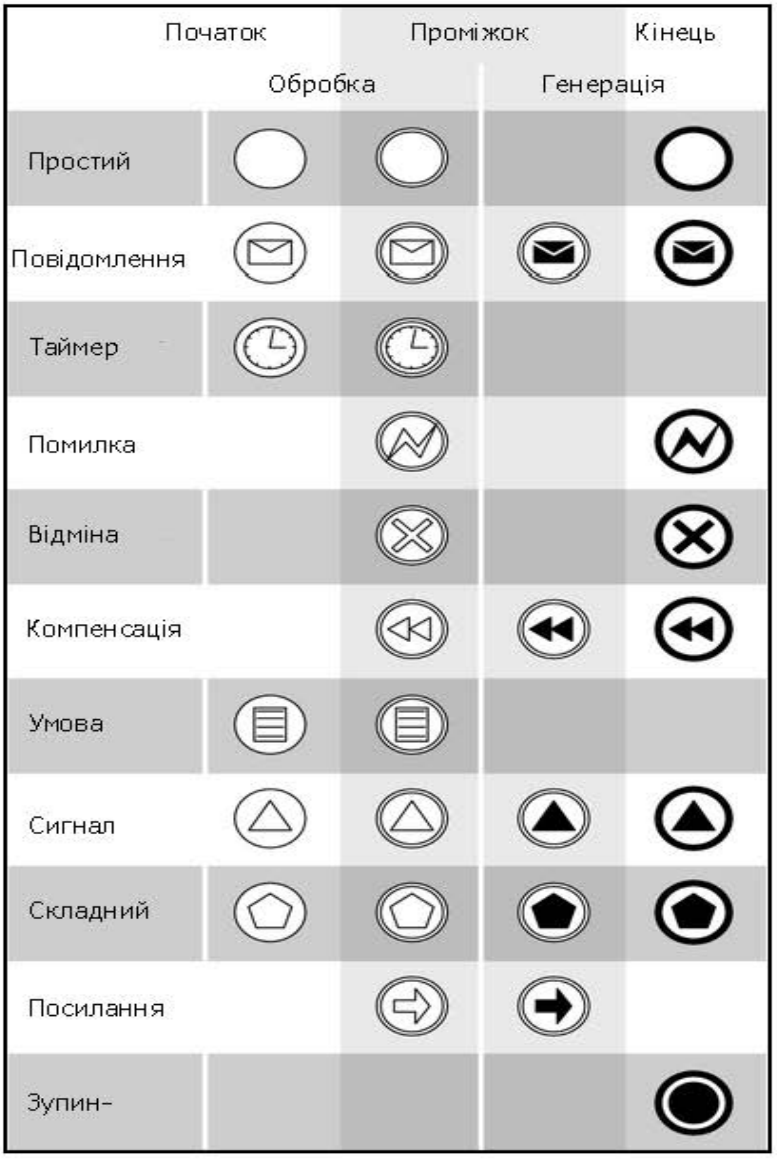

Рисунок 11.8 – Позначення подій в BPMN

Дії зображуються прямокутниками з округленими кутами. Серед дій розрізняють завдання і підпроцеси. Графічне зображення згорнутого підпроцесу – знак плюс біля нижньої межі прямокутника (рис. 11.9).

Логічні оператори зображуються ромбами і є точками прийняття рішень у процесі. За допомогою логічних операторів організовується розгалуження і синхронізація потоків управління в моделі процесу.

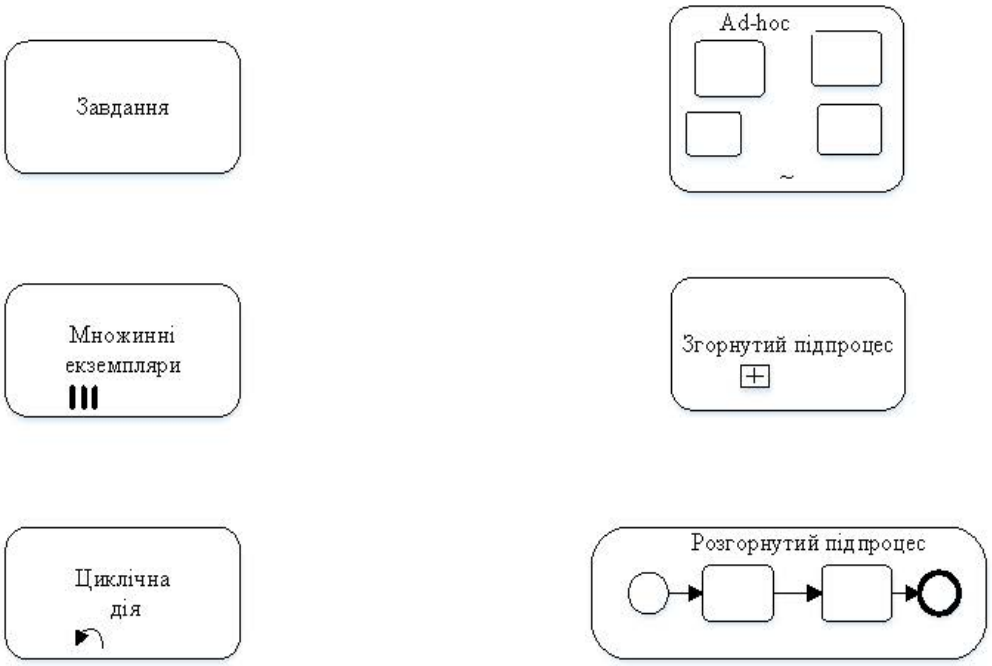

Рисунок 11.9 – Графічне зображення дій (завдання і підпроцеси)

Оператор виключного АБО, що керується даними

Оператор виключного АБО, що керується об’єктами

Оператор виключного АБО

Оператор І

Складений оператор

Фрагмент моделі бізнес-процесу за стандартом BPML наведено на рисунку 11.10.

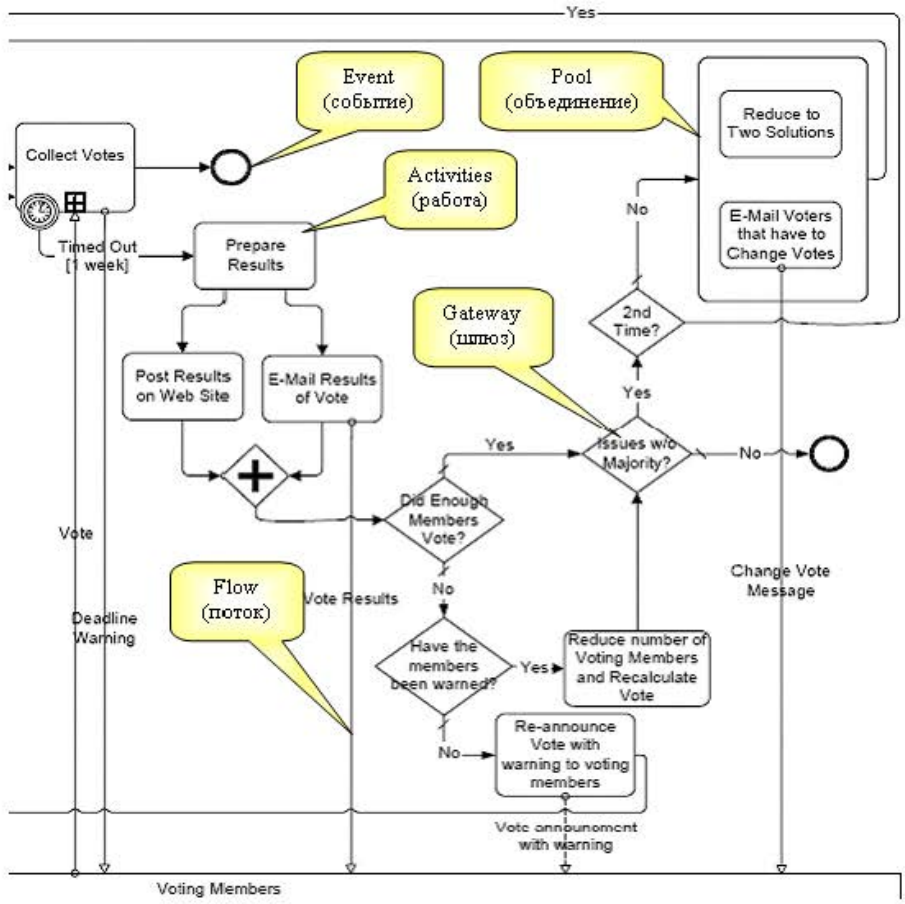

Рисунок 11.10 – Фрагмент моделі бізнес-процесу за стандартом BPML