Міністерство освіти і науки України
Державний університет «Одеська політехніка»

МЕТОДИЧНІ ВКАЗІВКИ
до самостійної роботи
з дисципліни «Бізнес-процеси в кібербезпеці»
другого (магістерський) рівня вищої освіти
спеціальності 125 «Кібербезпека»

Одеса, 2021

Міністерство освіти і науки України
Державний університет «Одеська політехніка»

МЕТОДИЧНІ ВКАЗІВКИ
до самостійної роботи
з дисципліни «Бізнес-процеси в кібербезпеці»
другого (магістерський) рівня вищої освіти
спеціальності 125 «Кібербезпека»

Затверджено
на засіданні кафедри КБПЗ
Протокол № 1 від 27.08.2021 р.

Одеса, 2021

Методичні вказівки до самостійної роботи з дисципліни «Бізнес-процеси в кібербезпеці» для другого (магістерського) рівня вищої освіти спеціальності 125 – Кібербезпека / Укл.: *Лебедєва О.Ю.* – Одеса, 2021. – 42 с.

Укладач: Лебедєва О.Ю., к.т.н., доц.

ЗМІСТ

Вступ

Дисципліна «Бізнес-процеси в кібербезпеці» відповідає освітньо-професійній програмі, навчальному плану підготовки фахівців другого (магістерського) освітньо-професійного рівня вищої освіти за спеціальністю 125 Кібербезпека, і є складовою циклу дисциплін професійної підготовки обов’язкової частини навчального плану.

Дисципліну викладають впродовж першого семестру другого (магістерського) рівня освіти. Згідно навчального плану передбачено підсумковий контроль у вигляді усного екзамену.

На самостійне вивчення винесені визначення й властивості бізнес-процесів, у тому числі розглядаються такі поняття як процесний підхід, еволюція підходів до побудови та використання бізнес-процесів, нотації ERD, Flowchart, UML, знання яких допоможе в розв'язку завдань професійної спрямованості здобувачами спеціальності 125 – Кібербезпека.

Методичні вказівки для самостійної роботи по дисципліні «Бізнес-процеси в кібербезпеці» містять теоретичні знання з побудови та використання бізнес-процесів в нотаціях ERD, Flowchart, практичні знання з використання ArgoUML в якості програмного додатку для побудови бізнес-процесів в нотації UML, а також завдання для самостійної роботи, виконання яких дозволить студентам більш глибоко розібратися в розглянутому матеріалі дисципліни.

1. Визначення бізнес-процесу

Говорячи про моделювання бізнес-процесів, ми користуватимемося термінологією відразу кількох галузей знань, які стосуються економіки, інформатики, моделювання складних систем.

Бізнес-процес визначається як логічно завершений ланцюжок взаємопов'язаних та взаємодіючих видів діяльності, що повторюються (дій, бізнес-функцій, робіт), в результаті яких ресурси підприємства використовуються для переробки об'єкта (фізично або віртуально) з метою досягнення певних вимірних результатів або створення продукції для задоволення внутрішніх чи зовнішніх споживачів (клієнтів). Як клієнт може виступати інший бізнес-процес. У ланцюжок зазвичай входять операції, які виконуються за певними бізнесовими правилами різними елементами організаційної структури підприємства. Під бізнес-правилами розуміють способи реалізації бізнес-функцій у рамках бізнес-процесу, а також характеристики та умови виконання бізнес-процесу.

Термін "моделювання" має два основні значення. По-перше, під моделюванням розуміють процес побудови моделі як деякого уявлення, образу оригіналу, що відображає найважливіші його риси та властивості. Якщо модель вже побудована, то моделювання – це процес дослідження (аналізу) функціонування системи (вірніше, її моделі).

Моделью бізнес-процесу називається його формалізований (графічний, табличний, текстовий, символьний) опис, що відображає реально існуючу або передбачувану діяльність підприємства.

Модель, як правило, містить такі відомості про бізнес-процес:

1) набір складових процесів кроків – бізнес-функцій;
2) порядок виконання бізнес-функцій;
3) механізми контролю та управління в рамках бізнес-процесу;
4) виконавців кожної бізнес-функції;
5) вхідні документи/інформацію, що використовуються кожною бізнес-функцією;
6) вихідні документи/інформацію, що генеруються кожною бізнес-функцією;
7) ресурси, необхідні для виконання кожної бізнес-функції;
8) документацію/умови, що регламентують виконання кожної бізнес-функції;
9) параметри, що характеризують виконання бізнес-функцій та процесу в цілому.

Для моделювання можна використовувати різні способи. Метод, або методологія моделювання, включає послідовність дій, які необхідно виконати для побудови моделі (процедуру моделювання), і нотацію (мова). Мова моделювання має свій синтаксис (умовні позначення різних елементів та правила їх поєднання) та семантику (правила тлумачення моделей та їх елементів).

Теоретично і практично існують різні підходи до побудови та відображенню моделей бізнес-процесів, основними з яких є функціональний та об'єктно-орієнтований. У функціональному підході головним структуроутворюючим елементом є функція (бізнес-функція, дія, операція), в об'єктно-орієнтованому підході – об'єкт.

Бізнес-функція є специфічним типом роботи (операцій, дій), що виконується над продуктами або послугами в міру їх просування в бізнес-процесі. Як правило, бізнес-функції визначаються самою організаційною структурою компанії, починаючи з функцій вищого керівництва через функції управління середнього та нижнього рівня та закінчуючи функціями, покладеними на виробничий персонал.

Функціональний підхід у моделюванні бізнес-процесів зводиться до побудови схеми бізнес-процесу у вигляді послідовності бізнес-функцій, з якими пов'язані матеріальні та інформаційні об'єкти, використовувані ресурси, організаційні одиниці тощо. Перевагою функціонального підходу є наочність послідовності та логіки операцій на бізнес -процесах підприємства, а недоліком – деяка суб'єктивність деталізації операцій.

У ролі об'єктів при моделюванні бізнес-процесів компанії можуть виступати конкретні предмети або реальні сутності, наприклад клієнт, замовлення, послуга тощо. Кожен об'єкт характеризується своїм станом - набором атрибутів, значення яких визначають стан, а також набором операцій для перевірки цього стану. Об’єктно-орієнтований підхід передбачає спочатку виділення об'єктів, та був визначення тих дій, у

яких беруть участь. При цьому розрізняють пасивні об'єкти (матеріали, документи, обладнання), над якими виконуються дії та активні об'єкти (організаційні одиниці, конкретні виконавці, програмне забезпечення), які здійснюють дії. Такий підхід дозволяє більш об'єктивно виділити операції над об'єктами та вирішити задачу про доцільність використання цих об'єктів.

Важливим поняттям будь-якого методу моделювання бізнес-процесів є зв'язки (зазвичай їх зображують у вигляді стрілок у графічних нотаціях). Зв'язки служать для опису взаємин об'єктів та/або бізнес-функцій один з одним. До таких взаємовідносин можуть належати: послідовність виконання в часі, зв'язок за допомогою потоку інформації, використання іншим об'єктом тощо.

Моделі бізнес-процесів застосовуються підприємствами для різних цілей, що визначає тип моделі, що розробляється. Модель бізнес-процесу як наочної, загальнозрозумілої діаграми може бути використаний для навчання нових співробітників їх посадовим обов'язкам, узгодження дій між структурними одиницями компанії, підбору чи розробки компонентів інформаційної системи тощо. Опис за допомогою моделей такого типу існуючих та бажаних бізнес-процесів використовується для оптимізації та вдосконалення діяльності компанії шляхом усунення «вузьких місць», дублювання функцій та ін. Імітаційні моделі бізнес-процесів дозволяють оцінити їх ефективність і подивитися, як виконуватиметься процес із вхідними даними, що не зустрічалися досі в реальній роботі підприємства. Модель бізнес-процесів, що виконуються, можуть бути запущені на спеціальному програмному забезпеченні для автоматизації процесу безпосередньо за моделлю.

Оскільки моделі бізнес-процесів призначені для широкого кола користувачів (бізнес-аналітики, рядові співробітники та керівництво компанії), найбільш широко використовуються моделі графічного типу, в яких відповідно до певної методології бізнес-процес представляється у вигляді наочного графічного зображення – діаграми, що складається в основному з прямокутників та стрілок. Таке уявлення має високу, багатовимірну інформативність для аналітика, яка виявляється у різних властивостях (колір, фон, накреслення тощо. буд.) і атрибутах (вага, розмір, вартість, час тощо.) кожного об'єкта і зв'язку. В останні роки розробники програмних засобів моделювання бізнес-процесів приділяють велику увагу перетворенню графічних моделей на моделі інших видів, зокрема у виконувані, призначенням яких є забезпечення автоматизації бізнес-процесу та інтеграція роботи задіяних у його виконанні інформаційних систем.

Згідно з іншою класифікацією, що прийшла з моделювання складних систем, виділяють такі види моделей бізнес-процесів:

− функціональні, що описують сукупність виконуваних системою функцій та їх входи та виходи;

− поведінкові, що показують, коли та/або за яких умов виконуються бізнес-функції, за допомогою таких категорій, як стан системи, подія, перехід з одного стану в інший, умови переходу, послідовність подій;

− структурні, що характеризують морфологію системи

− склад підсистем, їх взаємозв’язки;

− інформаційні, що відображають структури даних – їх склад та взаємозв'язки.

2. Еволюція підходів до побудови та використання моделей бізнес-процесів

Історія моделювання бізнес-процесів налічує вже майже сторіччя, хоча аж до початку 1990-х рр., коли термін «бізнес-процес» увійшов у широке вживання, говорили про опис того, яким чином організація здійснює свої функції та виконує ті чи інші завдання. Розвиток методів моделювання та автоматизації бізнес-процесів прийнято розділяти на три етапи, або три «хвилі».

Початок першого етапу відносять до 1920-х років. XX ст. і пов'язують з ім'ям Фредеріка Тейлора та його книгою "Принципи наукового управління". У цей період вперше була усвідомлена необхідність досліджувати бізнес-процеси, описувати їх у різних документах та діяти відповідно до цих описів. Опис бізнес-процесів виробляється у текстовому, табличному і графічному вигляді, причому останній дедалі більше формалізується.

У період «першої хвилі» для моделювання бізнес-процесів використовуються блок-схеми, орієнтовані графи, мережі Петрі, методології SADT, IDEF, DFD. Блок-схеми на основі нотації схем алгоритмів, програм, даних та систем, визначеної в ГОСТ 19.701-90 (в англ. літературі – ANSI flowcharts) залишаються і сьогодні найпростішою, але практично важливою формальною графічною мовою моделювання бізнес-процесів. Блок-схеми дозволяють швидко та наочно показати кроки бізнес-процесу у зрозумілій кожному формі, проте їх нотація не передбачає формалізованого опису багатьох деталей процесу, зокрема виконавців бізнес-функцій.

Мережі Петрі - використання цього апарату безпосередньо для опису бізнес-процесів хоч і має своїх прихильників, але не завоювало широкої популярності, тому що його графічна нотація не є інтуїтивно зрозумілою (з нею складно працювати бізнес-аналітикам та менеджерам). Крім того, є процеси, які неможливо описати за його допомогою. Мережі Петрі ляжуть в основу низки мов, спеціально розроблених для моделювання бізнес-процесів у рамках «третьої хвилі».

У 1980-х роках робляться перші спроби автоматизації бізнес-процесів шляхом реалізації у програмному забезпеченні для управління документами – системах електронного документообігу – функцій щодо відстеження послідовності виконуваних дій для автоматизації процедур затвердження та випуску документів. Успіх таких систем надихає розробників програмного забезпечення поширення аналогічного підходу на автоматизацію інших функціональних областей бізнесу. Бізнес-моделювання виділяється у самостійний науково-прикладний напрямок лише к початку 1990 років.

Більшість створених і застосовуваних досі методологій не призначалися спеціально для опису бізнес-процесів, а розроблялися для моделювання складних систем та проектування ПЗ. Вони часто позбавлені певної семантики. Моделі, отримані за допомогою таких методологій, як правило, сприймаються інтуїтивно, і їх інтерпретація може змінюватись в залежності від користувача або області додатків моделі. Ці моделі добре підходять для обговорення бізнес-процесів між співробітниками компанії та керівництвом, для чого вони, власне, і застосовувалися, але не можуть бути основою для роботи інформаційної системи, оскільки неповні та допускають різні інтерпретації.

Початок другого етапу ознаменував вихід книги М. Хаммера та Д. Чампі «Реінжиніринг корпорації: маніфест революції в бізнесі», яка відродила в управлінському середовищі інтерес до опису та аналізу бізнес-процесів з метою їхньої радикальної перебудови – реінжинірингу. РБП передбачає побудову двох моделей бізнес-процесу: як є (англ. as is) і як має бути (англ. to be), а потім впровадження останньої на підприємстві. Як наступний крок у автоматизації бізнес-процесів у 1990-х роках виникають системи управління потоками робіт WfMS (Workflow Management Systems) 2-го покоління, призначені для маршрутизації потоків робіт будь-якого типу в рамках бізнес-процесів компанії. Ці системи забезпечені середовищем розробника, яка теоретично може використовуватися для моделювання різних нестандартних бізнес-процесів, проте на практиці здебільшого впровадження нового або зміна наявного процесу вимагало залучення праці програмістів. Ще більш обмежені можливості з налаштування та зміни процесів надавали підтримуючі керування потоками робіт ERP-системи. Внесення будь-

яких суттєвих змін у бізнес-процес перетворювалося на дуже дорогий і довгостроковий проект з проектування та розробки програмного забезпечення, а моделі бізнес-процесів, побудовані аналітиками, використовувалися для чіткого формулювання вимог, які потім передавались програмістам.

Негнучкість моделей та засобів автоматизації, їх нездатність забезпечити оперативне реагування на постійні зміни у бізнес-середовищі стали основними недоліками систем «другої хвилі», що стимулювали розробку на початку 2000-х років методологій наступного – третього – покоління. Маніфестом «третьої хвилі» у моделюванні бізнес-процесів можна по праву назвати книгу Г. Сміта та П. Фінгара «Управління бізнес-процесами: третя хвиля». На зміну радикальному реінжинірингу приходить системне та «плавне» управління. Мінливість бізнес-процесів, можливість їх коригування у відповідь зміни у бізнесі стають головним критерієм використання інформаційних технологій як засобу, що дозволяє отримати переваги на ринку.

Ідея методологій та інструментів моделювання бізнес-процесів третього покоління полягає в тому, щоб дозволити керівництву та співробітникам компанії створювати та самим впроваджувати нові процеси «на льоту». Автоматизація бізнес-процесів здійснюється за допомогою так званих систем управління бізнес-процесами BPMS (Business Process Management System), які дають можливість безпосередньо та негайно реалізовувати бізнес-процеси відповідно до побудованої формальної моделі і не вимагають розробки будь-якого додаткового програмного забезпечення або його компонентів.

Для розробки зрозумілих машині «виконаних» моделей були потрібні точніші методи моделювання. До таких методів належать мови моделювання бізнес-процесів на основі XML: BPML, BPEL, XPDL. Однак побудова моделей безпосередньо цими мовами ускладнює відсутність графічної нотації. Як мову, що дозволяє побудувати наочну, зрозумілу непідготовленому користувачеві модель, яку потім можна однозначно перетворити на мову (спочатку це був BPML), виступила нотація BPMN.

«Третя хвиля» принесла у моделювання бізнес-процесів прагнення стандартизації. Методології побудови виконуваних моделей розробляються та випускаються організаціями зі стандартизації та міжнародними консорціумами.

3. Процесний підхід в управлінні

Класичний підхід під час побудови моделей - підхід до вивчення взаємозв'язків між окремими частинами моделі передбачає розгляд як відображення зв'язків між окремими підсистемами об'єкта. Такий (класичний) підхід можна використовувати при створенні досить простих моделей.

Системний підхід - це елемент вчення про загальні закони розвитку природи та один із виразів діалектичного вчення.

При системному підході до моделювання систем необхідно насамперед чітко визначити мету моделювання. Оскільки неможливо повністю змоделювати систему, що реально функціонує, створюється модель (система-модель, або друга система) під поставлену проблему. Таким чином, стосовно питань моделювання мета виникає з необхідних завдань моделювання, що дозволяє підійти до вибору критерію та оцінити, які елементи увійдуть у створювану модель.

Важливим для системного підходу є визначення структури системи - сукупності зв'язків між елементами системи, що відображають їхню взаємодію.

Системний підхід дозволяє вирішити проблему побудови складної системи з урахуванням всіх факторів та можливостей, пропорційних їх значущості, на всіх етапах дослідження системи S та побудови моделі M.

Системний підхід означає, що кожна система є інтегрованим цілим навіть тоді, коли вона складається з окремих роз'єднаних підсистем. Отже, основу системного підходу лежить розгляд системи як інтегрованого цілого, причому цей розгляд розробки починається з головного - формулювання мети функціонування.

При структурному підході виявляються склад виділених елементів системи і зв'язок між ними. Сукупність елементів та зв'язків між ними дозволяє судити про структуру системи. Остання, залежно від мети дослідження, може бути описана на різних рівнях розгляду.

При функціональному підході розглядаються окремі функції, тобто. алгоритми поведінки системи, і реалізується функціональний підхід, що оцінює функції, які виконує система, причому під функцією розуміється властивість, що призводить до досягнення мети. Оскільки функція відображає властивість, а властивість відображає взаємодію системи S із зовнішнім середовищем E, то властивості можуть бути виражені у вигляді або деяких характеристик елементів Si(j) і підсистем Si - системи, або системи S в цілому.

Розглянемо поняття «процесний підхід». Для цього звернемося до офіційного стандарту ISO 9000:2000. Під процесом тут розуміється «сукупність взаємозалежних та взаємодіючих видів діяльності, що перетворює входи у виходи, що становлять цінність для клієнта». У визначенні ISO 9000:2000 під процесом можна розуміти будь-яку діяльність, яка використовує певні ресурси (персонал, інформація, матеріальні ресурси, інфраструктура, технології) та служить для отримання певних виходів. Таке визначення процесу є досить загальним. Під нього підпадає будь-який підрозділ організації. Справді, у кожному підрозділі виконуються певні роботи, витрачаються ресурси, використовують устаткування. На виході підрозділу отримуємо певний результат: оброблені документи, готову продукцію, послуги тощо.

Серед переваг процесного підходу можна назвати:

- клієнтоорієнтованість;
- спрямованість на результат;
- гнучкість, більш оперативне прийняття рішень, проведення інновацій у зв'язку із зміною довкілля;
- безперервність керування;
- можливість побудови ефективної системи мотивації, спрямованої на максимальне врахування результатів роботи;
- прозорість за рахунок опису бізнес-процесів, їхньої розумної формалізації.

Ключовим поняттям процесного підходу є поняття "бізнес-процесу".

Моделювання це процес відображення реальної діяльності організації за допомогою спеціальної методології. Важливо розуміти, що процес моделювання суб'єктивний. Справа

в тому, що 80% інформації для формування моделей надходить від співробітників і керівників організації, що інтерв'юються. При цьому суб'єктивними є як думка працівників щодо реального перебігу робіт, так і погляд на процеси аналітика, який проводив інтерв'ю.

Модель «як є» (від «as is» – англ.) – це модель бізнес-процесу, побудована на основі суб'єктивного бачення бізнес-процесу, що існує в організації. При побудові моделі «як є» важливо пам'ятати, по-перше, про суб'єктивність, по-друге, актуальність моделі. Справа в тому, що у великих організаціях постійно відбуваються зміни. Модель процесів може стати неактуальною (невідповідною) вже за кілька місяців після її створення. Тому опис процесу має використовуватися в робочих документах процесу, постійно піддаватися коригування з метою забезпечення відповідності реальної діяльності.

Найважливішим поняттям є ефективність бізнес-процесу. Під ефективністю бізнес-процесу, як правило, розуміється відношення кінцевого результату (виходу) процесу до витрачених на його отримання ресурсів. Ефективність може вимірюватись на основі різних показників. Для кожного показника можуть бути розраховані допустимі чи цільові значення – критерії. Важливо, що практично недостатньо визначити перелік показників оцінки ефективності бізнес-процесу. Важливо також розробити методику їх вимірювання.

З урахуванням ISO 9000:2000 можна запропонувати таку схему розбиття діяльності організації на процеси:

– кількість процесів у організації безпосередньо залежить від чисельності персоналу та структури організації;
– розмежування між процесами у мережі доцільно здійснити по межах великих підрозділів.

Цей підхід пов'язані з тим, що передача результатів діяльності (виходу процесу) підрозділу, зазвичай, формалізована, тобто, визначено специфікацією. За передачу результатів, як і за проведення процесу, відповідає керівник підрозділу (власник процесу).

Мережа процесів є результатом, який може бути використаний для подальших робіт з покращення системи управління організації, наприклад, впровадження системи управління якістю, що відповідає вимогам ISO 9001:2000. Можливе й інше використання мережі процесів - вибір пріоритетних процесів для організації, детальне моделювання та аналіз вибраних процесів, подальша реорганізація (реінжиніринг) процесів. Важливим є те, що як керівники, так і виконавці мають комплексну картину процесів організації та можуть приймати адекватні рішення.

Застосування керувати діяльністю та ресурсами організації системи взаємозалежних процесів може називатися процесним підходом.

При впровадженні процесного підходу до управління застосовуються такі методики:

– створення мережі бізнес-процесів;
– визначення власників бізнес-процесів;
– моделювання (описи) бізнес-процесів;
– регламентації бізнес-процесів;
– керування бізнес-процесами на основі циклу PDCA;
– аудиту бізнес-процесів.

Ключовими моментами для впровадження процесного підходу до управління є:

– визначення та опис існуючих бізнес-процесів та порядку їх взаємодії у загальній мережі процесів організації;
– чіткий розподіл відповідальності керівників за кожен сегмент усієї мережі бізнес-процесів організації;
– визначення показників ефективності та методик їх вимірювання (наприклад, статистичних);
– розробка та затвердження регламентів, що формалізують роботу системи;
– управління ресурсами та регламентами при виявленні відхилень, невідповідностей у процесі або продукти чи зміни у зовнішньому середовищі (у тому числі зміна вимог замовника).

Оцінка функціонування, «якості протікання» бізнес-процесу здійснюється на підставі моніторингу та аналізу показників його результативності та/або ефективності, виявлених невідповідностей. При управлінні бізнес-процесом для його поліпшення доцільно

застосовувати цикл PDCA. Застосування циклу PDCA кожного окремого процесу характеризується наступним набором дій (таблиця 3.1)

Таблиця 3.1 - Набор дій циклу PDCA

Етап	Дії
Плануй (Plan):	Визначай цілі (показники) процесу та процедури роботи, виконання яких учасниками призведе до досягнення його цілей
Роби (Do):	Доведи до учасників процесу цілі (показники) та процедури роботи, забезпеч дотримання процедур роботи учасниками
Перевіряй (Check):	Контролюй показники, дотримання процедур роботи учасниками – аналізуй виявлені невідповідності
Дій (Action):	Покращуй процес за допомогою застосування результатів його аналізу, усунення причин невідповідностей

З допомогою процесного підходи до управління організація має такі можливості.

Можливість 1. Процесний підхід дозволяє оптимізувати систему корпоративного управління, зробити її прозорою для керівництва та здатною гнучко реагувати на зміни зовнішнього середовища. При впровадженні процесного підходу регламентуються:

- порядок планування цілей та діяльності;
- взаємодія між процесами та підрозділами організації;
- відповідальність та повноваження власників процесів та інших посадових осіб;
- порядок дій працівників у позаштатних ситуаціях;
- порядок та форми звітності перед вищим керівництвом;
- система показників, що характеризують результативність та ефективність діяльності організації в цілому та його процесів;
- порядок розгляду результатів діяльності та прийняття управлінських рішень щодо усунення відхилень та досягнення планових показників.

Впровадження в організації процесного підходу насамперед передбачає роботу з опису та регламентації бізнес-процесів, у межах якої:

- проводиться розподіл відповідальності за результати робіт, що входять до складу процесів;
- визначається система взаємодії процесів між собою, а також із зовнішніми постачальниками та споживачами;
- визначається перелік документації, необхідної для функціонування процесів (інструкції, регламенти, положення, методики, посадові інструкції тощо);
- складається графік розробки та впровадження цієї документації;
- встановлюються показники діяльності процесів, способи та форми збору інформації та порядок звітності перед керівниками;
- визначаються межі показників, що характеризують нормальний перебіг процесів;
- встановлюються критерії, якими починається робота з усунення причин відхилення.

Можливість 2. Процесний підхід дозволяє отримати та використовувати систему показників та критеріїв оцінки ефективності управління на кожному етапі виробничого/управлінського ланцюжка. Система показників, побудована в рамках процесного управління, структурується за чотирма напрямками:

- показники результату діяльності окремих процесів та організації в цілому (досягнення запланованих результатів за обсягом, якістю, номенклатурою та строками);
- показники ефективності діяльності окремих процесів та організації загалом (ставлення отриманих результатів до витрат часу, фінансових та інших ресурсів);

– показники товарів, вироблених процесами організації;
– показники задоволеності клієнтів результатами діяльності організації.

При впровадженні процесного підходу розробляється двоступінчаста система показників:

а) показники, за якими власник процесу оцінює результативність та ефективність свого процесу та робіт, що входять до його складу;

б) показники, якими власник процесу звітує перед вищим керівництвом про результати діяльності процесу.

До процесів, які існують в організації, входить також процес управління організацією. Власником цього процесу є генеральний директор. Управління діяльністю організації провадиться на основі звітних показників, які власники процесів передають вищому керівництву.

Можливість 3. Процесний підхід забезпечує впевненість у співзасновників організації у тому, що існуюча система управління націлена на постійне підвищення ефективності та максимальний облік інтересів зацікавлених сторін, оскільки:

– система заснована на вимірі показників діяльності організації, плануванні та досягненні безперервного поліпшення результатів діяльності;
– система спрямована на задоволення потреб п'яти груп осіб, зацікавлених у діяльності організації: співзасновники (інвестори); споживачі над ринком; особа організації; постачальники; суспільство.

Можливість 4. Розроблена та впроваджена система управління бізнес-процесами (СУБП) забезпечує реалізацію в організації процесного підходу відповідно до вимог ISO 9000:2000 та отримання відповідного сертифікату.

Можливість 5. Впровадження процесного підходу до управління та побудова системи менеджменту якості гарантує чітко визначений порядок та відповідальність за розробку, погодження, затвердження та ведення документації.

Можливість 6. Основою процесного підходи до управління є прийняття рішень, засноване на фактах, тому велике значення має у організації інформаційної системи. Інформаційна система, що впроваджується в організації, дозволяє отримувати власникам процесів об'єктивну інформацію для ведення управління в тому випадку, якщо вона будується в рамках єдиної системи управління організацією на основі процесного підходу.

4. Нотація Flowchart

Нотації Flowchart використовуються для представлення алгоритму (сценарію) виконання процесу і дозволяють задати причинно-наслідкові зв'язки та тимчасову послідовність виконання дій процесу. Нотації підтримують декомпозицію на підпроцеси.

Розроблені дві нотації в межах Flowchart:

– нотація "Процедура" (Cross Functional Flowchart);
– нотація "Процес" Процес (Basic Flowchart).

Нотація Cross Functional Flowchart відображає детальний алгоритм виконання бізнес-процесу, а також всіх учасників бізнес-процесу та як вони взаємодіють між собою у рамках Процедури. Доріжка на діаграмі означає посаду, підрозділ та роль. На доріжках Процедури розміщуються події, за які і відповідає посада, підрозділ, роль.

Нотація Basic Flowchart складається з прямокутників (бізнес-процеси), які входять і виходячи стрілки (потоки інформації, документів, ТМЦ). Також у нотації використовуються елементи типу «рішення», які дозволяють робити розгалуження. Для позначення початку виконання всього бізнес-процесу та її закінчення можна використовувати фігури типу «подія». Переваги процесу в простоті та наочності. З її допомогою можна швидко описати кроки бізнес-процесу. Використання Basic Flowchart не вимагає спеціальних знань, т.к. легко сприймається співробітниками з різним рівнем підготовки.

Відмінність між нотаціями "Basic Flowchart" і "Cross-functional Flowchart" полягає в тому, що додатково до графічних символів, що застосовуються в нотації "Basic Flowchart", у нотації "Cross-functional Flowchart" використовуються доріжки (Swim Lanes) виконавців дій процесу. Це дозволяє підвищити наочність діаграми.

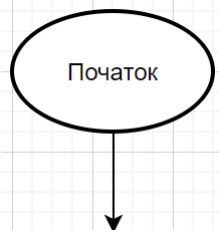
Діаграма читається зверху вниз, у напрямку, яке вказується стрілками. Додатково над кожною лінією може писатися коментар, що пояснює виконання наступної дії, або об'єкт, який задіяний під час виконання цієї функції.

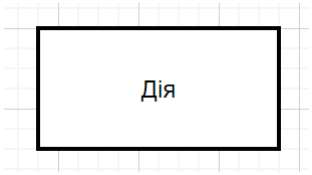
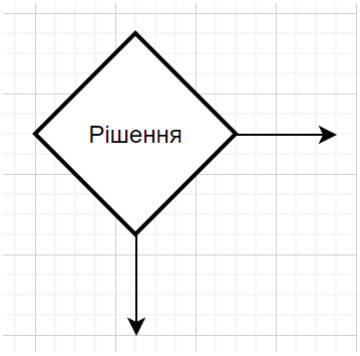
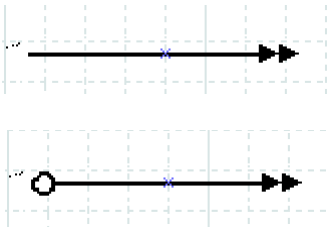
У нотації FlowChart можна відобразити як послідовність виконання функцій, а й їх циклічність, навіщо використовується блок у вигляді ромба з кількома виходами. Якщо виникає неоднозначність виконання наступної функції, наприклад, коли з блоку виходить відразу кілька стрілочок, прийнято над лінією писати умову, за якої виконується наступна функція.

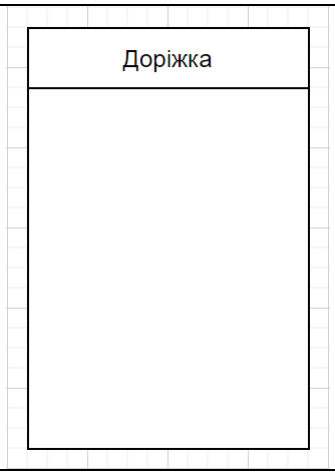
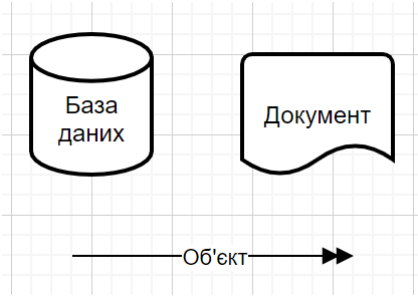
Нотація FlowChart найкраще підходить для опису нижнього рівня процесів, але не призначена для відображення взаємодій на більш високих рівнях управління, тому багато розробників програмних продуктів застосовують цю нотацію спільно з IDEF0 або розробляють власну варіацію для побудови процесів верхнього рівня.

Розглянемо призначення графічних символів нотації Flowchart (таблиця 4.1).

Таблиця 4.1 Графічні символи нотації Basic Flowchart

Графічний символ	Назва	Опис
	Подія	Події відображають стартові точки процесу в нотаціях Flowchart, що призводять до початку виконання процесу, і кінцеві точки, настанням яких закінчується виконання процесу. Початком процесу вважається подія, з якої виходять стрілки передачі управління. Кінець процесу вважається подія, в яку тільки входять стрілки передачі управління.

	Дія	Дія позначається прямокутним блоком. Усередині блоку міститься назва дії. Тимчасова послідовність виконання дій визначається розташуванням дій на діаграмі зверху вниз
	Рішення	Символ "Рішення" означає розгалуження, після якого процес може піти по одному і лише одному альтернативному напрямку залежно від певної умови. Символ "Рішення" може мати один або кілька входів та ряд альтернативних виходів. Якщо символ "Рішення" використовується для перевірки умови, то "Рішення" поміщається на діаграму після символу "Дія", у назві символу "Рішення" вказується умова, що перевіряється, а всі можливі варіанти значення умови показуються стрілками, що виходять. Якщо "Рішення" використовується для позначення дії, то всі можливі варіанти результатів цієї дії показуються стрілками, що виходять.
	Зв'язок передування	Стрілки "Зв'язок передування" позначають передачу управління від однієї дії в інший, тобто. попередня дія має закінчитись перш, ніж почнеться наступне. Стрілка, що запускає виконання дії, зображується зверху, що входить в дію. Стрілка, що позначає передачу управління іншому (іншому) дії, зображується знизу, що виходить з дії
	Потік об'єктів	Стрілки "Потік об'єктів" використовуються у випадках, коли необхідно показати, що з однієї дії об'єкти передаються до іншої, при цьому перша дія не запускає виконання другої. Стрілки "Потік об'єктів" позначаються стрілкою із двома трикутниками на кінці. Якщо позначення джерела об'єкта(ів) неважливе, такий об'єкт показується стрілкою з

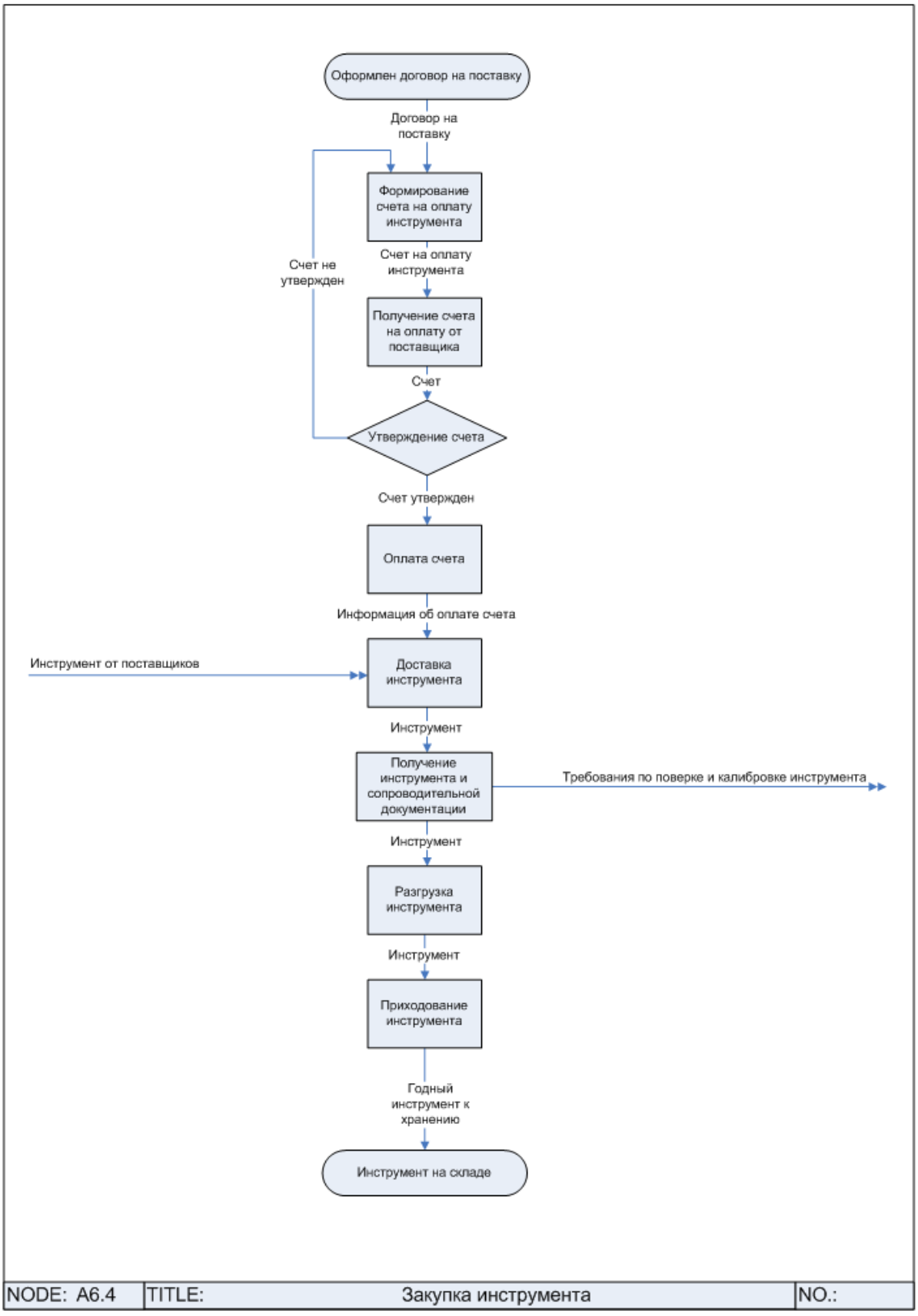
		тунельованим початком Якщо джерелом об'єкта(ів) є одна з процесів процесу, то такий об'єкт показується за допомогою стрілки, що виходить з дії-джерела і входить в дію-споживач, для виконання якого необхідний об'єкт
	Доріжки (символ діаграми процесу в нотації Cross-functional Flowchart)	Доріжки призначені для відображення виконавців дій та підпроцесів (оргодиниць чи об'єктів діяльності). Усередині блоку міститься найменування виконавця.
	Програмні продукти, Бази даних, Матеріальні об'єкти, Інше	Використовується для відображення на діаграмі програмних продуктів, баз даних, матеріальних об'єктів, об'єктів довідника Інше, що супроводжують виконання процесу. Поряд зі стрілкою розміщується найменування об'єкта діяльності.

Діаграма процесу в нотації Cross-Functional Flowchart ділиться оргодиницями на доріжки, в яких розміщуються дії. Над доріжками оргодиниць показується поле з назвою процесу в нотації Cross-functional Flowchart, ліворуч від першої доріжки знаходиться службове поле діаграми. Доріжки оргодиниць на діаграмі можна розташувати горизонтально або вертикально.

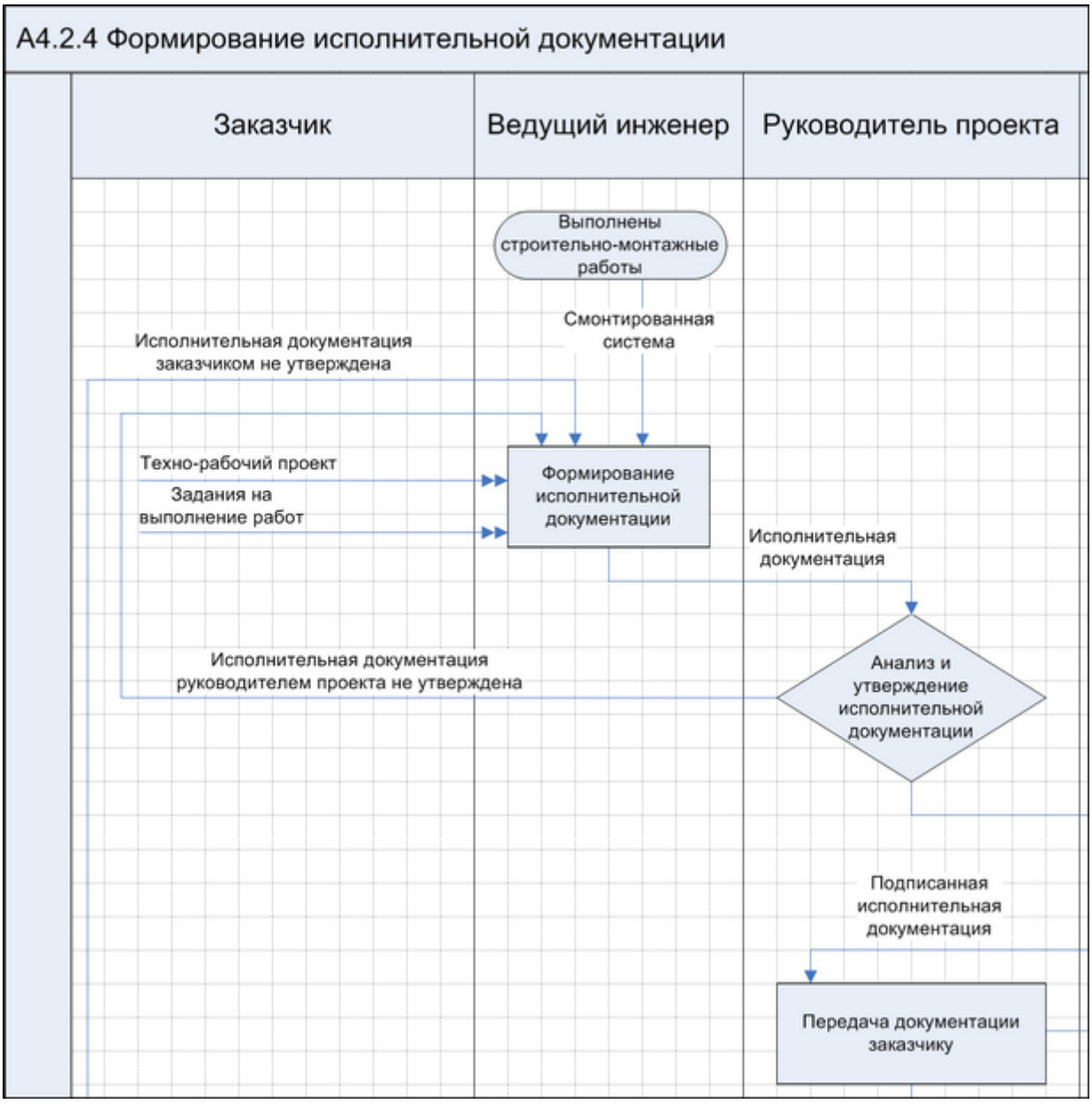
Події на доріжках Процедури пов'язані між собою інформаційними чи матеріальними потоками.

На Процедурі можуть використовуватися рішення (умови) для розгалуження бізнес-процесу.

Приклад процесу «Закупівля інструменту» в нотації Basic Flowchart



Приклад процесу «Формування виконавчої документації» в нотації Cross Functional Flowchart.


A4.2.4 Формирование исполнительной документации
Заказчик
Ведущий инженер
Руководитель проекта
Выполнены строительно-монтажные работы
Смонтированная система
Исполнительная документация заказчиком не утверждена
Техно-рабочий проект
Задания на выполнение работ
Формирование исполнительной документации
Исполнительная документация
Исполнительная документация руководителем проекта не утверждена
Анализ и утверждение исполнительной документации
Подписанная исполнительная документация
Передача документации заказчику

5. Діаграми «сутність-зв'язок» (ERD)

У проектуванні структури бази даних застосовується метод – так зване семантичне моделювання. Семантичне моделювання є моделювання структури даних, спираючись на зміст цих даних. Як інструмент семантичного моделювання використовуються різні варіанти діаграм сутність-зв'язок (ER – Entity-Relationship).

Діаграми «сутність-зв'язок» (ERD) призначені для розробки моделей даних та забезпечують стандартний спосіб визначення даних та відносин між ними. Ці діаграмні техніки використовуються, перш за все, для проектування реляційних баз даних (хоча також можуть успішно застосовуватися і для моделювання ієрархічних і мережевих баз даних).

Перший варіант моделі сутність-зв'язок був запропонований 1976 р. Пітером Пін-Шен Ченом. Як основну роботу прийнято розглядати дослідження Пітера Чена «Модель сутність-зв'язок — напрямок до уніфікованого представлення даних» (The Entity Relationship Model — Toward a Unified View of Data). Саме після цієї публікації нотація ERD набула загальної популярності і міцно увійшла як у наукове, так і в практичне застосування. Надалі багатьма авторами були розроблені свої варіанти подібних моделей (нотація Мартіна, нотація IDEF1X, нотація Баркера, нотація Ч. Бахмана та інші). Крім того, різні програмні засоби, що реалізують ту саму нотацію, можуть відрізнятися своїми можливостями. По суті, всі варіанти діаграм суть-зв'язок виходять з однієї ідеї – рисунок завжди наочніший за текстовий опис. Всі такі діаграми використовують графічне зображення сутностей предметної області, їх властивостей (атрибутів) та взаємозв'язків між сутностями.

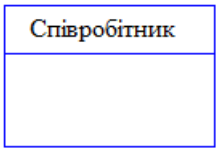
Діаграми "сутність-зв'язок" включають:

– сутності;
– атрибути;
– зв'язки.

Розглянемо роботу з ER-діаграмами близько до нотації Баркера як досить легкої у розумінні основних ідей.

Сутність – це клас однотипних об'єктів, інформація про які має бути врахована в моделі. Кожна сутність повинна мати найменування, виражене іменником в однині. Прикладами сутностей може бути такі класи об'єктів як «Постачальник», «Співробітник», «Накладна».

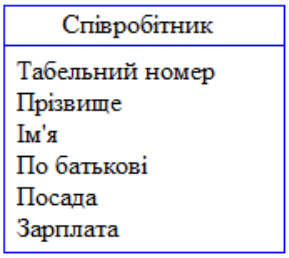
Кожна сутність моделі зображується у вигляді прямокутника з найменуванням:



Екземпляр сутності – це конкретний представник цієї сутності. Наприклад, представником сутності «Співробітник» може бути «Співробітник Іванов». Екземпляри сутностей мають бути помітні, тобто сутності повинні мати деякі властивості, унікальні для кожного екземпляра цієї сутності.

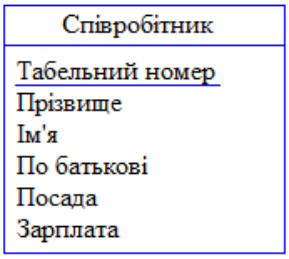
Атрибут сутності – це іменована характеристика, яка є певною властивістю сутності. Найменування атрибута має бути виражене іменником в однині (можливо, з прикметниками, що характеризують). Прикладами атрибутів сутності «Співробітник» можуть бути такі атрибути як «Табельний номер», «Прізвище», «Ім'я», «По батькові», «Посада», «Зарплата» тощо.

Атрибути зображуються в межах прямокутника, що визначає сутність:



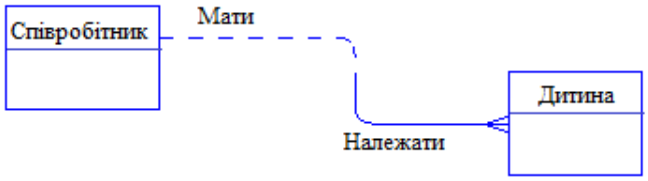
Ключ сутності – це надмірний набір атрибутів, значення яких у сукупності є унікальними для кожного екземпляра сутності. Не надмірність полягає в тому, що видалення будь-якого атрибута з ключа порушує його унікальність. Сутність може мати кілька різних ключів.

Ключові атрибути зображуються на діаграмі підкресленням:



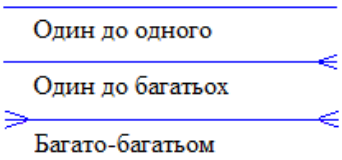
Зв'язок – це асоціація між двома сутностями. Одна сутність може бути пов'язана з іншою сутністю або сама із собою. Зв'язки дозволяють за однією сутністю знаходити інші сутності, пов'язані з нею. Наприклад, зв'язки між сутностями можуть висловлюватися такими фразами – «СПІВРОБІТНИК може мати кілька ДІТЕЙ», «кожен СПІВРОБІТНИК зобов'язаний числитися рівно в одному ВІДДІЛІ».

Графічно зв'язок зображується лінією, що з'єднує дві сутності:



Кожний зв'язок має два кінці та одне або два найменування. Найменування зазвичай виявляється у невизначеній дієслівній формі: «мати», «належати» тощо. Кожне з найменувань належить до кінця зв'язку. Іноді найменування не пишуться через їхню очевидність.

Кожен зв'язок може мати один із наступних типів зв'язку:

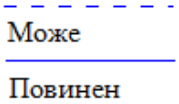


Зв'язок типу один до одного означає, що один екземпляр першої сутності (лівої) пов'язаний з одним екземпляром другої сутності (правою). Зв'язок один до одного найчастіше свідчить про те, що насправді ми маємо всього одну сутність, неправильно розділену на дві.

Зв'язок типу один-багатьом означає, що один екземпляр першої сутності (лівий) пов'язаний з кількома екземплярами другої сутності (правою). Це найчастіше використовуваний тип зв'язку. Ліва сутність (з боку "один") називається батьківською, права (з боку "багато") – дочірньою.

Зв'язок типу багато-багатьом означає, що кожен екземпляр першої сутності може бути пов'язаний з декількома екземплярами другої сутності, і кожен екземпляр другої сутності може бути пов'язаний з декількома екземплярами першої сутності. Тип зв'язку багато-багатьом є тимчасовим типом зв'язку, допустимим на ранніх етапах розробки моделі. Надалі цей тип зв'язку повинен бути замінений двома зв'язками типу одним-багатьом шляхом створення проміжної сутності.

Кожен зв'язок може мати одну з двох модальностей зв'язку:



Модальність «може» означає, що екземпляр однієї сутності може бути пов'язаний з одним або декількома екземплярами іншої сутності, а може бути і не пов'язаний з жодним екземпляром.

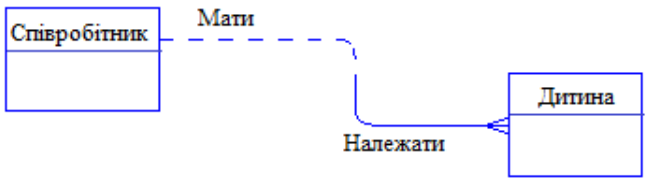
Модальність «повинен» означає, що екземпляр однієї сутності може бути пов'язаний щонайменше ніж із одним екземпляром інший сутності.

Зв'язок може мати різну модальність із різних кінців.

Описаний графічний синтаксис дозволяє однозначно читати діаграми, користуючись наступною схемою побудови фраз:

< Кожен екземпляр СУТНОСТІ 1> < МОДАЛЬНІСТЬ ЗВ'ЯЗКУ > < НАЙМЕНУВАННЯ ЗВ'ЯЗКУ > < ТИП ЗВ'ЯЗКУ > < екземпляр СУТНОСТІ 2>.

Кожен зв'язок може бути прочитаний як зліва направо, так і праворуч наліво. Наприклад для сутностей:



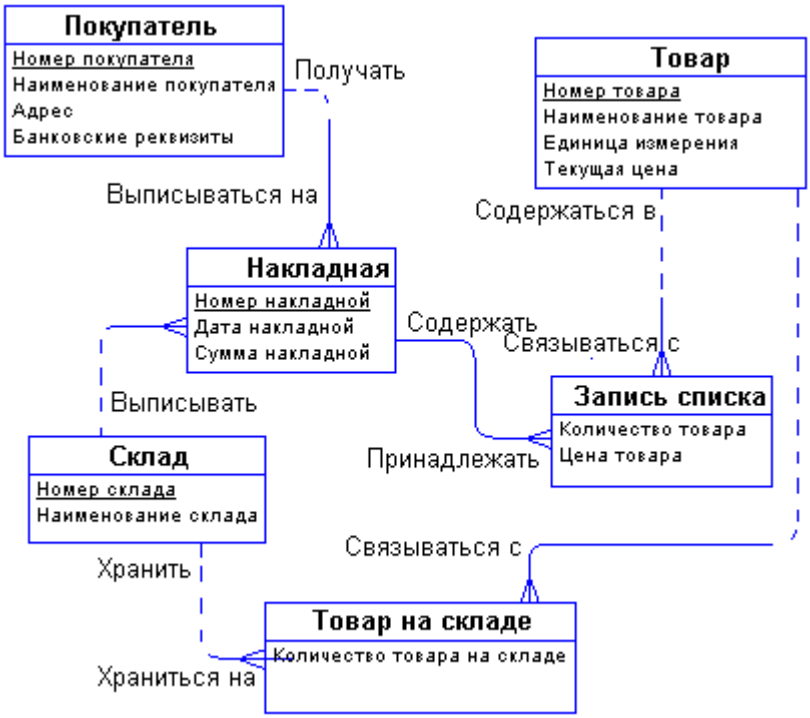
Зліва праворуч: «кожен співробітник може мати кілька дітей».
Праворуч наліво: «Кожна дитина зобов'язана належати рівно одному співробітнику».

Наприклад, нехай постає завдання розробити бізнес-процес на замовлення деякої оптової торгової фірми. Насамперед необхідно вивчити предметну область і процеси, які у ній. Для цього опитуємо співробітників фірми, читаємо документацію, вивчаємо форми замовлень, накладних та ін.

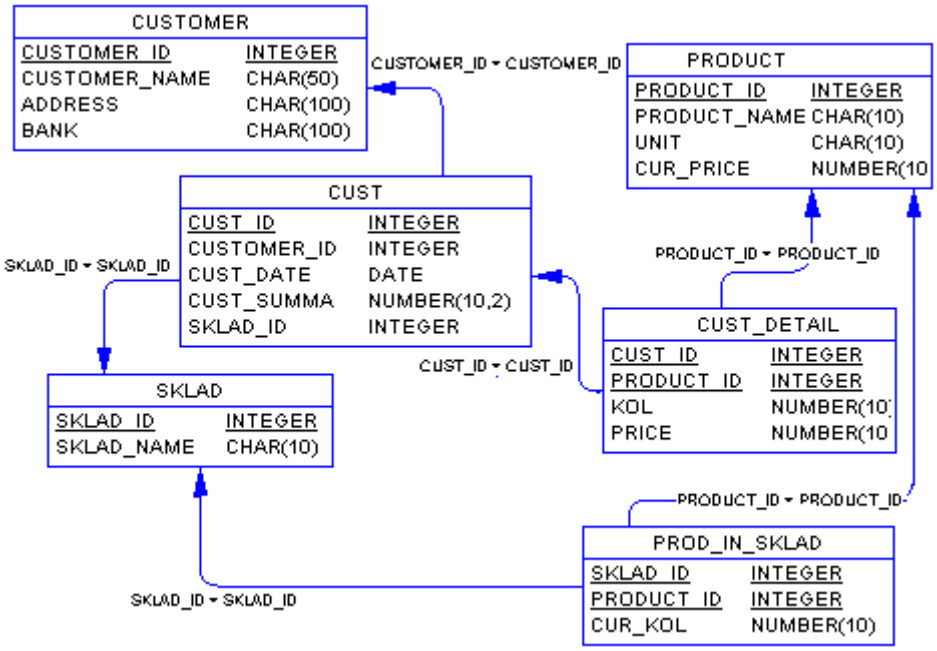
Нехай під час бесіди з менеджером з продажу з'ясувалося, що необхідно відобразити наступні дії:

– Зберігати інформацію про покупців.
– Друкувати накладні на відпущені товари.
– Стежити за наявністю товарів на складі.

Проаналізувавши предметну область та вищеописані дії отримуємо наступний опис поставленої задачі:

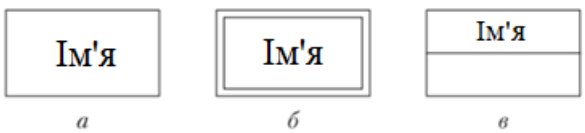


Розроблений вище приклад ER діаграми є прикладом концептуальної діаграми. Це означає, що діаграма не враховує особливостей конкретної СУБД. За цією концептуальною діаграмою можна побудувати фізичну діаграму, яка вже враховуватиме такі особливості СУБД, як допустимі типи та найменування полів та таблиць, обмеження цілісності тощо. Фізичний варіант діаграми може виглядати, наприклад, наступним чином:



Розглянемо опис розглянутих понять у нотації Пітера Чена.

Діаграми «сутність-зв'язок» можуть використовувати три види сутностей: незалежні, залежні та батьківські сутності в ієрархічному зв'язку (рисунок 1.1).



а *б* *в*

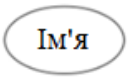
Рисунок 1.1 – Три види сутностей: а – незалежна; б – залежна; в – батьківська в ієрархічному зв'язку

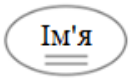
Незалежна сутність використовується для відображення незалежних даних. Такими є дані, які завжди присутні в системі. Незалежна сутність може мати стосунки з іншими сутностями в системі, однак це не є обов'язковою вимогою.

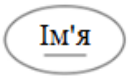
Залежна сутність ілюструє дані, які залежать від інших сутностей системи. Відповідно, залежна сутність обов'язково повинна мати стосунки з іншими сутностями.

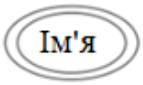
Залежно від складності моделі можна використовувати до п'яти типів різних атрибутів. Виділяються звичайні атрибути, первинні ключі, зовнішні ключі (використовуються в реляційній моделі даних), багатозначні атрибути, отримувані та успадковані атрибути (для ієрархічних зв'язків).

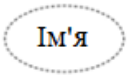
Позначення у нотації Пітера Чена:

Атрибут – 

Первинний ключ – 

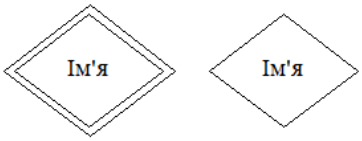
Зовнішній ключ – 

Багатозначний атрибут – 

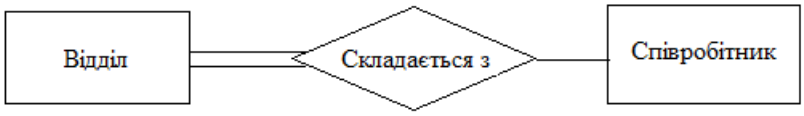
Отримуваний (успадкований) атрибут – 

Атрибут у діаграмах «сутність-зв'язок» може бути *складним*. У такому разі атрибути-компоненти будуть виглядати як приєднані до нього еліпси..

Усі зв'язки у діаграмах «сутність-зв'язок» відображаються у формі ромба, причому ім'я зв'язку вказується всередині. Подвійна рамка ромба позначає наявність ідентифікуючого зв'язку, тоді як одинарна говорить про неідентифікуючий зв'язок.



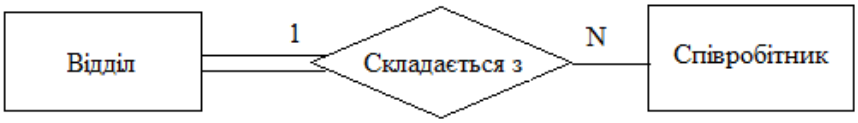
Важлива характеристика зв'язку в ERD це так звана обов'язковість. Поняття зв'язку в ERD має очевидно виражений двосторонній характер.



У цьому випадку означає, що відділ обов'язково має складатися із працівників. Однак кожен конкретний співробітник не обов'язково повинен входити до складу відділу, що розглядається. Подвійна лінія служить відображення обов'язковості, тоді як одинарна лінія ілюструє її відсутність.

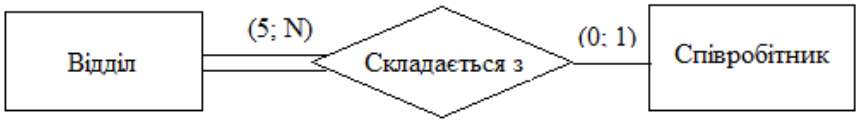
Зв'язок також може характеризуватись власними атрибутами. Атрибути зв'язку позначаються як приєднані еліпси, і загалом логіка їх відображення аналогічна відображенню атрибутів сутностей.

Розглянемо кардинальність (кратність, потужність) у діаграмах «сутність-зв'язок». *Кардинальність* ілюструє, скільки екземплярів однієї сутності може бути пов'язане з будь-яким числом екземплярів іншої сутності. Для будь-яких двох сутностей кардинальність може бути задана у вигляді (N; M), де N – число екземплярів першої сутності, а M – число екземплярів другої.



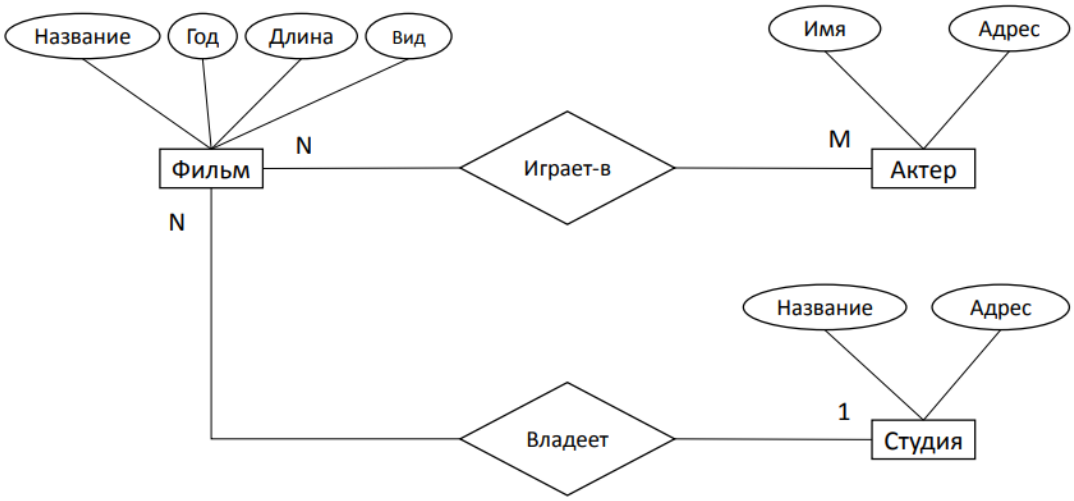
Наведена частина діаграми означає наступне: один відділ може складатися з будь-якої кількості співробітників (включаючи 0, тобто відсутність співробітників). Говорячи суворо, один конкретний екземпляр сутності «Відділ» може складатися з різних екземплярів сутності «Співробітник», причому кількість останніх варіюється від 0 до нескінченності.

Більш складна кардинальність у ERD позначається діапазоном.



Наприклад, відділ маркетингу повинен складатися щонайменше з п'яти екземплярів дочірньої сутності (у разі — маркетологів). Цифри (0,1) біля дочірньої сутності ілюструють її зв'язок з батьківською сутністю та означають, що будь-який співробітник може входити або не входити до якогось конкретного відділу.

Розглянемо приклад діаграми у нотації Пітера Чена.



6. UML. Робота в ArgoUML

Розглянемо основні кроки, які потрібно виконати у програмі ArgoUML.

Створюємо головну діаграму класів. Для цього двічі клацніть значок UntitledModel, з'явиться вміст пакета. Виберіть діаграму класів. За допомогою кнопки Новий пакет та панелі інструментів розмістіть на діаграмі новий пакет. У вкладці Властивості вкажіть ім'я пакета Entities (Сутність) (рисунок 6.1).

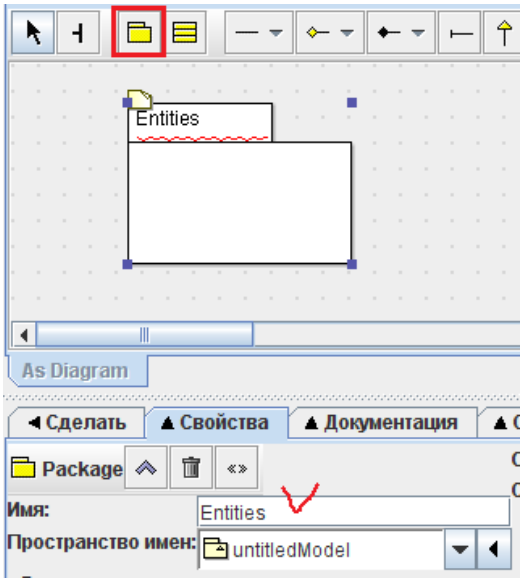
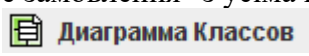


Рисунок 6.1 – Создание пакета в главной диаграмме классов

Аналогічно створюємо пакети Boundaries (Кордони) і Control (Керування).

Створимо діаграми класів для сценарію "Ввести нове замовлення" з усіма класами.

У меню Створити діаграму виберіть Діаграма класів 

. У вкладці Властивості введіть ім'я для нової діаграми класу Add New Order (Введення нового замовлення). Натисніть браузер на цій діаграмі, щоб відкрити її.

Створимо у вікні діаграми класів класи OrderOptions (Вибір замовлення). Для цього на панелі інструментів натиснути кнопку  . Створимо нову операцію у створеному класі (рисунок 6.2).

Рисунок 6.2 – Створення нової операції

У полі Ім'я операції, що створюється, введемо Create. Для вказівки повертається значення void для створеної операції створимо такий тип даних. Для цього клацніть правою кнопкою миші на значку UntitledModel і вибираємо пункт Новий тип даних (рисунок 6.3)

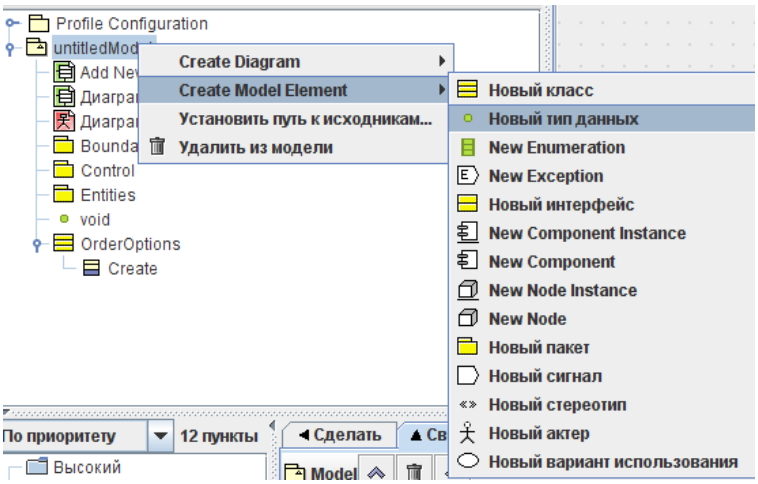


Рисунок 6.3 – Створення нового типу даних

У властивостях створюваної операції вибираємо Parametrs і подвійним клацанням миші натискаємо на return. Потім у полі Тип вибираємо створене значення void (рисунок 6.4).

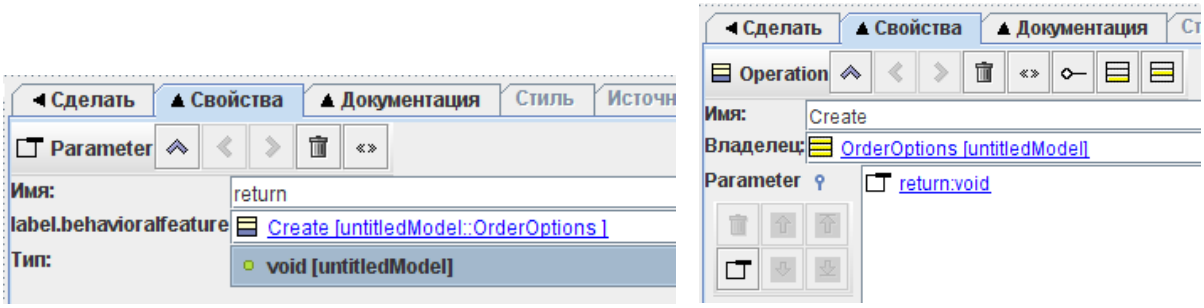
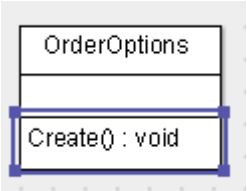


Рисунок 6.4 – Вибір типу значення, що повертається для операції

В результаті отримуємо наступний клас:



Аналогічно створюємо

– клас OrderDetail (Деталі замовлення) та його операції Open():void, SubmitInfo():void, Info():void,
– клас Order (Замовлення) та його операції Create():void, SetInfo():void, GetInfo():void,
– клас OrderMgr (Менеджер замовлень) та його операцію SaveOrder():void
– клас TransactionMgr (Менеджер транзакцій) та його операції SaveOrder():void, Commit():void.

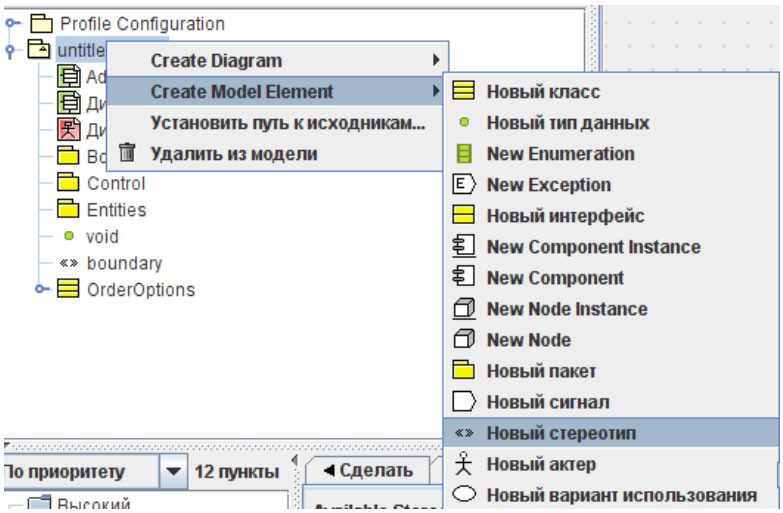
Ім'я класу може передувати стереотип.

Стереотипи є одним із трьох типів механізмів розширюваності в уніфікованій мові моделювання (UML). Вони дозволяють проектувальникам розширювати словник UML для створення нових елементів моделювання, одержуваних з існуючих, але з певними властивостями, які підходять для конкретної проблеми предметної області або для іншого спеціалізованого використання. Графічно стереотип відображається як ім'я, укладене в лапки («», або якщо такі лапки неприпустимі, <<>>) і розташоване над ім'ям іншого елемента.

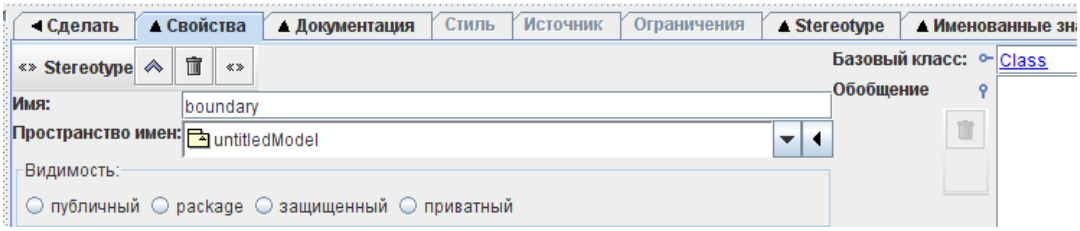
У наведеній нижче таблиці перераховані стандартні стереотипи класів.

Стереотип	Описание
«actor»	Действующее лицо
«auxiliary»	Вспомогательный класс
«enumeration»	Перечислимый тип данных
«exception»	Исключение (только в UML 1)
«focus»	Основной класс
«implementationClass»	Реализация класса
«interface»	Все составляющие абстрактные
«metaclass»	Экземпляры являются классами
«powertype»	Метакласс, экземплярами которого являются все наследники данного класса (только в UML 1)
«process»	Активный класс
«thread»	Активный класс (только в UML 1)
«signal»	Класс, экземплярами которого являются сигналы
«stereotype»	Новый элемент на основе существующего
«type»	Тип данных
«dataType»	Тип данных
«utility»	Нет экземпляров, служба

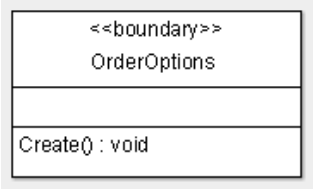
У прикладі імена стереотипів будуть схожі на назви створених раніше пакетів. Додамо стереотипів до класів: натискаємо на значку UntitledModel та вибираємо пункт Новий стереотип.



В імені стереотипу пишемо boundary та вибираємо базовий клас Class.



Тепер натискаємо на клас OrderOptions та в закладці Stereotype вибираємо створений стереотип. В результаті отримаємо:



Доповніть стереотипами, що залишилися класи, як показано на рисунку 6.5.

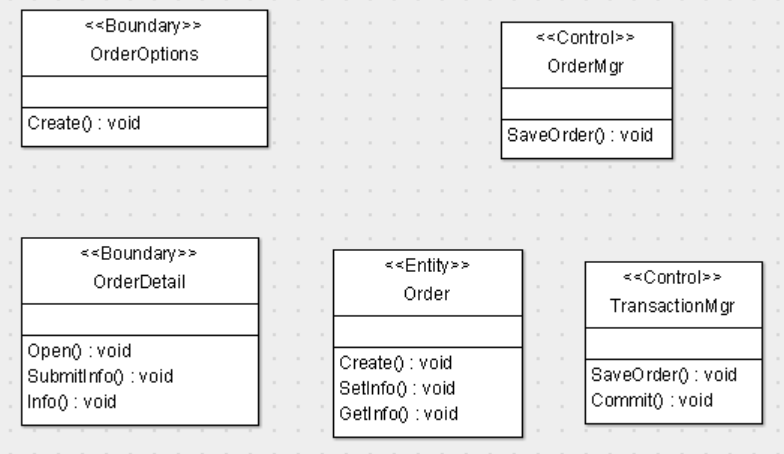


Рисунок 6.5 – Стереотипи та створювані класи

Об'єднаємо класи у пакети. Для цього перетягніть у браузері клас OrderDetail на пакет Boundaries, клас OrderOptions на пакет Boundaries, класи OrderMgr та TransactionMgr на пакет Control та клас Order на пакет Entities.

Додамо діаграму класів до кожного пакету:

1) Двічі клацніть лівою кнопкою на пакеті Boundaries у вікні діаграм.

2) У вікні виберіть Так.

3) У вкладці Властивості введіть ім'я для нової діаграми – MainB.

4) Натисніть діаграму, щоб відкрити його.

5) У браузері виділіть клас OrderOptions (потім OrderDetail), клацніть ПКМ та виберіть опцію Додати в діаграму, а потім натисніть мишею в полі діаграми.

6) Класи відображатимуться на діаграмі.

7) Виконайте ці кроки, щоб помістити класи OrderMgr і TransactionMgr на головну діаграму класів пакета Control (MainC).

8) Виконайте ці ж дії, щоб помістити клас Order на головну діаграму класів пакета Entities (MainE).

Розглянемо кроки по створенню діаграми варіантів використання в ArgoUML.

Запустимо ArgoUML. Відобразиться вікно, яке зображене на рисунку 6.6.

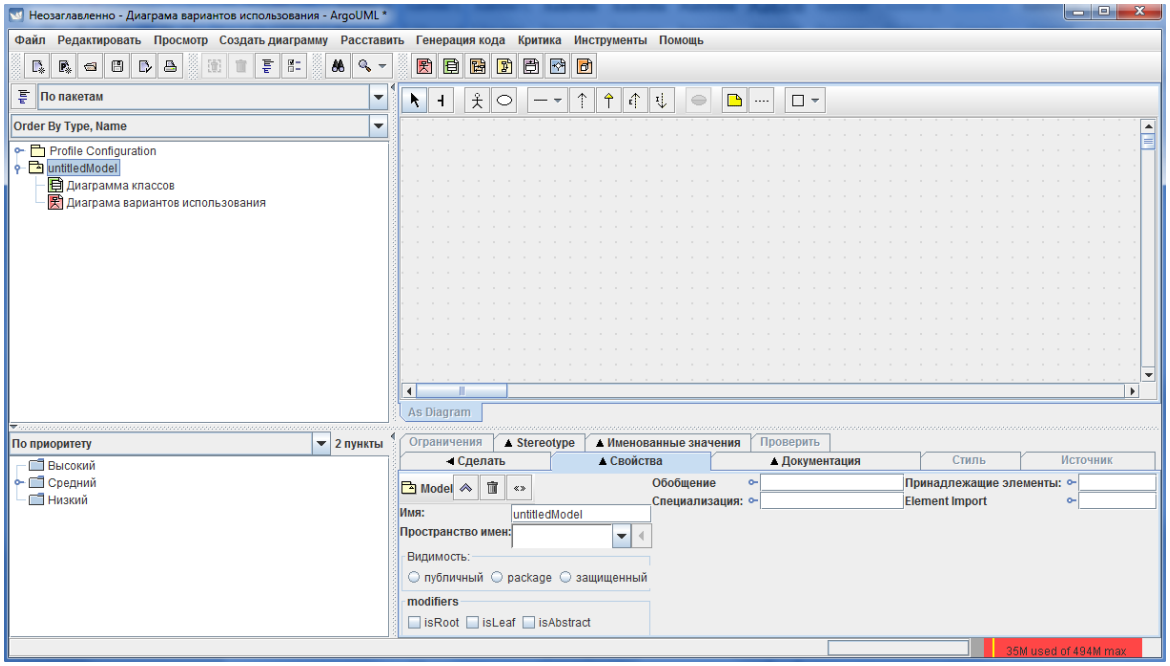


Рисунок 6.6 – Вікно програми ArgoUML

При створенні нового проекту створюється модель з іменем untitledModel. Замість цього імені можна ввести будь-яке інше, наприклад, MathTasks (рисунок 6.7).

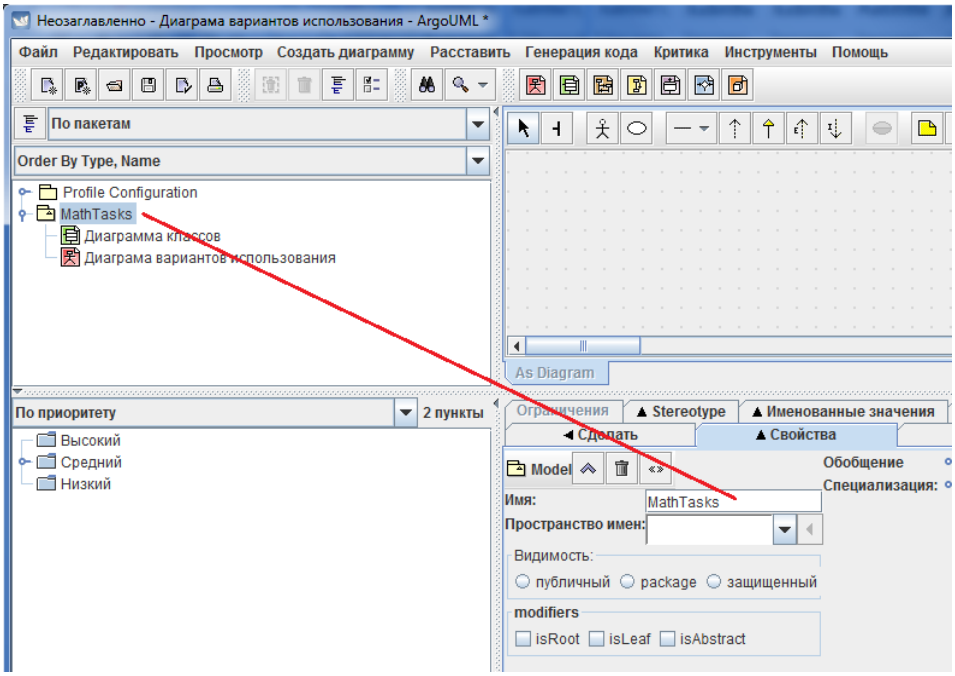


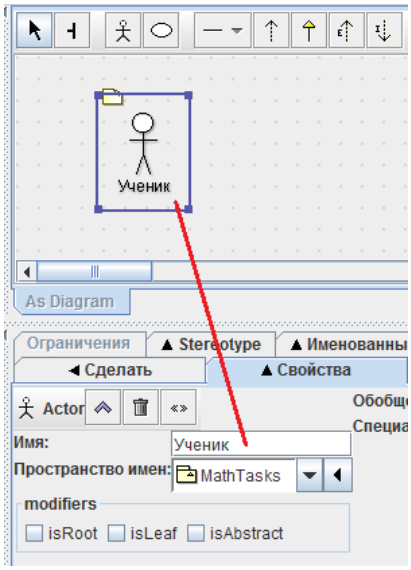
Рисунок 6.7 – Изменение имени модели

Меню Створити діаграму містить перелік діаграм, які можна створити для цього проекту. Також потрібну діаграму можна вибрати на панелі інструментів.

Щоб створити діаграму варіантів використання, на панелі інструментів виберіть піктограму Діаграма Варіантів використання ().

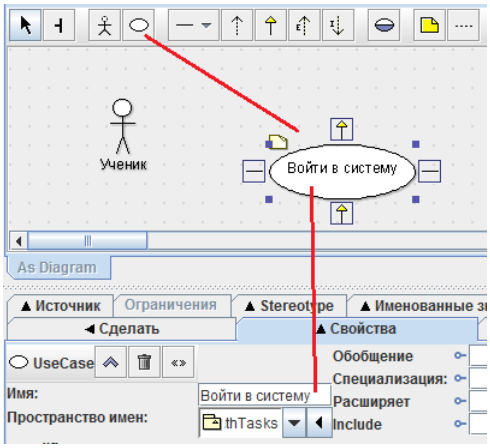
Із завдання видно, що в системі буде два актори: Учень та Вчитель.

Створимо актора. Виберемо піктограму Новий актор (). Потім лівою кнопкою миші натисніть по області, в якій буде зображена діаграма. На ній з'явиться новий актор. Встановимо для актора ім'я. Виділяємо актора та скористаємося вкладкою Властивості та полем Ім'я.



Аналогічно створюємо актора Вчитель.

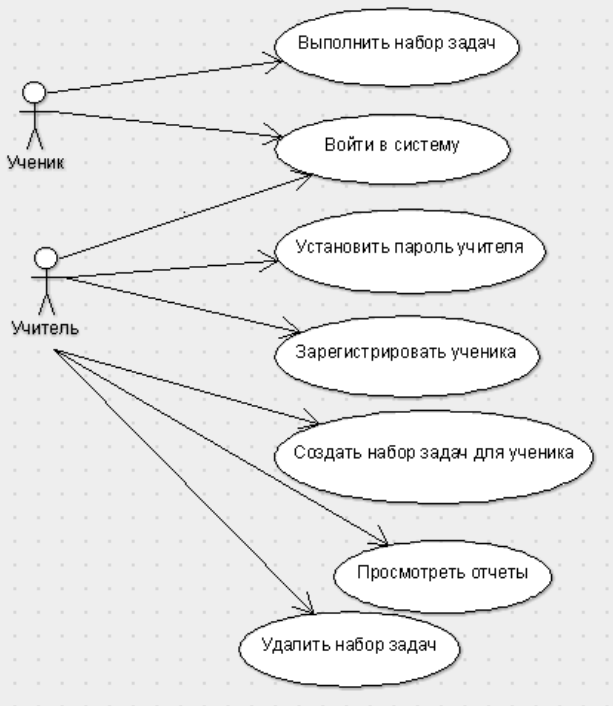
Учень може «Увійти до системи» та «Виконати набір завдань». Це будуть варіанти використання для Учня. Створимо варіант використання. Виберемо піктограму Новий варіант використання (). Потім лівою кнопкою миші натисніть по області, в якій буде зображена діаграма. На ній з'явиться новий варіант використання. Встановимо варіант використання ім'я. Це можна зробити у вкладці Властивості, поле Ім'я.



Аналогічно додається варіант використання «Виконати набір завдань».

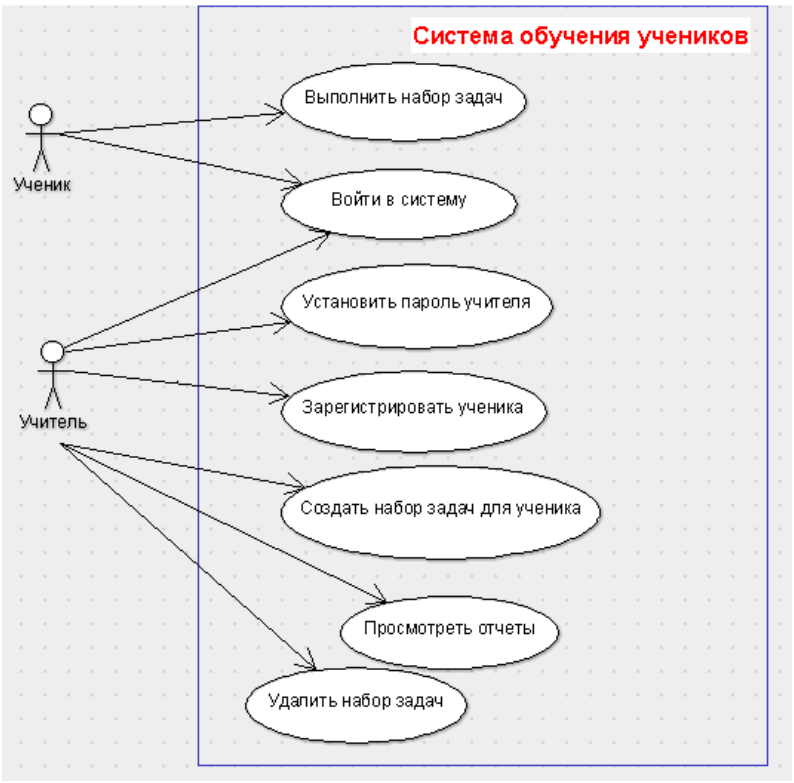
Встановимо відношення асоціації між створеними актором та варіантом використання. Виберемо піктограму Нова асоціація (). Проведемо її із затиснутою лівою кнопкою миші від актора до варіанту використання.

Аналогічно для актора Вчитель створюються варіанти використання "Встановити пароль вчителя", "Зареєструвати учня", "Створити набір завдань для учня", "Переглянути звіти", "Видалити набір завдань". Аналогічно встановити ставлення асоціації між Вчителем та його варіантами використання та додати відношення асоціації до варіанту використання «Увійти до системи».



Укладемо створені варіанти використання суб'єкт. Для цього виберемо піктограму Прямокутник (). Помістимо прямокутник на ділянку діаграми. Змінимо розміри прямокутника так, щоб він охоплював усі варіанти використання. Також у прямокутника властивості Заповнити на вкладці Стиль встановимо значення Не заповнювати.

Щоб створити ім'я суб'єкта, виберіть піктограму Текст (). Помістимо текст на область діаграми. У вкладці Стиль для розділу Лінія встановимо значення Не лінія. Подвійне клацання лівої клавіші миші по елементу Текст для введення тексту.

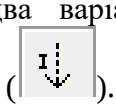


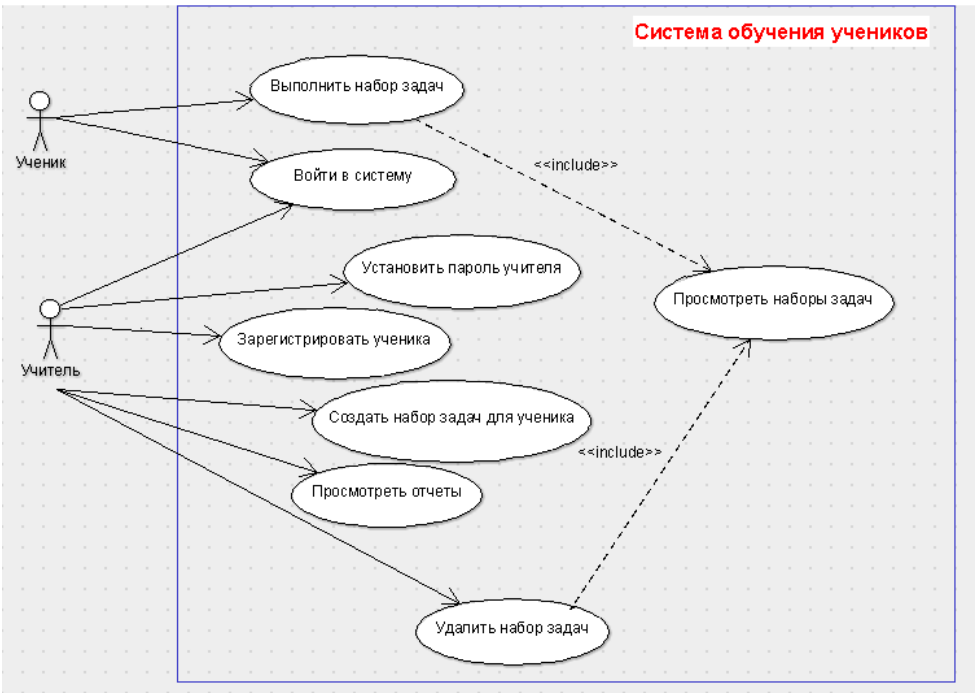
Після того, як визначено набір варіантів використання, можна розпочинати складання сценаріїв. Сценарії повинні описуватися з погляду користувача, при цьому важливо описувати взаємодію користувача з елементами інтерфейсу.

Для збереження проекту вибрати Файл та команду Зберегти проект як. Відкриється вікно, в якому можна встановити ім'я проекту та вибрати тип проекту. Діаграму можна зберегти як графічний файл для цього зайти в маню Файл і викликати команду Зберегти діаграми як графіку.

Незважаючи на простоту наведеного сценарію, у його послідовностях можна знайти дублювання, якщо воно має місце у ваших сценаріях - ви можете виділити деякі фрагменти опису в окремі варіанти використання (які можуть бути як самостійними, так і є лише частиною інших варіантів використання). При цьому між варіантами використання з'явиться ставлення розширення (extend), або включення (include).

Відношення включення вказує на те, що поведінка одного прецеденту включається в деякій точці в інший прецедент як складовий компонент. Особливості включення у тому, що включається прецедент може бути обов'язковим для доповнюваного, тобто, це ставлення задає дуже сильний зв'язок. Наприклад, якщо ми хочемо зобразити на діаграмі той факт, що «Видалення набору завдань» вчителем та «Виконання завдань» учнем не повинно відбуватися без обов'язкового «Перегляду наборів задач», то нам потрібно використовувати відношення включення.

Щоб з'єднати два варіанти використання відношенням включення, виберіть піктограму New include (). Проведемо її із затиснутою лівою кнопкою миші від одного варіанту використання до іншого.

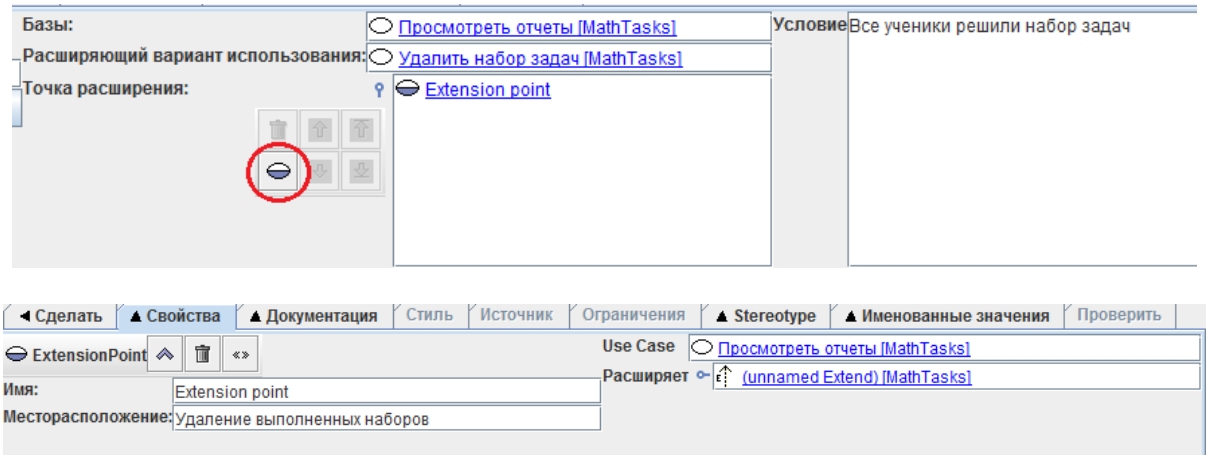


Відношення розширення відображає можливе приєднання одного варіанту використання до іншого в певній точці (точці розширення). При цьому наголошується на тому, що розширюючий варіант використання виконується лише за певних умов і не є обов'язковим для виконання основного прецеденту. На діаграмі такий вид відношення зображується стрілкою, спрямованою до прецедента, що розширюється, в окремому розділі якого може бути описана точка розширення, а умови розширення можуть бути наведені в коментарі з ключовим словом Condition.

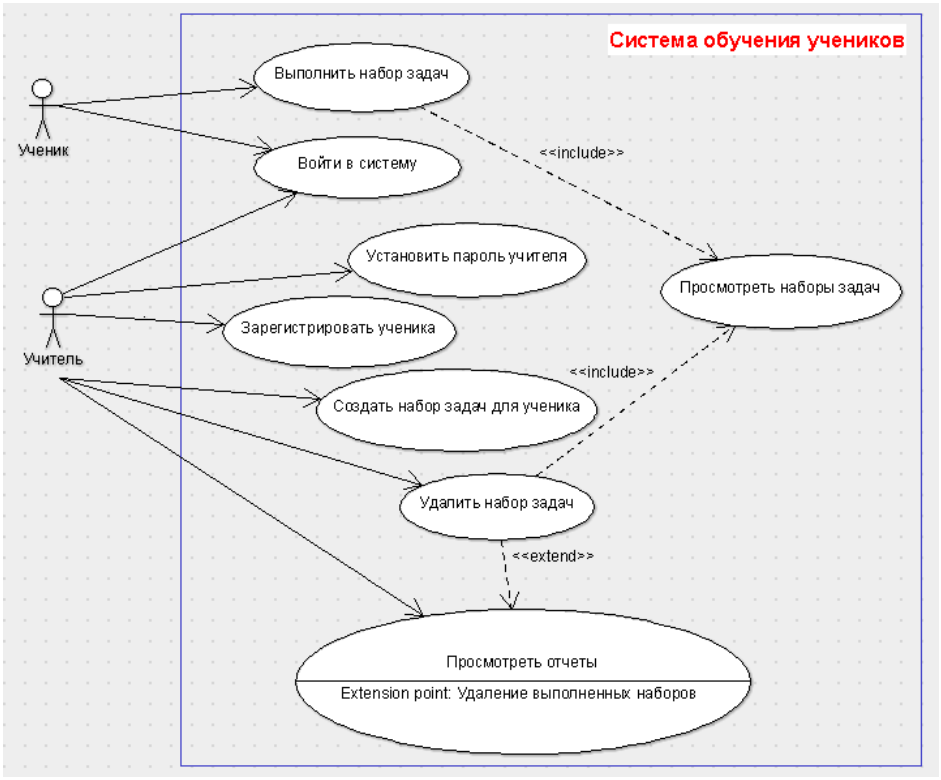
Наприклад, відношення розширення слід застосувати, якщо за технічним завданням потрібна можливість видалення набору завдань у варіанті використання «Перегляд звітів» за умови, що всі учні вирішили цей набір.

Щоб поєднати два варіанти використання ставленням розширення, виберемо піктограму New extend (). Проведемо її із затиснутою лівою кнопкою миші від одного

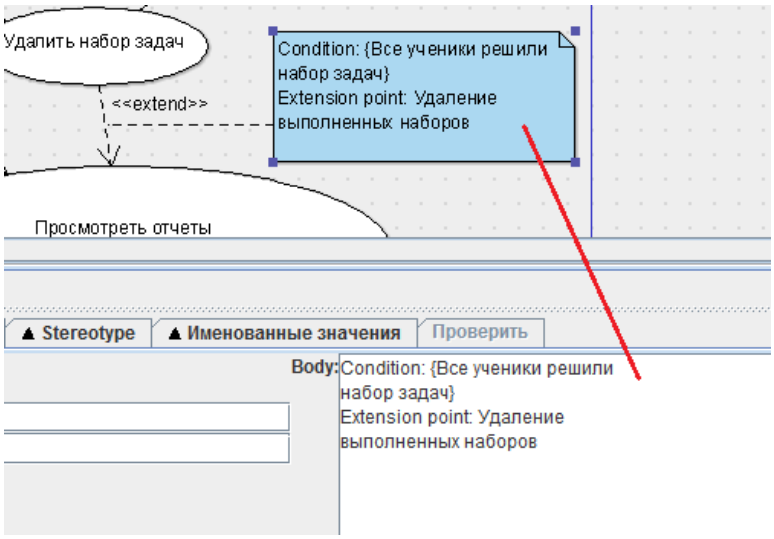
варіанту використання до іншого. У властивостях відношення вказується Точка розширення та Умова.



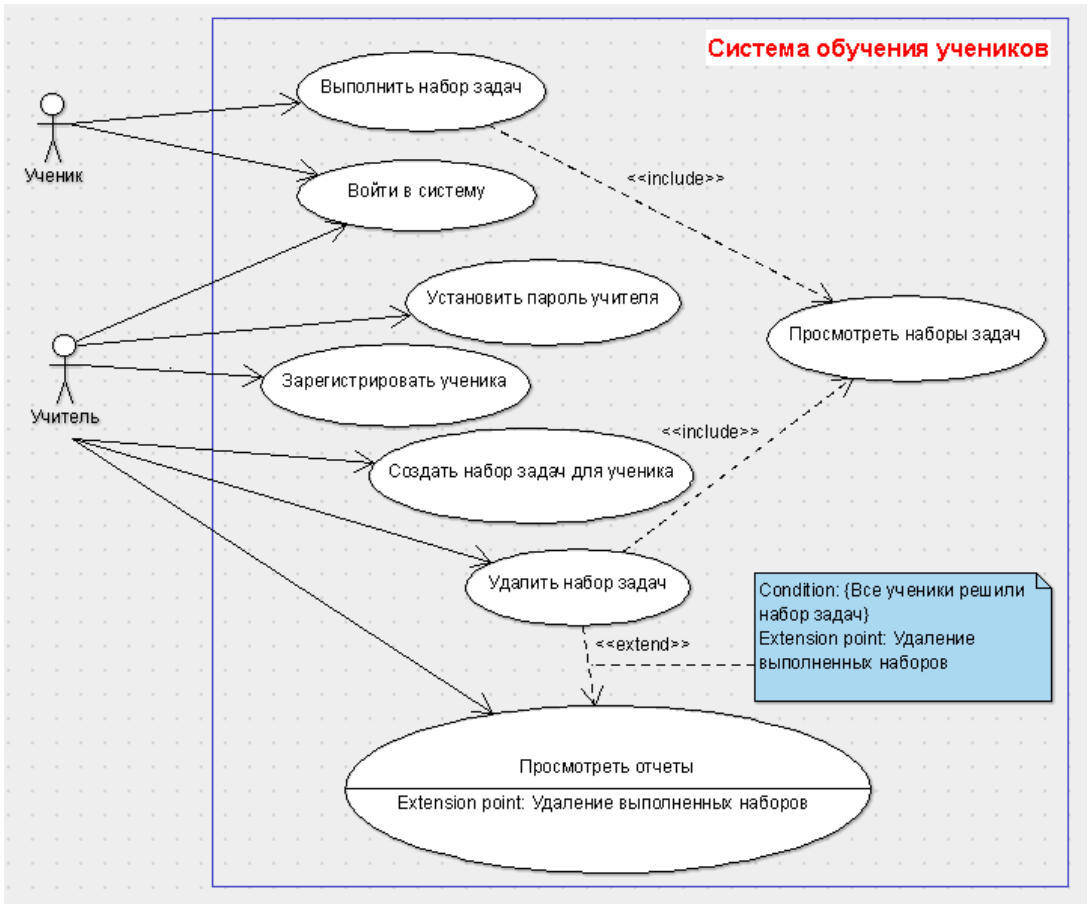
Щоб у варіанті використання показувалася точка розширення, виділіть його та в закладці Стиль необхідно поставити галочку Точки розширення.



Так як умова точки розширення не відображається на діаграмі, можна її додати інформаційно у вигляді примітки. Для додавання примітки виділимо відношення розширення на діаграмі та виберемо піктограму New comment ( ). Суть коментаря додається до поля Body.



В результаті отримуємо діаграму Варіантів використання.

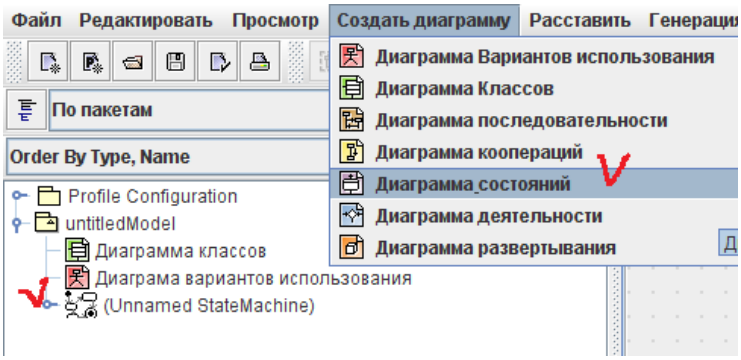


Для моделювання поведінки на логічному рівні у мові UML можуть використовуватися відразу кілька канонічних діаграм: станів, діяльності, послідовності та кооперації, кожна з яких фіксує увагу на окремому аспекті функціонування системи. Діаграма станів описує процес зміни станів лише одного класу, а точніше – одного екземпляра певного класу, тобто моделює всі можливі зміни у стані конкретного об'єкта. У цьому зміна стану об'єкта може бути викликано зовнішніми впливами із боку інших об'єктів чи ззовні. Саме для опису реакції об'єкта на подібні зовнішні дії та використовуються діаграми станів.

Діаграма станів сутнісно є графом спеціального виду, який представляє певний автомат. Вершинами цього графа є стани та інші типи елементів автомата (псевдосостояния), які зображуються відповідними графічними символами. Дуги графа

служать для позначення переходів зі стану до стану. Діаграми станів можуть бути вкладені одна в одну, утворюючи вкладені діаграми більш детального представлення окремих елементів моделі.

Для створення діаграми стану в ArgoUML необхідно в меню Створити діаграму вибрати пункт Діаграма станів.



В результаті з'явиться панель вибору компонентів діаграми стану:

Компонентами діаграми станів є:

– стани та підстани,
– переходи.

Автомат (state machine) – це опис послідовності станів, якими проходить об'єкт протягом життєвого циклу, реагуючи на події, і навіть опис реакцію ці події.

Формалізм звичайного автомата ґрунтується на виконанні наступних обов'язкових умов:

– Автомат не запам'ятовує історію переміщення зі стану до стану.
– У кожний момент часу автомат може перебувати в одному і лише в одному зі своїх станів.
– Тривалість знаходження автомата в тому чи іншому стані, а також час досягнення того чи іншого стану не специфікуються, тобто, час на діаграмі станів присутня у неявному вигляді, хоча для окремих подій може бути вказаний інтервал часу та у явному вигляді.
– Кількість станів автомата повинна бути обов'язково кінцевою (початковий та кінцевий стан).
– Кожен перехід повинен обов'язково з'єднувати два стани автомата. Допускається перехід із стану у собі, такий перехід ще називають "петлею".
– Автомат не повинен містити конфліктуючих переходів, тобто таких переходів з того самого стану, коли об'єкт одночасно може перейти в два і більше наступні стани (крім випадку паралельних підавтоматів).

Формалізм автоматів допускає вкладення одних автоматів до інших. І тут вкладені автомати отримали назву підавтоматів.

Стан (state) – ситуація у життєвому циклі об'єкта, протягом якої він задовольняє певній умові, виконує деяку діяльність чи очікує деяку подію.

Стан на діаграмі зображується прямокутником із заокругленими вершинами  . Цей прямокутник, у свою чергу, може бути поділений на дві секції горизонтальною лінією. Якщо зазначена лише одна секція, то записується лише ім'я стану. Інакше у першій записується ім'я стану, а другий – список деяких внутрішніх дій чи переходів у цьому стані (рисунок 6.8).

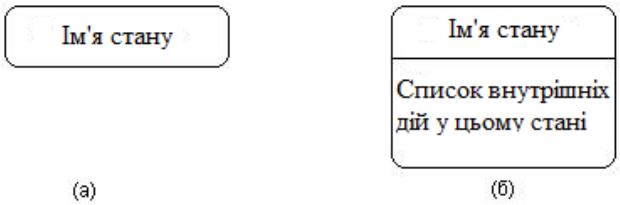


Рисунок 6.8 – Графічне зображення станів на діаграмі станів

Ім'я стану є рядком тексту, який розкриває змістовний зміст цього стану. Ім'я завжди записується з великої літери. Для ідентифікації імені стану рекомендується використовувати дієслова в даний час або відповідні дієприкметники. Ім'я стану може бути відсутнім. У цьому випадку стан є анонімним, і якщо на діаграмі станів їх кілька, всі вони повинні відрізнятися між собою.

Розглянемо перелік внутрішніх дій або діяльностей, які виконуються в процесі знаходження елемента, що моделюється в даному стані. Кожна дія записується у вигляді окремого рядка і має наступний формат:

< мітка-дії '/' вираз-дії >

Мітка дії вказує на обставини або умови, за яких виконуватиметься діяльність, визначена виразом дії. Якщо список виразів дії порожній, то роздільник у вигляді похилої межі '/' може не вказуватися.

Перелік позначок дії має фіксовані значення у мові UML, які не можуть бути використані як імена подій. Ці значення такі:

– entry - ця мітка вказує на дію, специфіковане наступним за нею виразом дії, що виконується в момент входу в цей стан (вхідна дія);

– exit - ця мітка вказує на дію, специфіковане наступним за нею виразом дії, що виконується в момент виходу з цього стану (вихідна дія);

– do – ця мітка специфікує діяльність ("do activity"), яка виконується протягом усього часу, поки об'єкт знаходиться в даному стані, або доти, доки не закінчиться обчислення, специфіковане наступним за нею виразом дії. В останньому випадку після завершення події генерується відповідний результат;

– include – ця мітка використовується для звернення до підавтомату, при цьому наступний за нею вираз дії містить ім'я підавтомату.

У решті випадків мітка дії ідентифікує подію, яка запускає відповідне вираз дії. Ці події називаються внутрішніми переходами і семантично еквівалентні переходам у саме цей стан.

Розглянемо приклад стану. Розглянемо ситуацію введення пароля користувача під час аутентифікації входу в деяку програмну систему. У цьому випадку список внутрішніх дій у даному стані не порожній і включає 4 окремі дії, перші дві з яких стандартні та описані вище, а два останні визначаються своєю специфікацією (рисунок 6.9).

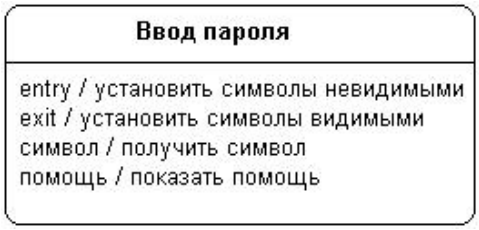
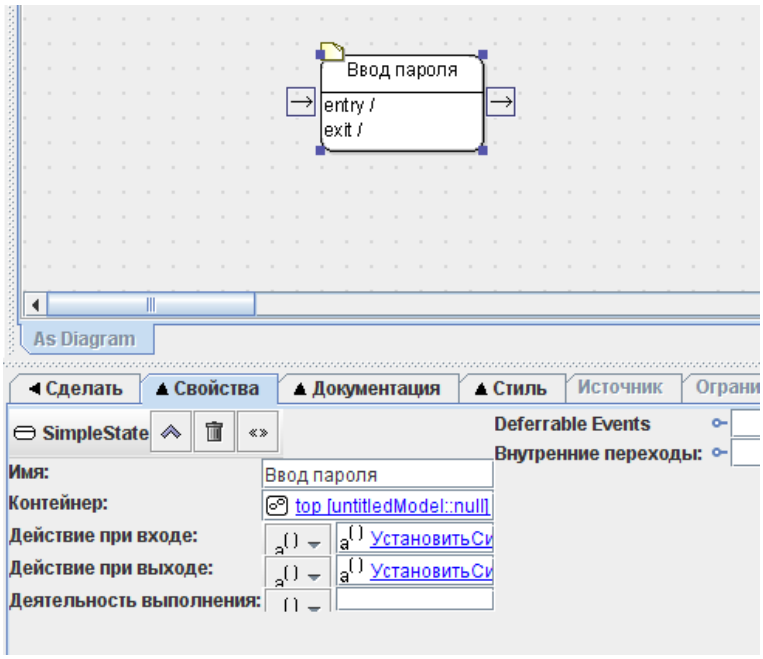


Рисунок 6.9 – Приклад стану із секцією внутрішніх дій

Приклад стану в ArgoUML:



Початковий стан є окремий випадок стану, який не містить жодних внутрішніх дій. У цьому стані знаходиться стандартний об'єкт у початковий момент часу. Графічно початковий стан у мові UML позначається у вигляді зафарбованого кружка, з якого може тільки виходити стрілка, що відповідає переходу.

Кінцевий (фінальний) стан є окремим випадком стану, який також не містить жодних внутрішніх дій. У цьому стані буде об'єкт за замовчуванням після завершення роботи автомата в кінцевий час. Графічно кінцевий стан у мові UML позначається у вигляді зафарбованого кружка, поміщеного в коло, в яке може лише входити стрілка, що відповідає переходу.

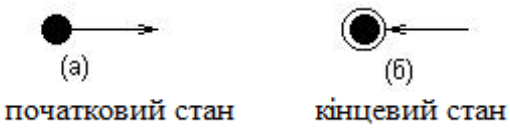
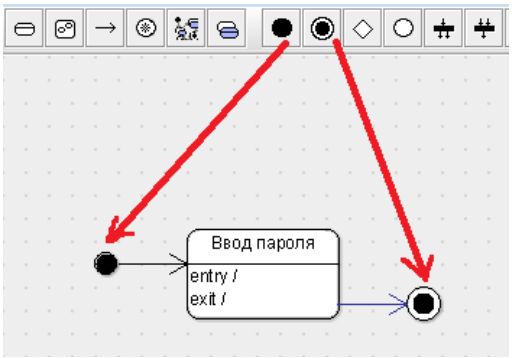


Рисунок 6.10 – Приклад початкового та кінцевого стану

Приклад початкового та кінцевого стану в ArgoUML:



Перехід (transition) – зв'язок між двома станами, що показує, що об'єкт, що у першому стані, має виконати деякі дії і перейти у друге, щойно відбудеться певна подія і будуть виконані певні умови.

Перехід здійснюється при настанні деякої події: закінчення виконання діяльності (do activity), одержання об'єкта повідомлення або прийому сигналу. На переході вказується ім'я події, і можуть вказуватись дії, що проводяться об'єктом у відповідь на зовнішні події при переході з одного стану в інший. Спрацьовування переходу може залежати не тільки від настання певної події, а й від виконання певної умови, яка називається сторожовою умовою.

Об'єкт перейде з одного стану в інший у тому випадку, якщо відбулася зазначена подія та сторожова умова набула значення "істина".

На діаграмі станів перехід зображується суцільною лінією зі стрілкою, яка спрямована на цільовий стан.

Кожен перехід може бути позначений рядком тексту, який має наступний загальний формат:

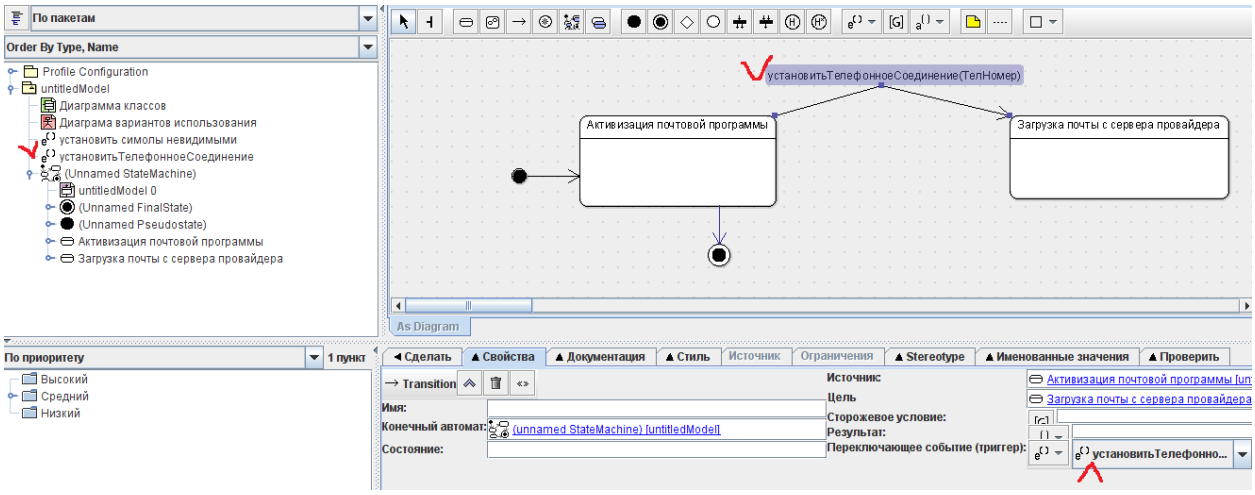
< сигнатура події >'['< сторожова умова >']' < вираз дії >.

При цьому сигнатура події описує певну подію з необхідними аргументами:

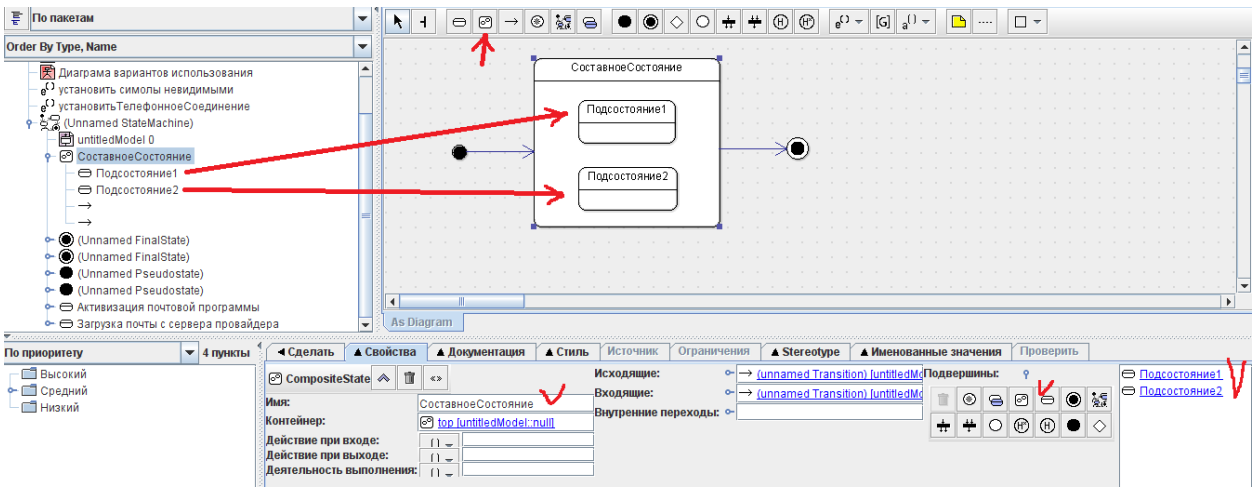
< ім'я події >'('< список параметрів, розділених комами >')'.

Подія (event) – специфікація істотного факту, що відбувається у часі та просторі. У контексті автомата подія – це вплив, що викликає перехід між станами.

Сторожова умова (guard condition), якщо вона є, завжди записується в прямих дужках і є деяким булевським виразом. Якщо сторожова умова набуває значення "істина", то відповідний перехід може спрацювати. Якщо ж сторожова умова набуває значення "брехня", то перехід неспроможне спрацювати, і за відсутності інших переходів об'єкт неспроможне перейти у цільовий стан із цього переходу.

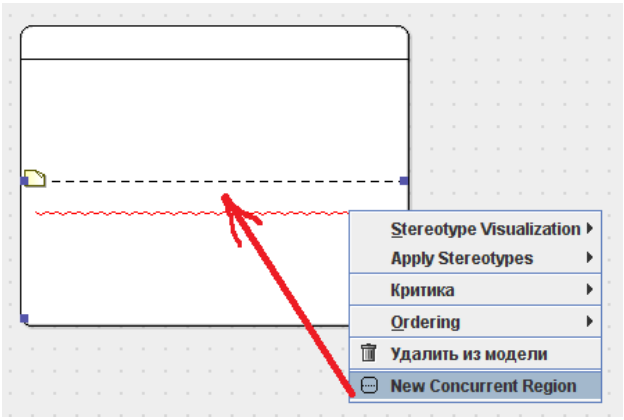


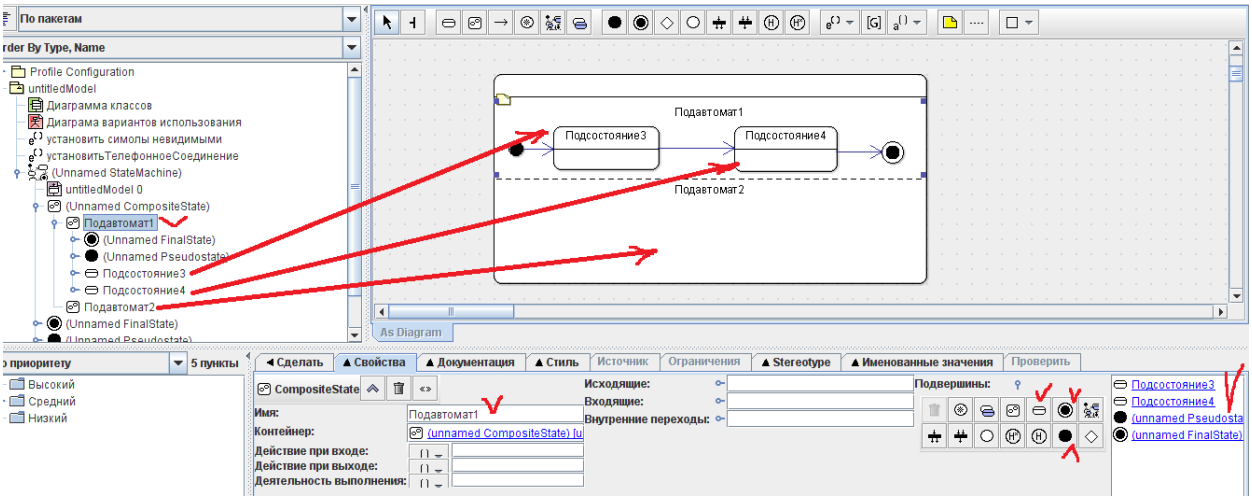
Складовий стан (composite state) – такий складний стан, що складається з інших вкладених у нього станів. Останні виступатимуть по відношенню до першого як субстанції (substate). Хоча між ними має місце композиції, графічно всі вершини діаграми, які відповідають вкладеним станам, зображуються всередині символу складового стану.



Послідовні підстани (sequential substates) використовуються для моделювання такої поведінки об'єкту, під час якого в кожний момент часу об'єкт може перебувати в одному і лише одному стані. Поведінка об'єкту в цьому випадку є послідовною зміною підходів, починаючи від початкового і закінчуючи кінцевим підстаном.

Паралельні стани (concurrent substates) дозволяють специфікувати два і більше підавтомата, які можуть виконуватися паралельно всередині складової події. Кожен із підавтоматів займає деяку область (регіон) усередині складового стану, яка відокремлюється від інших горизонтальною пунктирною лінією. Якщо на діаграмі станів є складовий стан із вкладеними паралельними підходами, то об'єкт може одночасно перебувати в кожному з цих підстанів.





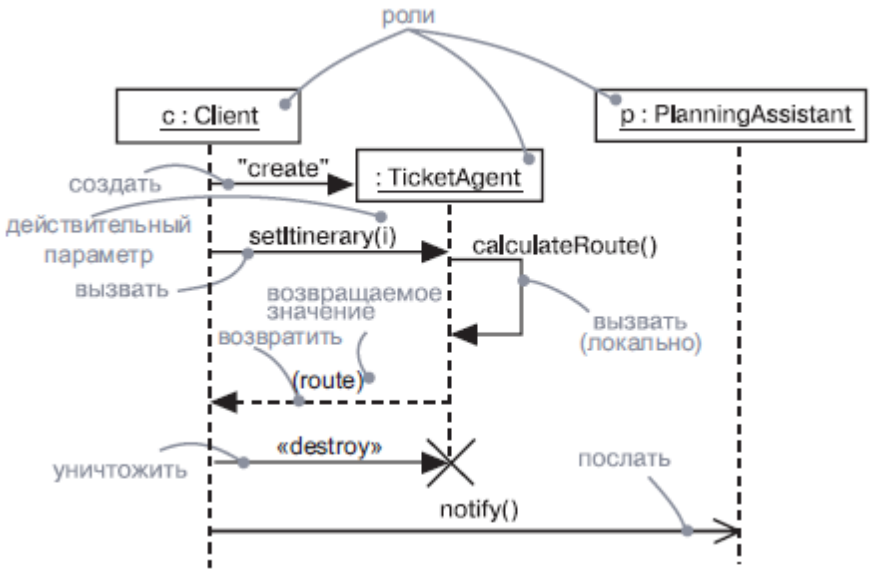
Діяльність (activity) специфікує роботу, що відбувається усередині автомата.

Дія (action) – примітивне обчислення, що призводить до зміни стану моделі або повернення значення.

Діаграми взаємодії, в тому числі діаграми послідовності і комунікації, використовуються в UML для моделювання динамічних аспектів систем.

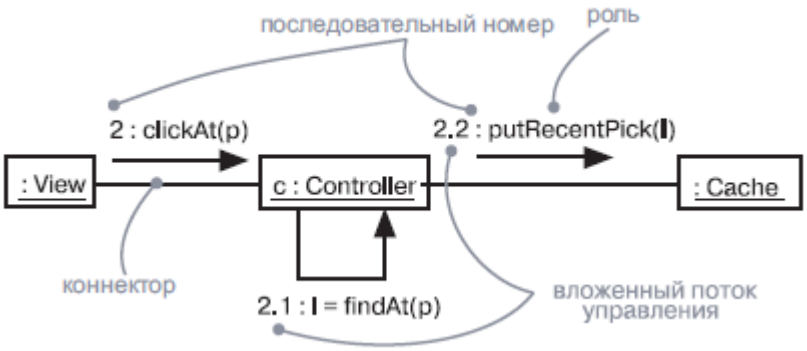
Діаграма взаємодії показує взаємодію ряду об'єктів, а також їх зв'язки та повідомлення, які можуть передаватися між ними.

Наприклад:

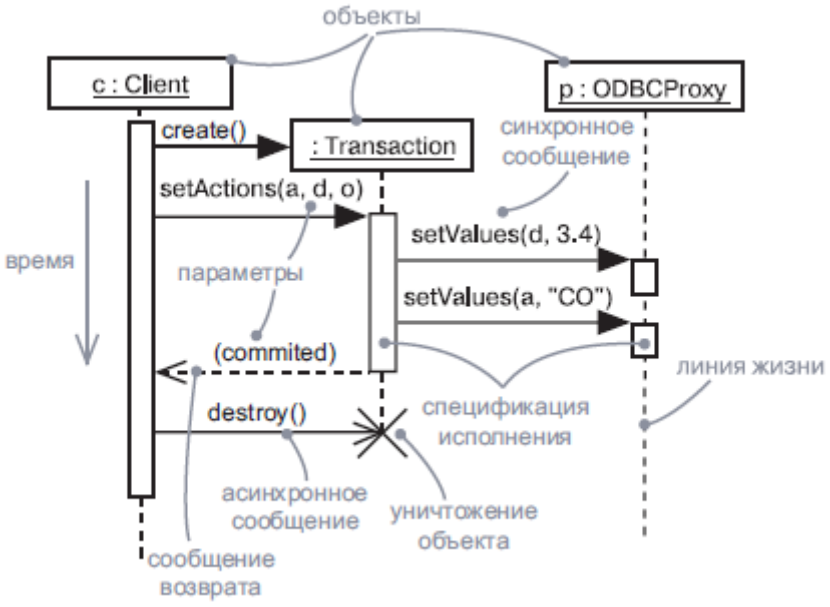


Діаграма послідовності – це діаграма взаємодії, яка підкреслює тимчасової порядок повідомлень. Зображується як таблиця, в якій представлені об'єкти, розташовані уздовж осі X, і повідомлення, впорядковані по ходу часу – уздовж осі Y.

Приклад 1:



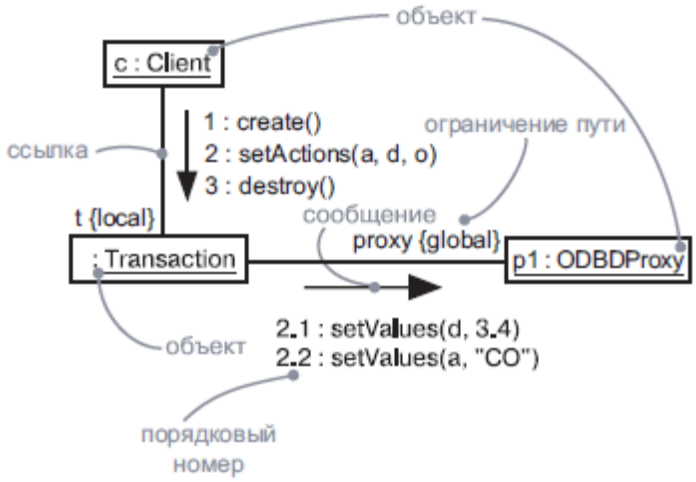
Приклад 2.



Діаграма комунікації – це діаграма взаємодії, яка виділяє структурну організацію об'єктів, які відправляють і приймають повідомлення. Графічно являє собою набір дуг та вершин.

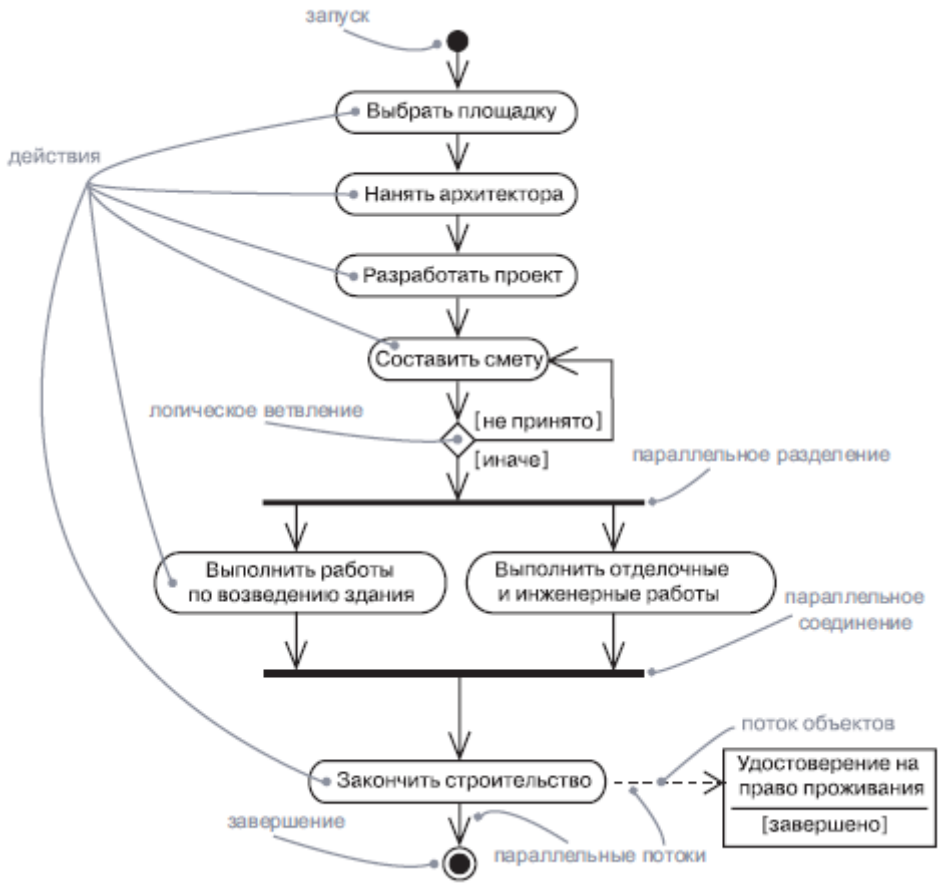
Діаграми комунікації описують організацію об'єктів, що беруть участь у взаємодії. Діаграма комунікації формується починаючи з розміщення об'єктів, що беруть участь у взаємодії, в вершинах графів. Далі у вигляді дуг графа зображуються посилання, які з'єднують ці об'єкти.

Приклад діаграми комунікації:



Діаграми діяльності – це один з п'яти видів діаграм, що застосовуються в UML для моделювання динамічних аспектів систем. По суті, діаграма діяльності являє собою блок-схему, яка показує, як потік управління переходить від однієї діяльності до іншої.

Приклад діаграми діяльності:



7. Завдання для самостійного рішення

Описати наступні процеси використовуючи нотації Flowchart, ERD, UML.

1. Моделювання бізнес-процесів приватного охоронного підприємства. Основні напрямки діяльності підприємства: фізична охорона об'єктів, консультаційні послуги, здійснення робіт із проектування, монтажу технічних засобів охорони. Необхідно спроектувати бізнес-процеси заданої предметної галузі.

- Для виконання монтажних робіт необхідно забезпечити наявність необхідних матеріалів та обладнання на складі підприємства.
- Спеціалізоване обладнання, встановлене у замовника, передається замовнику відповідальне зберігання.
- Залишки матеріалів, які не використані при проведенні монтажних робіт, повинні бути повернені на склад підприємства.
- Для забезпечення рівномірного завантаження працівників у кількох монтажних бригадах координується їхня діяльність.
- Розрахунок заробітної плати монтажних бригад на підприємстві виконується за відрядними розцінками.
- Договір на послуги пультової охорони передбачає періодичну оплату послуг.
- При цьому надання послуги може бути тимчасово припинено на прохання замовника або за рішенням підприємства у разі наявності замовника заборгованості.

2. Моделювання бізнес-процесів кафедри в університеті. Необхідно спроектувати бізнес-процеси заданої предметної галузі.

- Основними завданнями кафедри є проведення обліку контингенту студентів, розробка навчальних планів, розрахунок навантаження, надання місць для проходження практики, випуск студентів.
- Кожній групі призначається куратор, який працює зі студентами, фіксує рухи контингенту, спілкується з батьками.
- Викладачі працюють за складеним розкладом, кожен викладач має годинник консультацій.
- Навантаження розраховується згідно з певними нормативами. Навантаження може бути скориговане та планується на підставі навчальних планів кафедри.