**Лабораторна робота №3 Побудова діаграм послідовностей та діаграм кооперації**

**Мета:** **Навчитися будувати діаграми** **послідовностей**

**Постановка задачі**. Продовжуємо працювати з постановкою задачі з минулих ЛР та ПР **для власного курсового проекту**. **Як альтернативу можна розробити діаграми** по створенню застосунку для викладачів та студентів гіпотетичного навчального закладу**.**

**Завдання до лабораторної роботи**

1. Ознайомитись з теоретичною частиною та рекомендаціями до виконання роботи.
2. На основі визначених в попередніх ЛР та ПР акторів, прецедентів та класів .
3. Запустити Umbrello UML Modeller (в разі відсутності роботу виконувати з іншим доступним засобом UML-моделювання, зокрема з обраним онлайн UML-моделеолм).
4. Відкрити попередній проект (який обрали для реалізації – власний або запропонований викладачем, в ньому створити діаграму послідовностей для кожного раніш створеного прецеденту та створити діаграму кооперації (в Umbrello UML Modeller *communication diagram –* діаграма комунікації, яка в UML 1.x  називалася діаграмою кооперації - collaboration diagram)для кожного раніш створеного прецеденту з урахуванням раніш створеної діаграми послідовностей. В разі потреби – уточнити діаграми прецедентів та класів.
5. Визначити на діаграмі класи, задіяні в процесі реалізації прецеденту .
6. Додати нотатки, де записати своє прізвище та групу.
7. Збережіть проект у власній папці та закрийте проект. Для відсилки викладачу збережіть проект з іменем Вашої ЛР та розширенням xmi.
8. Оформите звіт, в якому надайте скріншоти створеної діаграми та відповіді на запитання в кінці ЛР.
9. Окремо в звіті надайте письмове вирішення завдання для самостійної роботи.
10. Файл зі звітом надати з іменем у форматі :

**ОPI<Номер групи><Номер лекції / практичної / лабораторної>[-<Номер завдання>][літера позначення типу роботи L – лекція, P –практична, R – лабораторна]<Прізвищеанглійською>**. Наприклад, **ОPI3103R**buts.doc.

1. Файл зі звітом та створену Umbrello UML Modeller модель (файл xmi) відішліть на електронну адресу викладача.

**Строк виконання цієї роботи ІПЗ-31 23.10.2024**

**ІПЗ-32 23.10.2024**

**ІПЗ-33 23.10.2024**

**Завдання для самостійної роботи на занятті**

Розгляньте приклад наприкінці ЛР, проаналізуйте діаграми і відтворіть постановку задачі, для вирішення якої створювались діаграми.

**ТЕОРЕТИЧНА ЧАСТИНА**

***Діаграма послідовностей*** відображає взаємодію об’єктів у динаміці. *В UML взаємодія об’єктів розглядається як обмін між ними інформацією у вигляді повідомлень.* Оскільки повідомлення несе певну інформацію, то воно певним чином впливає на отримувача. UML повністю відповідає основним принципам ООП, відповідно до яких взаємодія між об’єктами полягає у відсилці та прийманні повідомлень.

*Діаграма послідовностей відноситься до діаграм взаємодії UML, які описують поведінкові аспекти системи і розглядає взаємодію об’єктів у часі, тобто відображує особливості передачі та приймання повідомлень у часі.*

*Діаграма послідовностей потрібно використовувати для уточнення діаграм прецедентів, надаючи більш детальний опис логіки сценаріїв використання.* Ця обставина робить їх хорошим засобом документування проекту з точки зору сценаріїв використання, оскільки діаграми послідовностей зазвичай містять об’єкти, які взаємодіють у рамках сценарію, повідомлення, якими вони обмінюються та пов’язані з повідомленнями результати, що повертаються. Таким чином,

***Діаграма послідовності (sequence diagram) ‒ це спосіб опису поведінки системи на основі визначення послідовності повідомлень, що передаються.***

Фактично, діаграма послідовності – це запис протоколу конкретного сеансу роботи системи (або фрагменту такого протоколу). Важливим аспектом діаграм послідовності є явне відображення плину часу, оскільки на ній має значення не тільки наявність графічних зв’язків між елементами, але й взаємне розташування елементів на діаграмі відповідно до явно не відображеної вісі часу, яка може бути направлена згори донизу, і пізніше направлене повідомлення розташоване нижче. Вісь часу може бути направлена горизонтально, в цьому випадку вважається, що час плине зліва направо. Таким чином, на діаграмах послідовностей показано обмін повідомленнями (тобто виклик методів) між декількома об’єктами у окремій обмеженій часом ситуації. Об’єкти є екземплярами класів. Основний наголос на діаграмах послідовностей робиться на порядок і моментах часу, у які повідомлення надсилаються об’єктам.

На діаграмах послідовностей об’єкти буде показано вертикальними штриховими лініями з назвою об’єкта над ними. Вісь часу також має вертикальний напрямок, її спрямовано вниз, повідомлення, які надсилаються від одного об’єкта до іншого, буде позначено стрілками з назвами операції і параметрів (див. рис.1).

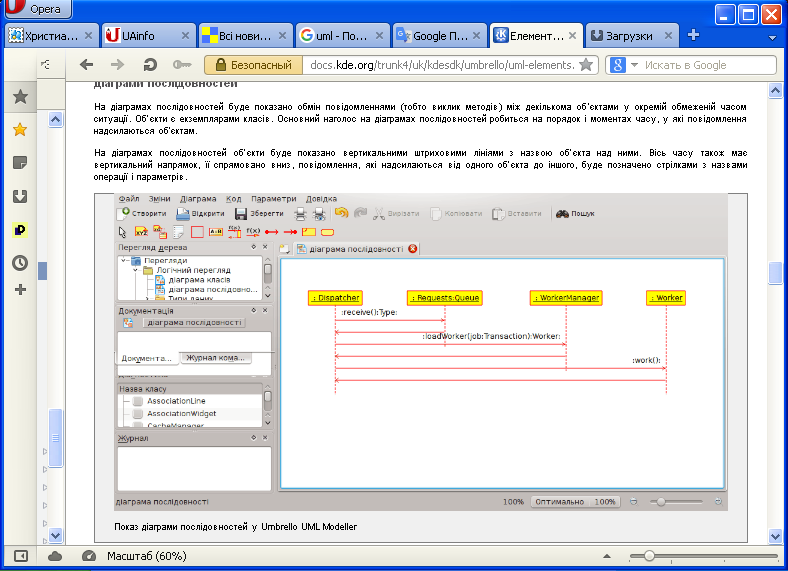


Рис.1 Показ діаграми послідовностей у Umbrello UML Modeller

Для створення діаграми послідовностей у Umbrello UML Modeller потрібно обрати позицію меню Diagram->New-> Sequence diagram… Після чого буде надане вікно (Рис.2), де потрібно ввести ім’я своєї діаграми, натиснути "ОК", після чого можна приступати до побудови діаграм (як на рис.1).



Рис.2 Вікно введення імені діаграми

Повідомлення можуть бути або синхронними, звичайного типу повідомленнями, за виклику яких керування передається викликаному об’єкту до завершення виконання методу, або асинхронними, за виклику яких керування передається назад напряму об’єкту, який здійснював виклик. За використання синхронного повідомлення збоку від викликаного об’єкта буде показано вертикальний блок, який показуватиме перебіг виконання програми.

Оскільки на діаграмі послідовності неявно присутня вісь часу, це дозволяє візуалізувати тимчасові відношення між переданими повідомленнями. За допомогою діаграми послідовності можна представити взаємодію елементів моделі як своєрідний часовий графік "життя" всієї сукупності об'єктів, зв'язаних між собою для реалізації варіанта використання програмної системи, досягнення бізнес-мети або виконання якого-небудь завдання.

На діаграмі послідовності зображуються об'єкти, які безпосередньо беруть участь у взаємодії, при цьому ніякі статичні зв'язки з іншими об'єктами не візуалізуються. Для діаграми послідовності ключовим моментом є саме динаміка взаємодії об'єктів у часі. При цьому діаграма послідовності має як би два виміри. Один – простягається зліва направо у вигляді вертикальних ліній, кожна з яких зображує лінію життя окремого об'єкту, що бере участь у взаємодії. Другий вимір діаграми послідовності - вертикальна тимчасова вісь, спрямована зверху вниз.

Кожен об'єкт графічно зображується у формі прямокутника й розташовується у верхній частині своєї лінії життя (рис. 3). Усередині прямокутника записуються власне ім'я об'єкта (з малої літери) й ім'я класу, розділені двокрапкою. Якщо на діаграмі послідовності відсутнє власне ім'я об'єкта, то повинне бути зазначене ім'я класу, до якого він належить. Такий об'єкт вважається *анонімним*. Ім'я класу також може бути відсутнім, але при цьому повинне бути зазначене власне ім'я об'єкта. Такий об'єкт вважається *сиротою*.



Рис. 3. Графічні елементи діаграми послідовності

Початковому моменту часу відповідає сама верхня частина діаграми. При цьому процес взаємодії об'єктів реалізується за допомогою повідомлень, що одні об’єкти посилають іншим. Повідомлення зображуються у вигляді горизонтальних стрілок, які можуть мати ім'я повідомлення. Повідомлення утворять певний порядок щодо часу своєї ініціалізації. Інакше кажучи, повідомлення, розташовані на діаграмі послідовності вище, передаються раніше тих, які розташовані нижче. При цьому масштаб на осі часу не вказується, оскільки діаграма послідовності моделює лише часову впорядкованість взаємодій типу "пізніше".

***Лінія життя об'єкта (object lifeline)*** - вертикальна лінія на діаграмі послідовності, що представляє існування об'єкта протягом певного періоду часу. Лінія життя об'єкта зображується пунктирною вертикальною лінією, асоційованою з єдиним об'єктом на діаграмі послідовності. Лінія життя служить для позначення періоду часу, протягом якого об'єкт існує в системі і може потенційно брати участь у всіх її взаємодіях. Якщо об'єкт існує в системі постійно, то і його лінії життя триває по всій робочій області діаграми послідовності від самої верхньої її частини до самої нижньої (об'єкт Користувач Петро на рис. 4., можливість такого позначення в Umbrello UML Modeller відсутня).

Окремі об'єкти, закінчивши виконання своїх операцій, можуть бути знищені, щоб звільнити займані ними ресурси. Для таких об'єктів лінія життя обривається в момент їх знищення. Для позначення моменту знищення об'єкта застосовується спеціальний символ у формі латинської букви "X". На рис 4 цей символ використовується для знищення об'єкта «Повідомлення про помилку», утвореного від об'єкту «Клієнтська сесія» . Нижче цього символу пунктирна лінія не зображується, оскільки відповідного об'єкта в системі вже немає, і цей об'єкт повинен бути виключений із всіх наступних взаємодій

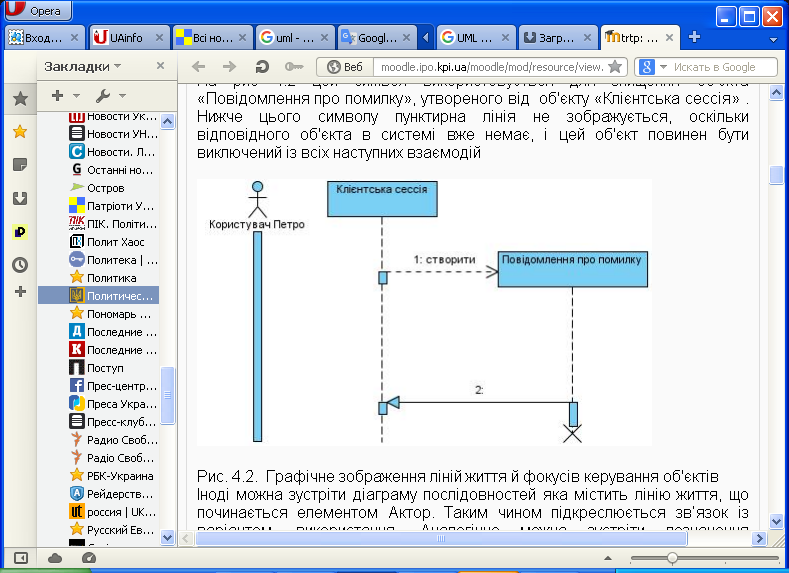


Рис. 4. Графічне зображення ліній життя й фокусів керування об'єктів

Іноді можна зустріти діаграму послідовностей яка містить лінію життя, що починається елементом Актор (в Umbrello UML Modeller відсутня)). Таким чином підкреслюється зв’язок із варіантом використання. Аналогічно можна зустріти позначення граничного, керуючого та класу-сутності.

Не обов'язково створювати всі об'єкти в початковий момент часу, окремі об'єкти в системі можуть створюватися в міру необхідності, що заощаджує ресурси системи й підвищує її продуктивність. У цьому випадку прямокутник такого об'єкта зображується не у верхній частині діаграми послідовності, а в тій, котра відповідає моменту створення об'єкта (об'єкт «Повідомлення про помилку», утворений від об'єкту «Клієнтська сесія» на Рис.4.). Об'єкт створюється зі своєю лінією життя а, можливо, і з фокусом керування.

***Фокус керування (focus of control)*** - спеціальний символ на діаграмі послідовності, що вказує період часу, протягом якого об'єкт виконує деяку дію, перебуваючи в активному стані. У процесі функціонування об’єктно-орієнтованих систем одні об'єкти можуть перебувати в активному стані, безпосередньо виконуючи певні дії, або в стані пасивного очікування повідомлень від інших об'єктів.

Фокус керування зображується у формі витягнутого вузького прямокутника (об'єкт Клієнтська сесія на рис 4.), верхня сторона якого позначає початок одержання фокуса управління об'єкта (початок активності), а її нижня сторона - закінчення фокуса керування (закінчення активності). Цей прямокутник розташовується нижче позначення відповідного об'єкта й може заміняти його лінію життя (об'єкт Користувач Петро на рис. 4), якщо на всьому її протязі він активний.

Періоди активності об'єкта можуть чергуватися з періодами його пасивності або очікування. У цьому випадку в такого об'єкта фокуси керування змінюють своє зображення на лінію життя й навпаки (об'єкт Клієнтська сесія на рис 4). Одержати фокус керування може тільки об'єкт, у якого в цей момент є лінія життя. Якщо ж об'єкт був знищений, то знову виникнути в системі він уже не може. Замість нього може бути створений лише екземпляр цього ж класу, який буде іншим об'єктом.

В окремих випадках ініціатором взаємодії в системі може бути актор або зовнішній користувач. При цьому актор зображується на діаграмі послідовності першим об'єктом ліворуч зі своїм фокусом керування (рис. 4). Найчастіше актор і його фокус керування будуть існувати в системі постійно, відзначаючи характерну для користувача активність в ініціюванні взаємодій із системою. Актор може мати власне ім'я або залишатися анонімним.

В окремих випадках об'єкт може посилати повідомлення самому собі, ініціюючи так називані рефлексивні повідомлення. Для цієї мети призначене спеціальне зображення (повідомлення об'єкта на рис 5). Такі повідомлення зображуються у формі повідомлення, початок і кінець якого стикаються з лінією життя або фокусом керування того самого об'єкта. Подібні ситуації виникають, наприклад, при обробці натискань на клавіші клавіатури при уведенні тексту в документ, що редагується, при наборі цифр номеру телефону абонента.

Якщо в результаті рефлексивного повідомлення створюється новий підпроцес або гілка керування, то говорять про *рекурсивний* або *вкладений фокус керування*. На діаграмі послідовності рекурсія позначається невеликим прямокутником, приєднаним до правої сторони фокуса керування того об'єкта, для якого зображується дана рекурсивна взаємодія (об'єкт б на рис 5).

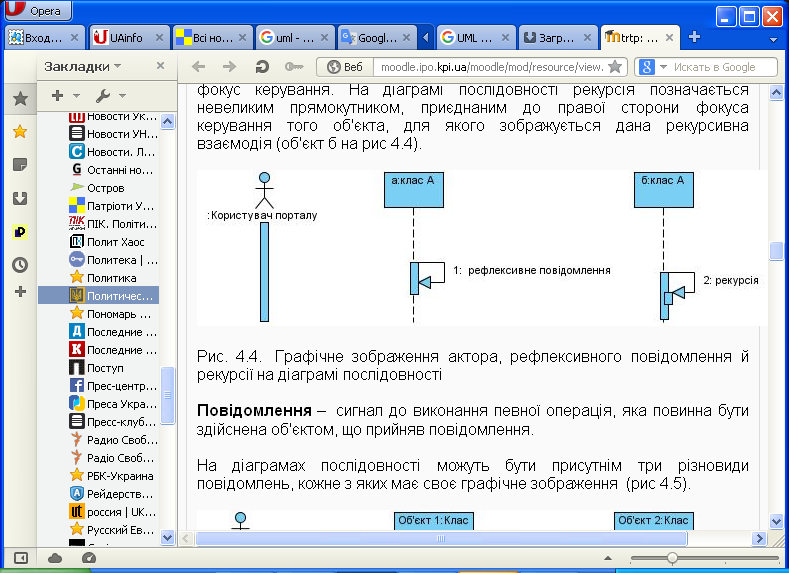


Рис. 5. Графічне зображення актора, рефлексивного повідомлення й рекурсії на діаграмі послідовності

***Повідомлення*** – сигнал до виконання певної операція, яка повинна бути здійснена об'єктом, що прийняв повідомлення.

На діаграмах послідовності можуть бути присутнім три різновиди повідомлень, кожне з яких має своє графічне зображення (рис 6).



Рис. 6. Графічне зображення різних видів повідомлень між об'єктами на діаграмі послідовності

Перший різновид повідомлення (рис 6, 2:виклик процедури) найпоширеніший. Він використовується для виклику процедур, виконання операцій або позначення окремих вкладених потоків керування. Початок цієї стрілки, як правило, стикається з фокусом керування того об'єкта-клієнта, що ініціює це повідомлення. Кінець стрілки стикається з лінією життя того об'єкта, що приймає це повідомлення й виконує у відповідь певні дії. При цьому приймаючий об'єкт може одержати фокус керування, стаючи в цьому випадку активним. Передаючий об'єкт може втратити фокус керування або залишитися активним.

Другий різновид повідомлення (рис 6, 1:асинхронний виклик) використовується для позначення простого асинхронного повідомлення, що передається в довільний момент часу.

Третій різновид повідомлення (рис 6, 3:повернення з процедури) використовується для повернення з виклику процедури. Прикладом може служити просте повідомлення про завершення обчислень без надання результату розрахунків об'єкту-клієнтові. У процедурних потоках керування ця стрілка може бути опущена, оскільки її наявність неявно передбачається наприкінці активізації об'єкта. У той же час вважається, що кожен виклик процедури має свою пару - повернення виклику. Для непроцедурних потоків керування, включаючи паралельні й асинхронні повідомлення, стрілка повернення повинна вказуватися явно.

**Стереотипи повідомлень**

У мові UML передбачені деякі стандартні дії, що виконуються у відповідь на отримання відповідного повідомлення. Вони можуть бути явно вказані на діаграмі послідовності у формі стереотипу поряд з повідомленням, до якого вони відносяться. В цьому випадку вони записуються в лапках. Використовуються наступні позначення для моделювання дій:

* "call" (викликати) - повідомлення, що вимагає виклику операції або процедури приймаючого об'єкту. Якщо повідомлення з цим стереотипом рефлексивне, то воно ініціює локальний виклик операції у того самого об’єкта, що надіслав повідомлення;
* "return" (повернути) - повідомлення, що повертає значення виконаної операції або процедури об'єкту, що викликав її. Значення результату може ініціювати розгалуження потоку управління;
* "create" (створити) - повідомлення, що вимагає створення іншого об'єкту для виконання певних дій. Створений об'єкт може отримати фокус управління, а може і не отримати його;
* "destroy" (знищити) - повідомлення з явною вимогою знищити відповідний об'єкт. Посилається у тому випадку, коли необхідно припинити небажані дії з боку об'єкту, що існує в системі, або коли об'єкт більше не потрібний і повинен звільнити задіяні їм системні ресурси;
* "send" (послати) - позначає посилку іншому об'єкту деякого сигналу, який асинхронно ініціюється одним об'єктом і приймається (перехоплюється) іншим. Відмінність сигналу від повідомлення полягає в тому, що сигнал повинен бути явно описаний в тому класі, об'єкт якого ініціює його передачу.

Нижче представлена діаграма послідовності із використанням стереотипних значень (рис. 7). На діаграмі проілюстровано скорочену варіацію на тему казки про «Ріпку». Зауважимо, що на початку ріпки в системі не існувало, даний об’єкт був посаджений (створений) дідом, що відображається стереотипом «create». Після невдалою спробі витягти ріпку, дід звернувся до бабці (стереотип «send»), яка в свою чергу покликала мишу(стереотип «send»). Знищення ріпки відображено стереотипом «destroy».

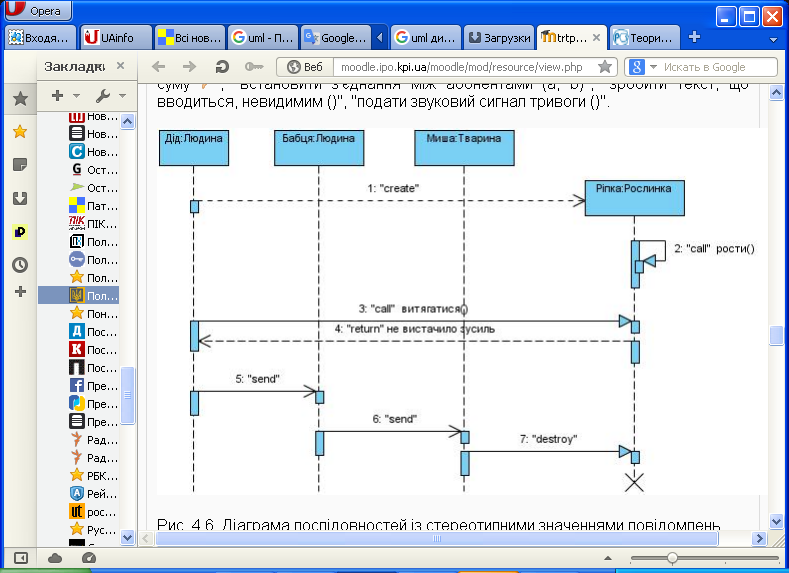


Рис. 7. Діаграма послідовностей із стереотипними значеннями повідомлень

В Umbrello UML Modeller можливість визначення стереотипів відсутня, повідомлення можуть мати власне позначення операції, виклик якої вони ініціюють у приймаючого об'єкту. В цьому випадку поряд із стрілкою записується ім'я операції з круглими дужками, в яких можуть указуватися параметри або аргументи відповідної операції. Якщо параметри відсутні, то дужки все одно повинні бути присутніми після імені операції (Рис.9).

**Загублені та знайдені повідомлення**

***Загублені повідомлення*** (Lost messages) – це повідомлення, які надсилаються, проте не надходять до зазначеного отримувача або отримувач не наявний на даній діаграмі. ***Знайдені повідомлення*** (Found messages) – це повідомлення, що надходять від невідомого відправника або відправник не наявний на даній діаграмі. На наступній діаграмі представлено їх графічне зображення.

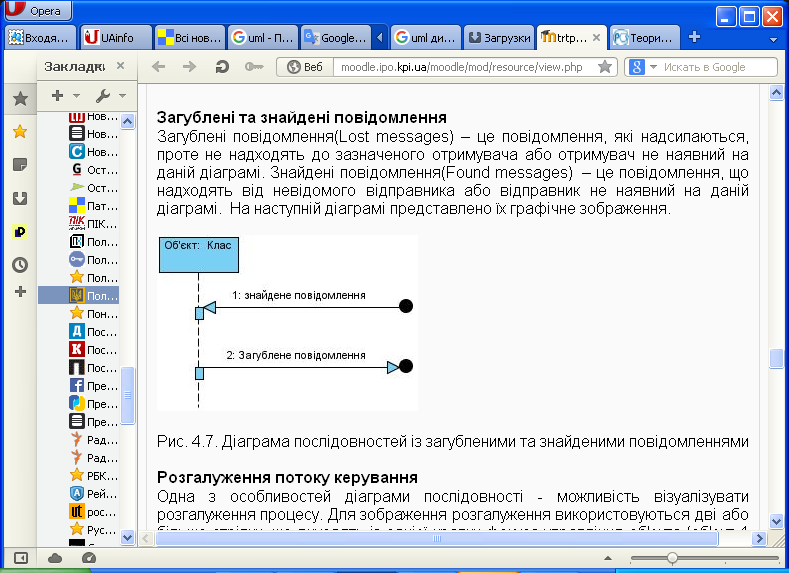


Рис. 9. Діаграма послідовностей із загубленими та знайденими повідомленнями

**Розгалуження потоку керування**

Одна з особливостей діаграми послідовності - можливість візуалізувати розгалуження процесу. Для зображення розгалуження використовуються дві або більше стрілки, що виходять із однієї крапки фокуса управління об'єкта (об'єкт 1 на рис. 8). Поруч із кожною з них у формі булевського виразу повинна бути явно зазначена відповідна умова гілки.

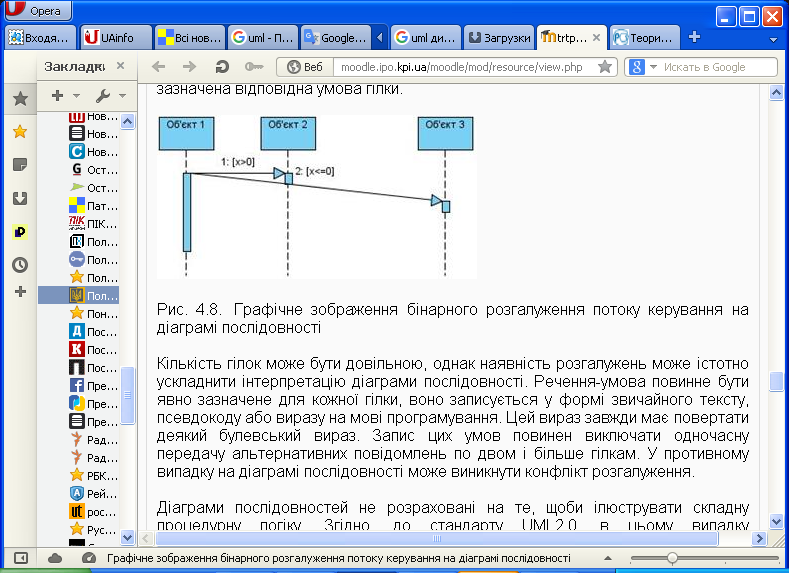


Рис. 8. Графічне зображення бінарного розгалуження потоку керування на діаграмі послідовності

Кількість гілок може бути довільною, однак наявність розгалужень може істотно ускладнити інтерпретацію діаграми послідовності. Речення-умова повинне бути явно зазначене для кожної гілки, воно записується у формі звичайного тексту, псевдокоду або виразу на мові програмування. Цей вираз завжди має повертати деякий булевський вираз. Запис цих умов повинен виключати одночасну передачу альтернативних повідомлень по двом і більше гілкам. У противному випадку на діаграмі послідовності може виникнути конфлікт розгалуження. Діаграми послідовностей не розраховані на те, щоби ілюструвати складну процедурну логіку.

**Діаграми взаємодії (interaction diagrams)**

Діаграми взаємодії є моделями, що описують поводження взаємодіючих груп об'єктів. Як правило, діаграма взаємодії охоплює поводження тільки одного варіанта використання. На такій діаграмі відображається ряд об'єктів і ті повідомлення, якими вони обмінюються між собою в рамках даного варіанта використання.

Цей підхід буде проілюстрований на прикладі простого варіанта використання, що описує таку поведінку:

• «Менеджер» проглядає поточний «Звіт»;

• якщо «Звіт» застарів, «Менеджер» надсилає запит «Виконавцеві» на оновлення «Звіту»;

• «Виконавець» створює новий «Звіт»;

• «Звіт» направляється «Менеджеру» ;

• «Менеджер» виконує перегляд нового «Звіту».

Існує два види діаграм взаємодії: діаграми послідовності і діаграми кооперації/співпраці або, як їх ще називають, кооперативні діаграми.

**Діаграми послідовності (sequence diagrams)**

Діаграми послідовності ми вже розглядали на ЛР№3. Пригадаємо та розглянемо її стосовно нашого прикладу. На діаграмі послідовності об'єкт зображується у вигляді прямокутника на вершині пунктирної вертикальної лінії (рис. 1). Ця вертикальна лінія називається лінією життя об'єкта (lifeline). Вона являє собою фрагмент життєвого циклу об'єкта в процесі взаємодії.

Кожне повідомлення представляється у вигляді стрілки між лініями життя двох об'єктів. Повідомлення з'являються в тому порядку, як вони показані на діаграмі (зверху вниз). Кожне повідомлення може бути позначено ім'ям, за бажанням можна вказати також аргументи й деяку керуючу інформацію. Також можна використовувати самоделегування - повідомлення, яке об'єкт посилає самому собі, при цьому стрілка повідомлення вказує на ту ж саму лінію життя.



Рис. 1. Діаграма послідовності

Із всієї можливої керуючої інформації два її види мають істотне значення. По-перше, це умова, що показує в якому випадку посилається повідомлення. Наприклад, можна ввести умову: [Звіт\_застарів() = true]. Тоді запит на оновлення звіту буде посилатися тільки при виконанні цієї умови. По-друге, корисним керуючим маркером є маркер ітерації, який показує, що повідомлення посилається багато разів для множини об'єктів-адресатів (наприклад, \*оновити).

Активізації - прямокутники на лініях життя - показують, коли метод стає активним (під час його виконання або при очікуванні результату виконання якої-небудь процедури). Використовуючи механізм активізацій, можна більш чітко показати зміст самоделегування. Без цього досить важко визначити, де саме виконуються наступні після самоделегування звернення - у методі, який викликає, чи у методі, який викликається.

**Діаграми кооперації (collaboration diagrams)**

Другим видом діаграми взаємодії є діаграма кооперації, яка з версії UML 2.x називається діаграмою комунікації *(communication diagram)*. Далі будемо використовувати стару назву. На діаграмах кооперації показується взаємодія між об’єктами, які беруть участь у певній події. Ця інформація більшою чи меншою мірою подібна до інформації, показаної на діаграмі послідовностей, але там наголос робиться на часовій характеристиці взаємодії, а на діаграмах кооперації основний наголос робиться на взаємодії між об’єктами та її топології на передньому плані.

Головна особливість діаграми кооперації полягає у можливості графічно представити не тільки послідовність взаємодії, але й всі структурні відношення між об’єктами, що приймають участь в цій взаємодії. Поведінка системи може описуватися на рівні окремих об'єктів, які обмінюються між собою повідомленнями, щоб досягти потрібної мети або реалізувати певний сервіс. Таким чином, за допомогою діаграми кооперації можна описати повний контекст взаємодій як своєрідний часовий "зріз" сукупності об'єктів, які взаємодіють між собою для виконання певного завдання або цілі програмної системи.

На діаграмах кооперації повідомлення надіслані від одного з об’єктів до іншого позначаються стрілочками, поряд з якими показано назву повідомлення, параметри і послідовність повідомлення. Діаграми кооперації найкраще підходять для показу специфічного перебігу виконання або ситуацій у програмі. Такі діаграми є найкращим засобом для швидкого показу і пояснення окремого процесу у програмній логіці.

Діаграма кооперації щодо розглянутого вище прикладу представлена на рис. 2.



Рис. 2. Діаграма кооперації

На діаграмі кооперації екземпляри об'єктів показані у вигляді піктограм. Лінії між ними позначають повідомлення, обмін якими здійснюється в межах даного варіанта використання.

Діаграма кооперації є іншим варіантом діаграми послідовностей.

*Переваги діаграми кооперацій*:

1) Діаграма кооперацій більш компактна, відображає взаємодію, прив'язуючись до архітектури системи.

2) На ній зручніше розташовувати компоненти і відстежувати потоки подій.

3) Краще видно зв'язок з БД

На діаграмі кооперацій вісь часу відсутня, тому нумерація повідомлень обов'язкове.

На рис 3. показано діаграма кооперації у Umbrello UML Modeller.

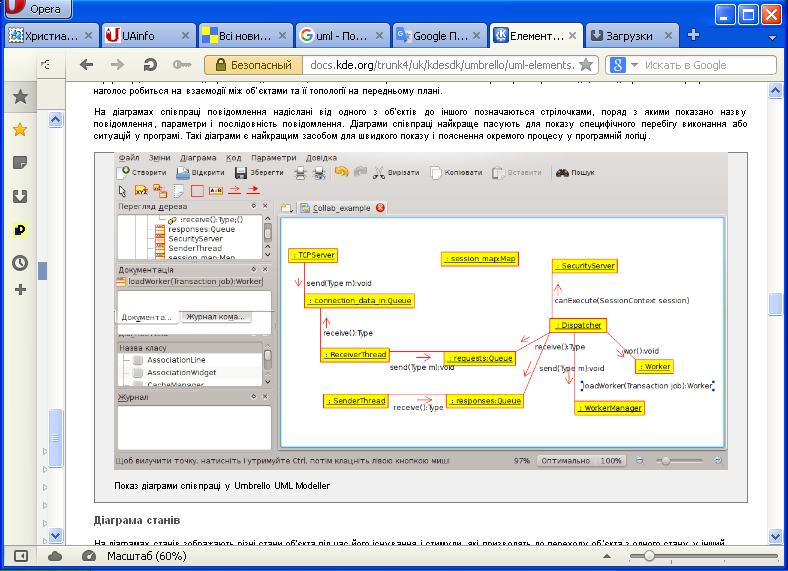


Рис. 3. Показ діаграми кооперації у Umbrello UML Modeller

Розглянемо більш детально діаграми кооперації в загальному випадку (без прив’язки до конкретного інструменту UML-моделювання).

Головна особливість діаграм кооперації – можливість графічно надати не тільки послідовність взаємодії, але й всі структурні відносини між об’єктами, що в ній приймають участь.

Прямокутником на діаграмі кооперації позначаються об’єкти, що приймають участь у взаємодії. В прямокутнику позначається ім’я об’єкту, його клас і, можливо, значення атрибутів. Як і в діаграмі класів асоціації між об’єктами вказуються у вигляді різних з’єднуючих ліній, до яких можна вказати імена асоціацій та ролей, які грають об’єкти в цій асоціації. Додатково можуть бути зображені динамічні зв’язки – потоки повідомлень. Вони надаються у вигляді з’єднувальних ліній між об’єктами, над якими знаходяться стрілка, що вказує напрямок, ім’я повідомлення та порядковий номер в загальній послідовності ініціалізації повідомлень.

На відміну від діаграми послідовностей на діаграмі кооперації не вказується час у вигляді окремого виміру, тому послідовність взаємодії та паралельні потоки можуть бути визначеними за допомогою порядкових номерів. Тому, якщо необхідно розроблювати специфікацію взаємодії між об’єктами в реальному часі, краще використовувати діаграму послідовностей.

**Кооперація** *(collaboration)***.**

Це поняття є одним з фундаментальних у мові UML. Воно слугує для позначення множини об’єктів, що взаємодіють з певною метою, в загальному контексті моделюємої системи. Мета кооперації – специфікувати особливості окремих найбільш значущих операцій у системі. Кооперація визначає структуру поведінки системи у термінах взаємодії учасників цієї кооперації.

Кооперація може бути представлена на двох рівнях:

* рівні специфікації (Specification-Level) - показує ролі класифікаторів и ролі асоціацій у розглядаємій взаємодії;
* рівні прикладів - вказує екземпляри (instance) та зв’язки, які створюють окремі ролі в кооперації.

Діаграма кооперації рівня специфікації показує ролі, які виконують елементи – учасники взаємодії. Елементами кооперації на цьому рівні є класи та асоціації, які позначають окремі ролі класифікаторів (які відповідають класам на діаграмі класів) та асоціації між учасниками кооперації.

Кооперація рівня специфікації зображується на діаграмі пунктирним еліпсом, всередині якого записується ім’я цієї кооперації. Таке представлення кооперації відноситься до окремого варіанту використання і деталізує особливості його подальшої реалізації. Символ еліпса кооперації з’єднується відрізками пунктирної лінії з кожним з учасників цієї кооперації, в якості яких можуть виступати об’єкти або класи. Кожна з цих пунктирних ліній помічується роллю (*role*) учасника. Ролі відповідають іменам елементів в контексті всієї кооперації. Ці імена трактуються як параметри, які обмежують специфікацію елементів при будь-якій їх появі в окремих представленнях моделі.

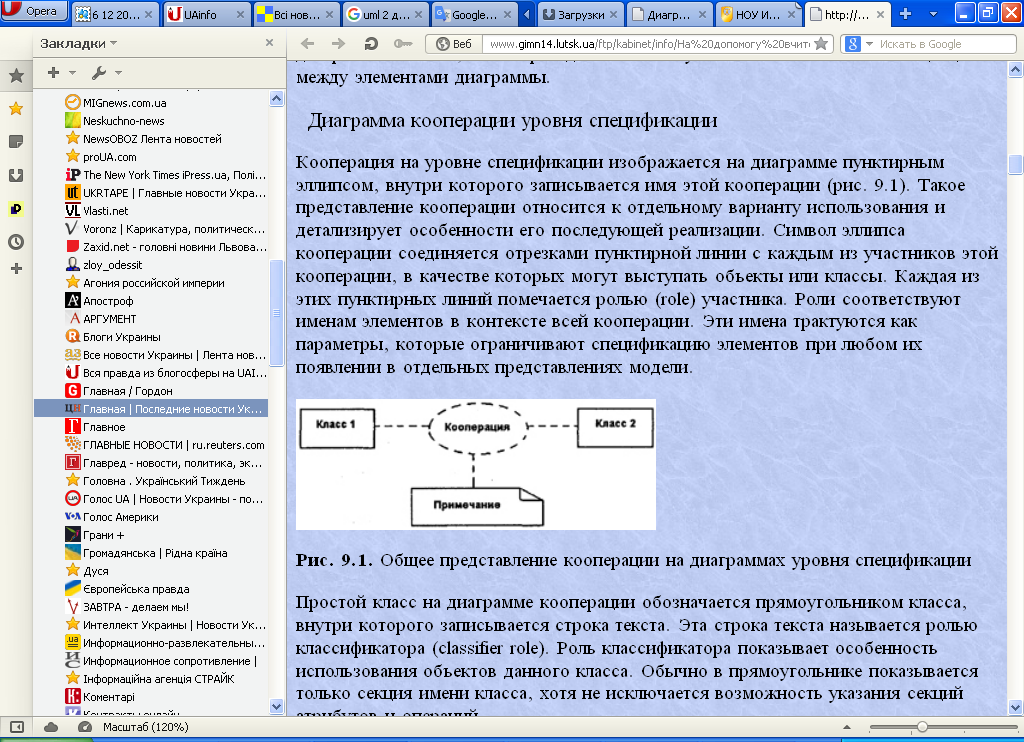
******

Рис. 4. Загальне представлення кооперації на діаграмах рівня специфікації

Простий клас на діаграмі кооперації позначається прямокутником класу, всередині якого записується рядок тексту, який називається роллю класифікатору (*classifier role*). Роль класифікатора показує особливість використання об’єктів даного класу. зазвичай в прямокутнику показується лише секція імені класу.

Для визначення окремих кооперацій, які є спеціалізацією іншої кооперації, використовується звичайна стрілка узагальнення, направлена від символу дочірньої кооперації до символу кооперації-предка. Ролі дочірніх кооперацій можуть бути спеціалізаціями ролей кооперацій-предків.

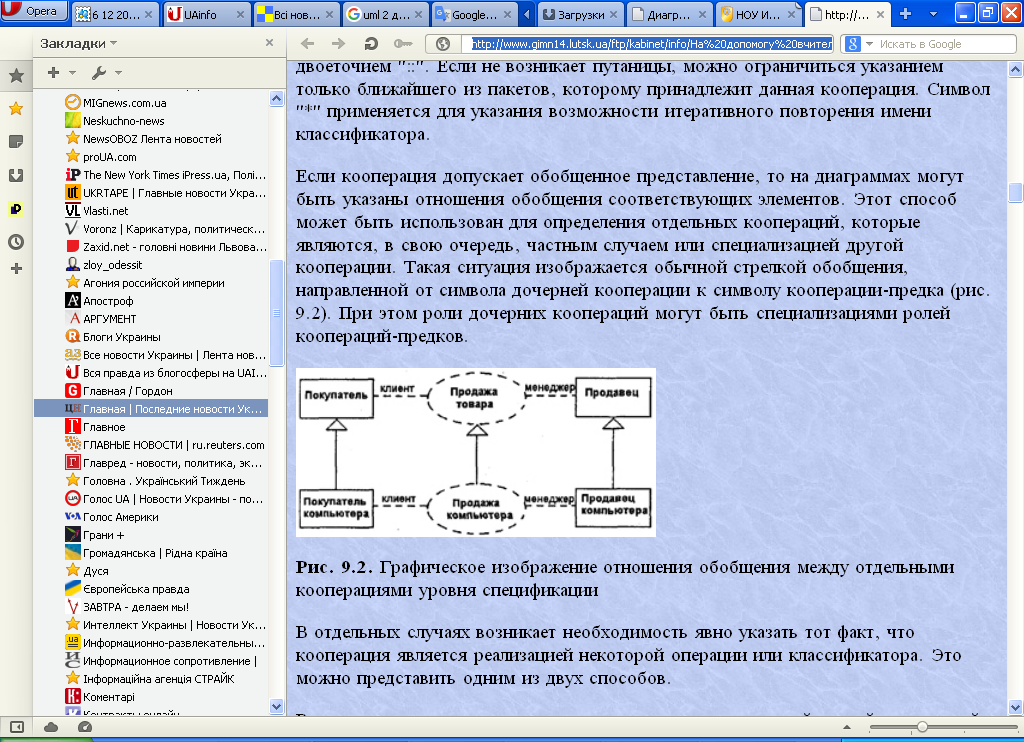


Рис. 5. Графічне зображення відношення узагальнення між окремими коопераціями рівня специфікації

Діаграма кооперації рівня прикладів є сукупністю об’єктів (екземпляри класів) та зв’язків (екземпляри асоціацій). При цьому зв’язки доповнюються стрілками повідомлень. На цьому рівні показуються тільки об’єкти, що мають безпосереднє відношення до реалізації операції або класифікатора. Тут зовсім не обов’язково зображувати усі властивості або усі асоціації, оскільки на діаграмі кооперації присутні тільки роли класифікаторів, а не самі класифікатори. Таким чином, якщо класифікатор потребує повного опису всіх своїх екземплярів, роль класифікатора потребує лише опису тих властивостей та асоціацій, які необхідні для участі в окремій кооперації.

Таким чином, ***одна і та ж сукупність об’єктів може брати участь в різних коопераціях.*** В залежності від розглядаємої кооперації можуть змінюватися як властивості окремих об’єктів, так і зв’язки між ними. Це є відмінністю діаграми кооперації від діаграми класів, для якої потрібно вказувати усі все властивості та асоціації між елементами діаграми.

Об’єкти є основними елементами з яких будується діаграма кооперації рівня прикладів. Оскільки об’єкт (*object*) є окремим екземпляром класу, який створюється на етапі виконання програми, він може мати своє власне ім’я та конкретні значення атрибутів. В загальному випадку ім’я задається так

*<Ім’я об’єкту>'/' <Ім’я ролі класифікатору> ':' <Ім’я класифікатору >*

*[':' < Ім’я класифікатору >]*

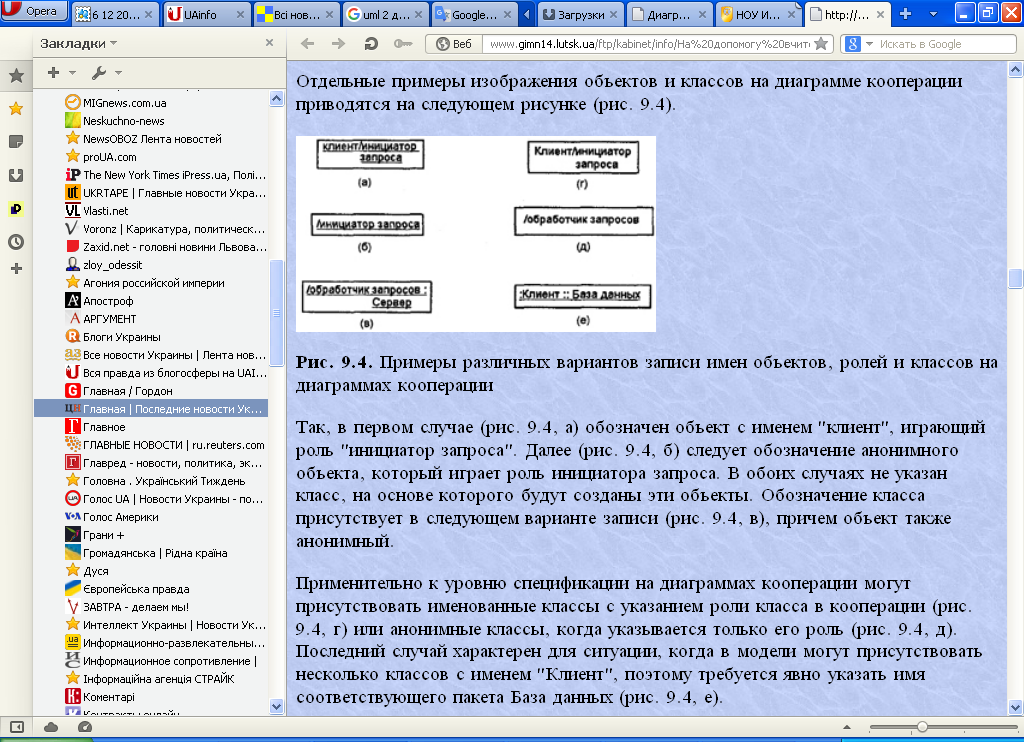


Рис. 6. Приклади різних варіантів запису імен об’єктів, ролей та класів на діаграмах кооперації

У першому випадку (рис.6, а) позначений об’єкт с іменем "клієнт", що грає роль "ініціатор запиту". Далі (рис.6, б) надано позначення анонімного об’єкту, який грає роль ініціатору запиту. В обох випадках не вказаний клас, на основі якого будуть створені ці об’єкти. Позначення класу присутнє в наступному варіанті запису (рис.6,ºв), причому об’єкт також анонімний. На діаграмах кооперації можуть бути присутніми іменовані класи з вказівкою ролі класу в кооперації (рис., г) або анонімні класи, коли вказується тільки його роль (рис.6, д). Останнє характерне для ситуації, коли в моделі можуть бути декілька класів з іменем "Клієнт", тому потрібно явно вказати ім’я відповідного пакета База даних (рис.6, е).

***Мультіоб’єкт*** *(multiobject)* є множиною об’єктів на одному з кінців асоціації. На діаграмі кооперації мультіоб’єкт використовується для того, щоб показати операції та сигнали, які адресовані всій множені об’єктів, а не тільки одному. Мультіоб’єкт зображається двома прямокутниками, один з яких виступає з під правої верхньої вершини іншого (рис.7, а). Стрілка повідомлення відноситься до всієї множини об’єктів, які позначають даний мультіоб’єкт. На діаграмі кооперації може бути явно вказано відношення композиції між мультіоб’єктом та окремим об’єктом з його множини (рис.7, б).

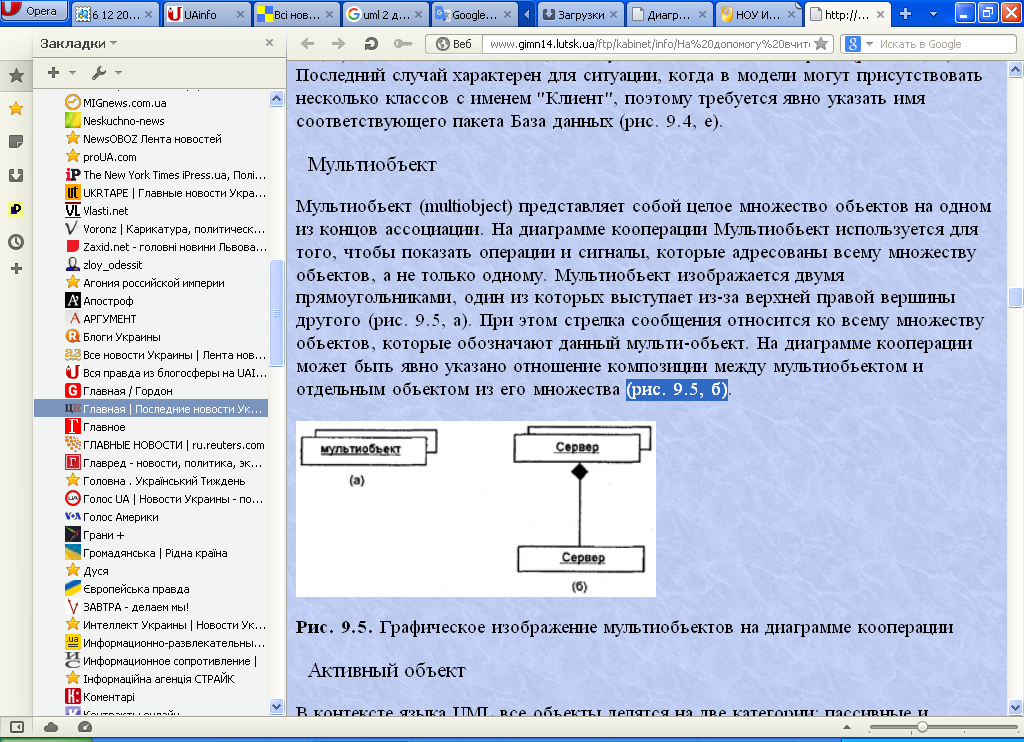


Рис. 7. Графічне зображення мультіоб’єктів на діаграмі кооперації

В прикладі на рис.8 показано ситуацію з викликом функції друку з текстового редактора. Анонімний активний об’єкт "Текстовый редактор" спочатку посилає повідомлення анонімному мультіоб’єкту "Принтер", яке ініціює вибір єдиного об’єкту "Принтер", можливо, задовольняючого деяким додатковим умовам. Після цього обраному об’єкту посилається повідомлення про необхідність друку документу з текстового редактору. ***Активний об’єкт*** *(active object)* має свій деякий полегшений потік (*thread*) керування - нитку і може ініціювати діяльність по керуванню іншими об’єктами. Потік керування може виконуватися паралельно с іншими обчислювальними нитями або нитями керування в межах одного обчислювального процесу або процесу керування. Активні об’єкти на канонічних діаграмах позначаються прямокутником з більш широкими границями, інколи може бути явно вказано ключове слово *{active}* або префікс "а:".

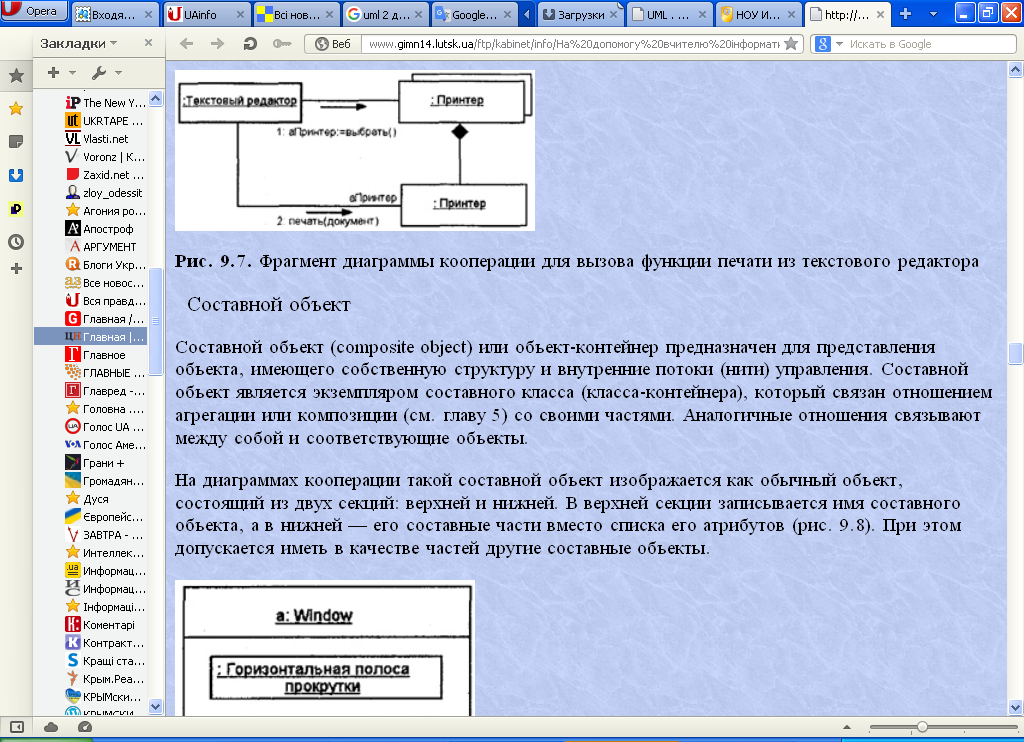


Рис. 8. Фрагмент діаграми кооперації для виклику функції друку з текстового редактора

***Складений об’єкт*** *(composite object)* або об’єкт-контейнер призначений для представлення об’єкта з власною структурою і внутрішніми потоками (нитями) керування. Складений об’єкт є екземпляром класу-контейнера, зв’язаного відношенням агрегації або композиції зі своїми частинами.

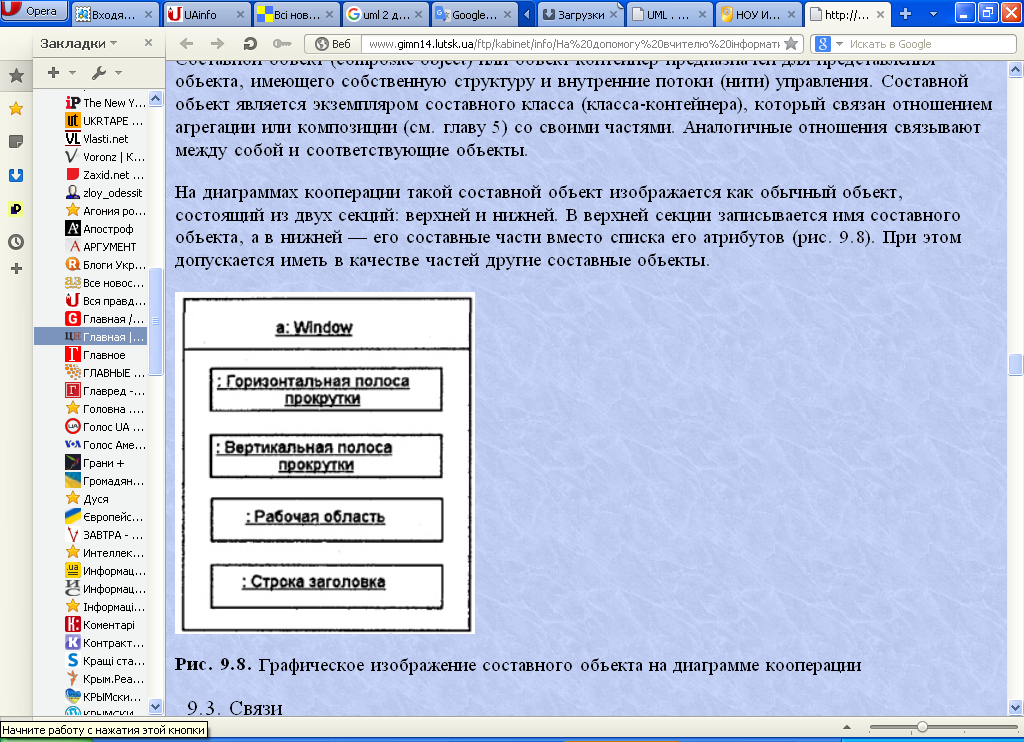


Рис. 9. Графічне зображення вкладеного об’єкта на діаграмі кооперації

**Зв’язки**

Все зв’язки на діаграмі кооперації можуть бути тільки анонімними, для них не вказується кратність. Використовуються такі стереотипи:

* «association» - асоціація (передбачається по замовчуванню, тому цей стереотип можна не вказувати);
* «parameter» - параметр метода. Відповідний об’єкт може бути тільки параметром деякого метода;
* «local» - локальна змінна метода. Її область видимості обмежена лише сусіднім об’єктом;
* «global» - глобальна змінна. Її область видимості поширюється на всю діаграму кооперації;
* «self» - рефлексивний зв’язок об’єкту з самим собою, що допускає передачу об’єктом повідомлення самому себе. На діаграмі кооперації рефлексивний зв’язок зображається петлею в верхній частині прямокутника об’єкту.

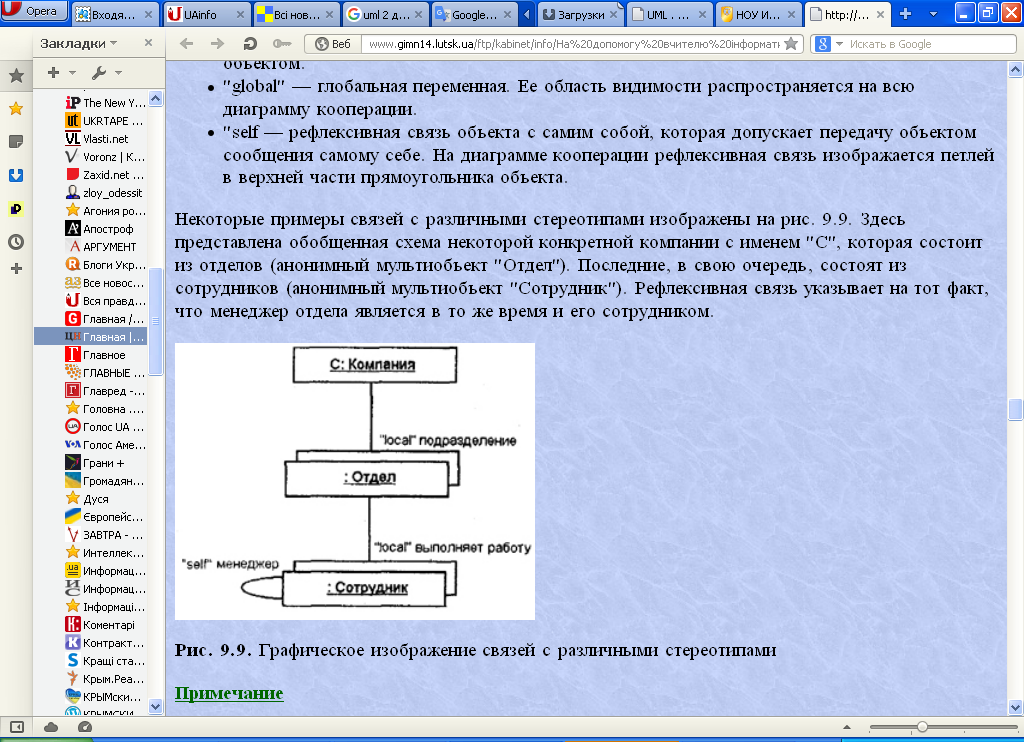


Рис. 10. Графічне зображення зв’язків з різними стереотипами

**Повідомлення**

Повідомлення на діаграмі кооперації специфікує комунікацію між двома об’єктами, один з яких передає іншому деяку інформацію. При цьому, перший об’єкт очікує, що після отримання повідомлення другим об’єктом буде виконана деяка дія.

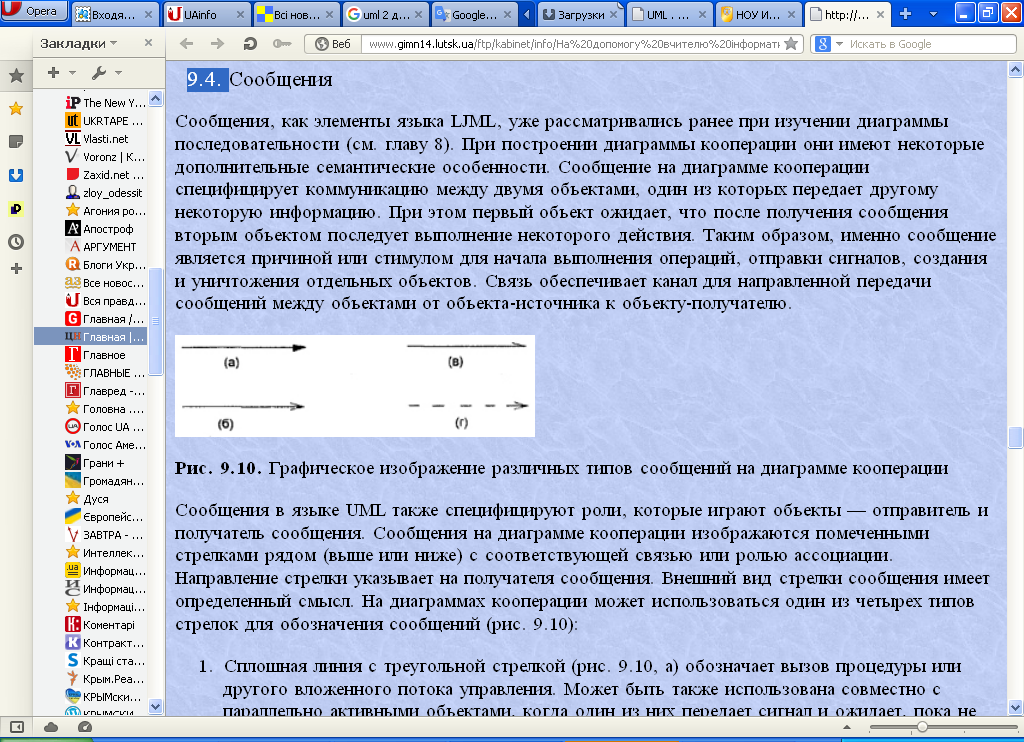


Рис. 11. Графічне зображення різних типів повідомлень на діаграмі кооперації

Зовнішній вид стрілки повідомлення має такий смисл:

1. суцільна лінія з трикутною стрілкою (рис. 11, а) позначає виклик процедури або іншого вкладеного потоку керування. Зазвичай такі повідомлення є синхронними, тобто такі, що ініціюються по завершенні деякої діяльності або при виконання деякої умови;
2. суцільна лінія з V-образною стрілкою (рис. 11, б) позначає простий потік керування. Зазвичай такі повідомлення є асинхронними;
3. суцільна лінія с полустрілкою (рис. 11, в) позначає асинхронний потік керування. Відповідні повідомлення формуються в довільні, заздалегідь не відомі моменти часу, як правило, активними об’єктами. Зазвичай повідомлення такого типу є початковими в послідовності потоку керування й часто ініціюються акторами;
4. пунктирна лінія с V-образною стрілкою (рис. 11, г) позначає повернення з виклику процедури.

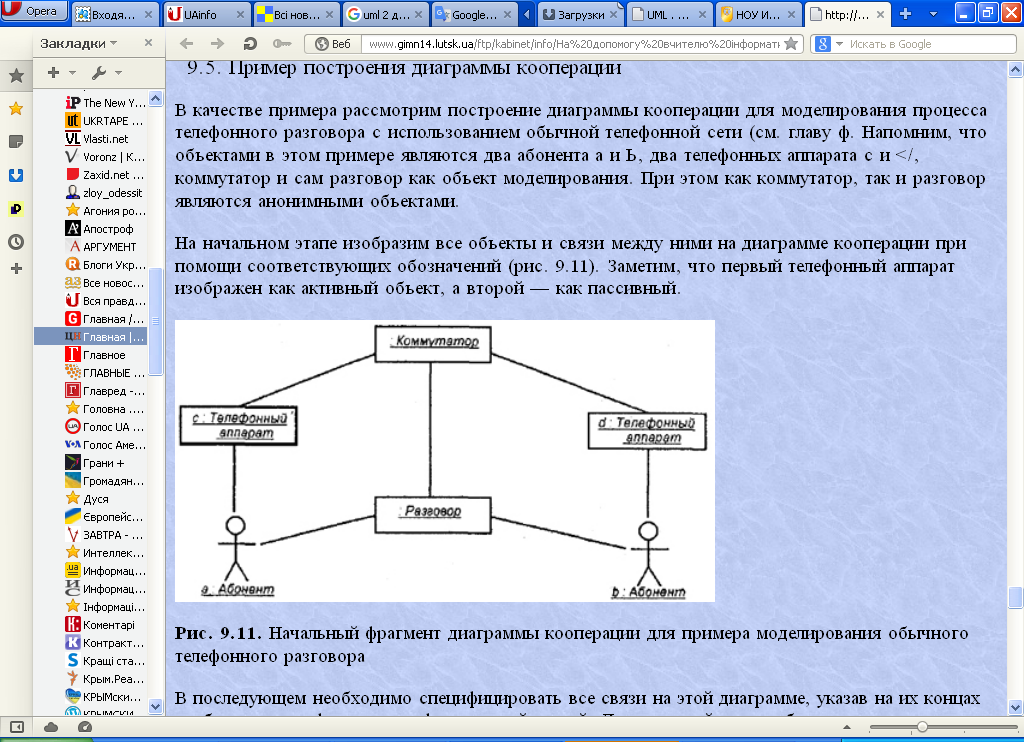
**Приклад побудови діаграми кооперації**.

Рис. 12. Початковий фрагмент діаграми кооперації для прикладу моделювання звичайної телефонної розмови

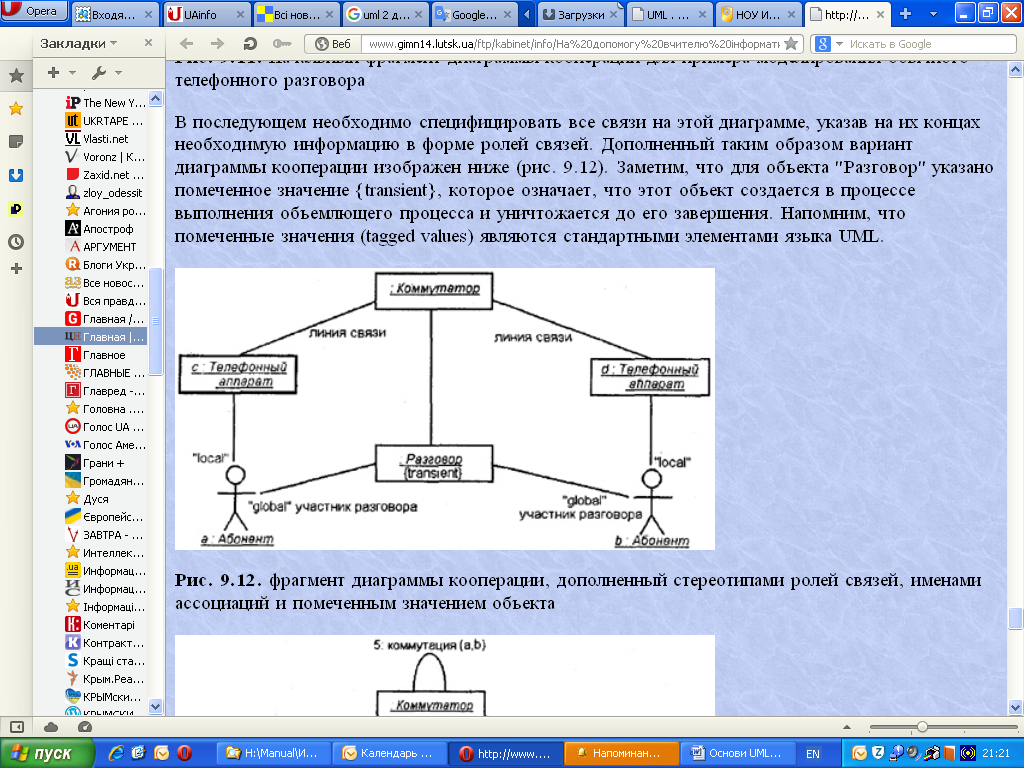


Рис. 13. Фрагмент діаграми кооперації, доповнений стереотипами ролей зв’язків, іменами асоціацій та поміченим значенням об’єкта



Рис. 14. Остаточний варіант діаграми кооперації для прикладу моделювання звичайної телефонної розмови

***Деякі зауваження щодо використання Umbrello UML Modeller.***

В результаті реалізації прецедентів в системі можуть створюватися нові об’єкти. Наприклад, прецедент "Формування звітності" передбачає створення нового об’єкту або сукупності об’єктів, якщо звітів декілька. Аналогічно, відомості про склад груп студентів по навчальним предметам та про склад навчальних предметів групи студентів попередньо повинні бути створені як об’єкти.

Актор з роллю Адміністратора по суті є екземпляром певного класу (в залежності від того, який екземпляр класу буде виконувати цю роль). Це також потрібно враховувати при побудові діаграм кооперації.

Таким чином, формування діаграм кооперації змусить переглянути раніш створені діаграми прецедентів, класів, послідовностей з метою їхнього розширення.

В безкоштовній версії Umbrello UML Modeller не реалізований весь потужний інструментарій UML 2.х, але наявними засобами можна побудувати діаграму кооперації.

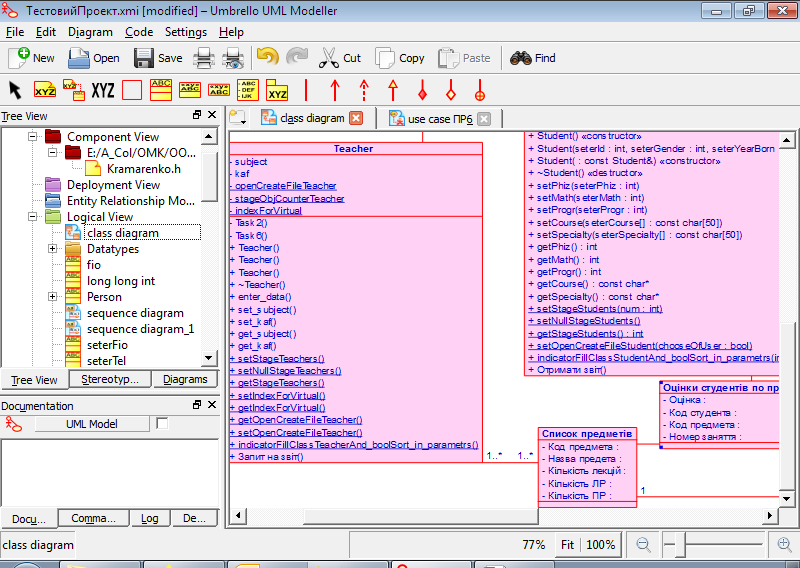
**Контрольні запитання**.

1. Дайте визначення діаграми послідовностей. Чи надає Umbrello UML Modeller можливість реалізувати бінарного розгалуження?
2. Які з розглянутих елементів діаграми послідовності не надано можливості реалізувати в Umbrello UML Modeller?
3. Яке призначення діаграм взаємодії? Як співвідносяться між собою діаграми варіантів використання й діаграми взаємодії?
4. Назвіть два види діаграм взаємодії. Що таке «життєва лінія» на діаграмі послідовності?
5. Як на діаграмі послідовності зображені повідомлення? Що таке самоделегування? Що показує активізація об'єкта?
6. Відмінність кооперативних діаграм від діаграм послідовності. Переваги й недоліки кожного виду взаємодії.

Для побудови UML-діаграм пропонується ряд безкоштовних інструментів за наданим нижче посиланням:

<https://www.guru99.com/uk/best-uml-tools.html> - перелік безкоштовних on-line редакторів .

**ПРИКЛАДИ**



Додано методи

