УДК 004.415:37.015.6

Полотай О. І.

ВИКОРИСТАННЯ ДІАГРАМИ КЛАСІВ UML ДЛЯ ЗАПРОВАДЖЕННЯ ОСВІТНІХ ІТ-ПРОЕКТІВ У ВНЗ

Описано принципи UML моделювання. Розглянуто класифікацію діаграм моделі UML. Наведено детальний опис діаграми класів та розглянуто приклади її використання для запровадження освітніх ІТ-проектів у ВНЗ.

Ключові слова: освіта, ІТ-проект, діаграма, UML, модель, моделювання, метод, класи, об'єктно-орієнтоване програмування.

Polotaj O.

EXAMPLE OF UML CLASS DIAGRAMS FOR THE INTRODUCTION EDUCATION IT PROJECTS IN UNIVERSITIES

Describes the the principles of UML modeling. We consider a classification model diagrams UML. A detailed description of the class diagram and discussed examples of its use for the implementation of educational IT projects in universities

Keywords: education, IT project, diagram, UML, model, modeling method, class, object-oriented programming.

Вступ. Зміна парадигми освіти, розвиток нових інформаційних та телекомунікаційних технологій викликають нову потребу у запровадження у ВНЗ нових освітніх ІТ-проектів. Причиною цього є швидкий розвиток сусупільства, його зайнятість та завантаження, зростання кількості запитів суспільства.

Освітня галузь є складною багато компонентною системою, що становить основу інтелектуального, культурного, духовного, соціального, економічного розвитку суспільства і держави. У вищих навчальних закладах останніми роками інтенсивно запроваджується проекти інформатизації, в тому числі і ІТ-проекти, проекти дистанційного навчання, проекти моніторингу освіти, тощо. Дані проекти полягають у використанні розробленої, внаслідок запровадження ІТ-проекту, інформаційної

системи (ІС) та безперервного її супроводу. Для того, щоб ця ІС ефективно фунціонувала, була зручною у використанні, необхідно використовувати сучасні інформаційно-комунікаційні технології та засоби моделювання. Також, ІС необхідно вдосконалювати, розширювати перелік її основних функцій.

Для ефективного забезпечення вищенаведеного, необхідно вирішувати наступні задачі:

1. Побудова моделі інформаційних процесів ІС.
2. Відображення користувачів і їхніх функцій, підметів автоматизації в прив'язці до структури ІС.
3. Відображення структури інформаційних та фізичних об'єктів ІС та їх взаємозв'язків.
4. Дослідження сценаріїв взаємодії суб'єктів і об'єктів у системі.
5. Побудова моделі станів системи.
6. Побудова правил доступу до ресурсів ІС.

Для вирішення вказаних задач можуть бути використані засоби моделювання мови UML .

Аналіз останніх досліджень і публікацій. Даною проблематикою UML-моделювання займались такі зарубіжні та вітчизняні вчені як Боггс У., Боггс М., Буч Р., Рамбо Дж., Джекобсон А., Вендров А. М., Кватрані Т., Українець А. Г., Бондаренко М. Ф. Оскільки офіційна документація по мові UML досить важка для розуміння; тому виходить багато книг, котрі описують її з різними акцентами. Наприклад, базова система позначень UML популярно і доступно викладена в книзі Мартіна Фовлера (Martin Fowler) [4,2], яка вважається одним з найкращих посібників по вивченню мови. Слід відзначити книги, написані головними авторами UML – Г. Бучем, І. Джекобсоном, Д. Рамбо (в іншому перекладі Румбахом або Рамбо): в [13] викладено детальну інформацію про використання UML (покриває близько 80% мови) та проілюстровано застосування мови на великій кількості прикладів; в [5] розглядається процес об'єктно-орієнтованої розробки програмного забезпечення; в довіднику з UML [5,12] охоплюється вся мова та робиться спроба розкрити її змістовну семантику. Початківцям варто звернути увагу на видання [3].

Постановка завдання. Метою статті є показати суть та можливості використання мови UML для запровадження освітніх ІТ-проектів у ВНЗ.

Виклад основного матеріалу. UML - (Unified Modeling Language) – уніфікована мова моделювання, яка була розроблена для специфікації, конструювання, відображення та документування складних програмних систем. На сьогоднішній день UML знаходить широке застосування в якості неофіційного стандарту при розробках у таких областях, як керування вимогами до інформаційних систем; моделювання бізнес-процесів; аналіз, проектування, кодування і тестування програмного забезпечення. UML може бути використаний не лише для уніфікації представлення даних щодо ІС, але і для їхньої інтеграції, спрямованої на підвищення адекватності багато-модельного дослідження складних систем. Перспективи розвитку UML пов'язані з розвитком нової компонентної розробки додатків (Component-Based Development).

В основу UML покладено декілька об'єктно-орієнтованих методів, кожен із яких був орієнтований на підтримку окремих етапів об'єктно-орієнтованого аналізу та проектування (ООАП) [1,7] :

- метод Граді Буча (Grady Booch), що одержав умовну назву Booch або Booch'91, Booch Lite (пізніше – Booch'93) і вважався найбільш ефективним на етапах проектування та розробки різних програмних систем [1,8];

- метод Джеймса Румбаха (James Rumbaugh), що одержав назву Object Modeling Technique – OMT (пізніше – OMT-2) та найбільше підходив для аналізу процесів обробки даних в інформаційних системах [11];

Таблиця 1

Класифікація діаграм моделі UML

Український варіант	Англійський варіант
1	2
Структурні діаграми:	Structure Diagrams:
Класів	Class diagram
Компонент	Component diagram
Композитної / складеної структури	Composite structure diagram
Кооперації (UML 2.0)	Collaboration (UML2.0)
Розгортування	Deployment diagram
Об'єктів	Object diagram
Пакетів	Package diagram
Діаграми поведінки:	Behavior Diagrams:
Діяльності	Activity diagram
Скінчених автоматів (станів)	State Machine diagram
Прецедентів	Use case diagram
Діаграми взаємодії :	Interaction Diagrams:
Кооперації (UML1.x) / Кооперації (UML 2.0)	Collaboration (UML1.x) / Communication diagram (UML2.0)
Огляду взаємодії (UML 2.0)	Interaction overview diagram (UML2.0)
Послідовності	Sequence diagram
Синхронізації (UML 2.0)	UML Timing Diagram (UML2.0)

- метод Айвара Джекобсона (Ivar Jacobson), Object-Oriented Software Engineering – OOSE, що містив засоби подання варіантів використання, які мають істотне значення на етапі аналізу вимог у процесі проектування бізнес-застосувань.

Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.Діаграми дають можливість представити систему (як ділову, так і програмну) у такому виді, щоб її можна було легко перевести в програмний код.Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність реалізації програмних систем у кілька разів і помітно поліпшити якість кінцевого продукту.UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки. [10]

Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи. Діаграми підвищують супровід проекту і полегшують розробку документації.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів. В UML використовується 13 видів діаграм:

З точки зору об’єктно-орієнтований підходу основною діаграмою UML є діаграма класів. Діаграма демонструє класифікацію моделей з точки зору їх представлення (реалізації). Базовими елементами діаграми класів виступають класи та відношення між ними. Клас може зображатися у деталізованій формі (як клас моделі), скороченій (як класи предметні та інформаційні) та найпростішій (як класи вербальні та формалізовані

Не можна уявити собі “просто модель”, вона буде або “предметною”, або “інформаційною”. Хоча таке абстрактне поняття існує. Абстрактний клас містить властивості, що притаманні всім іншим класам у класифікації. Форми зображення класів - проста умовність. Насправді, будь-який клас має у своєму складі, крім назви, ще й атрибути (характеристики) і операції, які визначають можливу поведінку об’єктів даного класу. Відношення між окремими класами, які можуть бути зображені на діаграмі, визначають, фактично, різні типи ієрархій.

Типова ієрархія визначає класифікацію, а відношення, яке її зображує на діаграмі, у UML називають узагальненням (спадкуванням). Саме таку ієрархію (класифікацію) наведено на рис. 1.

Узагальнення для будь-яких класів має один і той же зміст: спадкування загальних характерис-

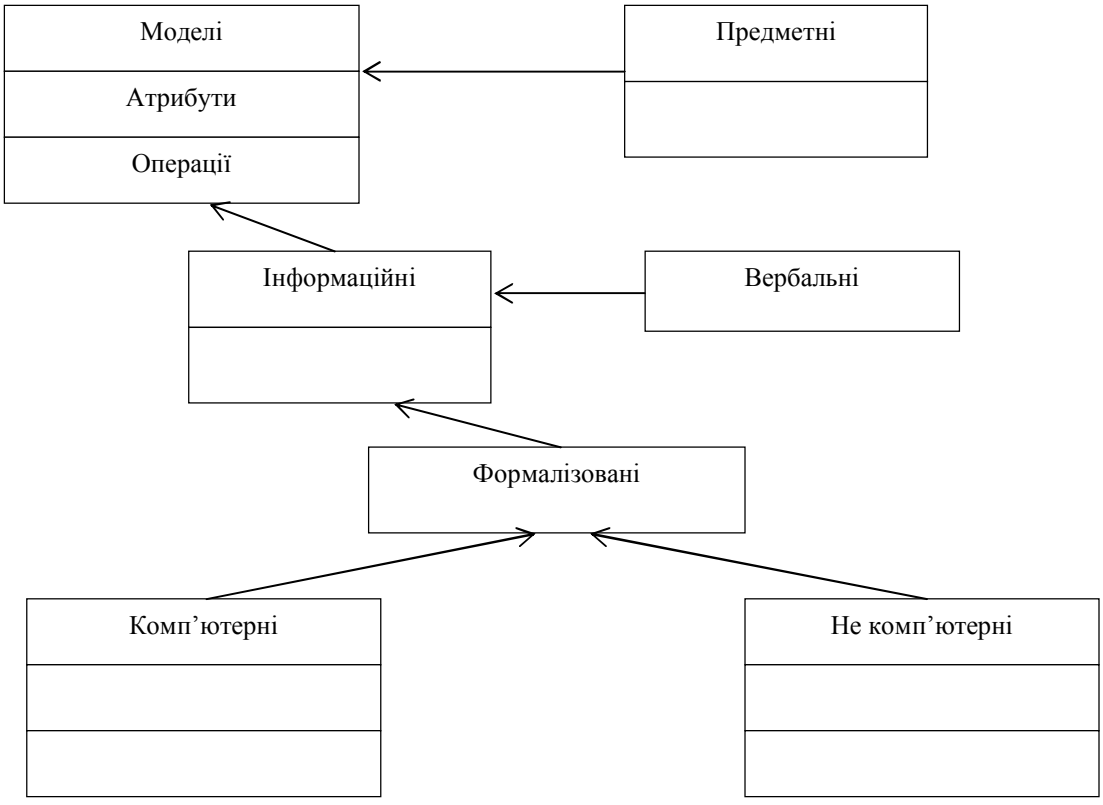

Рис. 1. Приклад діаграми класів UML

тик. Наприклад, класи вербальні та формалізовані мають властивості (тобто успадкували атрибути та операції) інформаційної моделі, що відрізняє їх від предметних моделей. У відношенні “узагальнення” клас, який є носієм загальних характеристик, називають головним класом, а класи, які їх успадковують, - спадкоємцями (підкласами).

Розглянемо діаграму класів ВНЗ як одну із моделей для реалізації запровадження освітнього ІТ-проекту. Кожен клас має свої функціональні особливості, інформація про які зберігатиметься в БД СДН. Діаграма класів UML – це графічне представлення набору елементів, яке зображене у вигляді зв’язаного графу з вершинами (сутностями) і ребрами (зв’язками). Представлені в ній класи є стійкими, тобто такими, що існують протягом всього часу роботи системи.

Під класом [9] розумітимемо іменований опис сукупності об’єктів із спільними атрибутами, операціями, зв’язками та семантикою, що графічно зображений у вигляді прямокутника. Атрибутом класу є іменована властивість класу, що описує множину значень, котру можуть приймати екземпляри цієї властивості. Клас може мати будь-яку кількість атрибутів, або не мати їх зовсім.

Суттєвою деталлю опису особливостей класу є їх видимість, що вказує на можливість використання даної властивості чи операції іншими компонентами системи. В UML існує три рівні видимості: 1) відкритий (public) – будь-який зовнішній класифікатор може використовувати відкриті особливості, позначається знаком “+” перед іменем атрибуту чи операції; 2) захищений (protected) – будь-який потомок даного класифікатора може користуватись його захищеними властивостями, позначається знаком “#”; 3) закритий (private) – тільки даний класифікатор може використовувати закриті властивості, позначається знаком “-”. Видимість визначають для того, щоб приховати деталі реалізації, і показати тільки ті особливості, котрі необхідні для реалізації обов’язків класу. По замовчуванні властивості вважаються відкритими.

За обов’язками, котрі реалізують об’єкти кожного класу їх можна групувати у загальні класи: “користувачі”, “абітурієнти”, “переглядачі”. Клас “користувачі” є батьківським класом (суперкласом), що специфікується властивістю “root”, і означає наявність класів-потомків, котрі наслідують особливості даного класу. У цей клас входять всі фізичні особи, котрі є елементами колективу ВНЗ, та мають можливість вносити зміни в БД ІАС, вони поділяються на два великі класи-потомки: “студенти” та “працівники”. Клас “працівники” також має чотири класи-потомки, що конкретизують об’єкти за виконуваною роллю у структурі ВНЗ. Зрозуміло, що клас “абітурієнти” об’єднує всіх абітурієнтів даного ВНЗу, вони мають можливість подавати та отримувати інформацію від СДН, але не можуть самостійно вносити зміни у БД. В клас “переглядачі” входять особи, котрі прагнуть отримати інформацію про заклад в цілому або певні конкретні дані, вони лише переглядають інформацію не вносячи ніяких змін. В цю категорію входять також батьки студентів, котрі отримують дані про успішність студента. Клас “ВНЗ” містить один об’єкт, що відображає атрибути юридичної особи – представляє навчальний заклад. Класи “факультет”, “кафедра”, “деканат”, “група”, “відділ” відображають елементи організаційної структури закладу.

Взаємодія між класами UML зображається різнотипними лініями. Зв’язок між загальною сутністю – батьківським класом і її конкретним втіленням – підкласом-потомком є узагальненням (generalization). Такий тип зв’язку встановлений між класами “користувачі” та “студенти”, “працівники”, а також між класом “працівники” та конкретними їх категоріями. Всі інші ребра діаграми відображають зв’язок – асоціацію (association), котрий вказує, що об’єкти одного класу певним чином взаємодіють з об’єктами іншого класу. Звичайна асоціація характеризує зв’язок між рівноправними класами, які знаходяться на одному концептуальному рівні. Коли ж асоціація відображає взаємодію між класами типу “частина – загальне” і клас “загальне” знаходиться на вищому концептуальному рівні ніж клас “частина”, то асоціація буде агрегатною.

Існують такі взаємодії між класами: “студент” – “група”; “викладачі”, “навчально-допоміжний персонал” “адміністратори освіти” і “кафедра”; “навчально-допоміжний персонал” “адміністратори освіти” і “деканат”; “адміністратори освіти”, “обслуговуючий персонал” і “відділ”. Якщо зв’язок типу “частина – загальне” є таким, що видалення “загального” приводить до знищення всіх його “частин”, то агрегатну асоціацію називають композитною. Прикладом композитної агрегатної асоціації виступатиме взаємодія між класами “ВНЗ” і “відділ”, “факультет”; “факультет” і “кафедра”, “група”. Звичайна асоціація надає можливість здійснювати навігацію, на лінії асоціації ставиться стрілка, що вказує напрям. Як приклад, клас “кафедра” звітує класу “відділ”, а “відділ” керує роботою “кафедри”. Всі види асоціативних зв’язків характеризуються набором ознак: іменем, що описує на природу взаємодії; кратністю, яка вказує на кількість об’єктів класу з даною роллю, котрі мають входити в кожну асоціацію.

Особливістю запропонованої схеми є те, що кожен клас виступатиме як в ролі суб’єкта так в ролі об’єкта інформаційної системи. Наприклад, клас “факультет” буде виступати суб’єктом по відношенню до класу “група”, яку він формує, та виступатиме об’єктом для класу “ВНЗ”, що встановлює характеристики класу “факультет”.

Висновки. Запропонована діаграма класів ІС ВНЗ відображає властивості об’єктів та суб’єктів, їх взаємодію і дозволяє аналізувати множину станів елементів системи. Для повного переліку конкретних характеристик кожного з класів необхідно провести детальне дослідження руху інформаційних потоків, визначити, які типи даних генерує кожен клас, як ці дані обробляються та зберігаються в системі. Засобами для проведення такого дослідження можуть бути динамічні види діаграм UML: діаграми взаємодії, діаграми станів та діаграми діяльності [9].

На сьогодні для UML-моделювання існує широкий вибір програмних засобів. Найбільше роз-

повсюдженими пакетами програм є Rational Rose, Visual UML, BPwin, Silverrun, Process Analyst, Together, System Architect, Objecteering та інші. Для побудови UML-діаграм можна використовувати MS Visio. Оскільки UML призначений для об'єктно-орієнтованого проектування систем, окремі програмні продукти забезпечують розробку структури програми включаючи засоби захисту інформації. Зокрема Rational Rose забезпечує комплексність підходу і інтеграцію з MS Visual Studio на рівні прямої й оберненої генерації кодів, інжиніринг і реінжиніринг модулів і бібліотек форматів EXE, DLL, TLB, OCX, підтримку CORBA, IDL, ADO, COM, Java.

Мова UML є потужним, гнучким засобом моделювання, з відкритим до вдосконалення описом стандарту. Неоднозначність як деяких конструкцій самої мови, так і поглядів на семантику мови та наявність в її специфікації неформальних описів потребує подальшого розвитку формальної основи для повної та несуперечливої інтерпретації мови.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Booch G. Object-oriented analysis and design with applications. Second edition. – The Benjamin/ Cummings Publishing Company Inc., 1994. – 589 p. 8.
2. Booch G., Rumbaugh J., Jacobson I. The Unified Modeling Language User Guide. – MA.: Addison-Wesley Publishing Co., 1999. – 512 p. 3.
3. Chonoles M. J., Schardt J.A. UML 2 for Dummies. – Hungry Minds, 2003. – 412 p. 7.
4. Fowler M., Scott K. UML Distilled. – MA.: Addison-Wesley, 2000. – 472 p. 1.
5. Jacobson I., Booch G., Rumbaugh J. The Unified Software Development Process. – MA.: Addison-Wesley Publishing Co., 1999. – 512 p. 4.
6. Rumbaugh J., Jacobson I., Booch G. Unified Modeling Language Reference Manual. – MA.: Addison-Wesley Publishing Co., 1999. – 576 p. 5.
7. Буч Г. Объектно-ориентированное проектирование с примерами применения / Буч Г. ; [пер с анг.] – Москва: Конкорд, 1996. – 519 с. 9.
8. Буч Г. Объектно-ориентированный анализ и проектирование: с примерами приложений на C++. / Буч Г. ; [пер с анг.]– Санкт-Петербург: "Издательство Бином", "Невский диалект", 1998. – 560 с. 10.
9. Буч Г. Язык UML: Руководство пользователя / Буч Г., Рамбо Дж., Якобсон А. – М.: ДМК, 2000. – 356 с.: ил. 13.
10. Київський освітній портал [Електронний ресурс]: http:// www.edu.kiev.ua 12.
11. Рамбо Д. Тенденции в развитии языка UML и разработки ПО / http://www.interface.ru/fset.asp?Url=/rational/umltend.htm. 11.
12. Рамбо Д. UML. Специальный справ очник / Рамбо Д., Якобсон А., Буч Г. ; [пер. с анг.] – Санкт-Петербург: Изд. дом "Питер", 2002. – 654 с. 6.
13. Фаулер М. UML в кратком изложении. Применение стандартного языка объектного моделирования / Фаулер М., Скотт К. ; [пер. с англ.] – Москва: Мир, 1999. – 416 с. 2.
14. Чекурін В. Модель функціонування інформаційно-аналітичної системи багаторівневого моніторингу якості освіти / Чекурін В., Острей С., Острей О. // Фізико-математичне моделювання та інформаційні технології. – Львів. – 2007. – №6. – С. 66-76.
15. Якобсон А. Унифицированный процесс разработки программного обеспечения / Якобсон А., Буч Г., Рамбо Дж. – СПб.: Питер, 2002. – 498 с.:ил.