УДК 004(076.5)

ДІАГРАМА КЛАСІВ UML ЯК ЗАСІБ МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ МОНІТОРИНГУ ОСВІТИ

Острей О.Р.,
Волинський національний університет ім. Лесі Українки

У статті розглянуто особливості функціонування інформаційно-аналітичної системи багаторівневого моніторингу освіти. Описана діаграма класів UML для моделювання інформаційно-аналітичної системи багаторівневого моніторингу вищого навчального закладу. Ця діаграма описує статичний вид системи відносно процесів, відображає властивості об'єктів та суб'єктів, їх взаємодію. Також дозволяє аналізувати множину станів елементів системи.

In the article the features of functioning of the informational-analytical system for multilevel monitoring of education are considered. The described class diagram UML for the design of the informational-analytical system for multilevel monitoring of higher educational establishment. This diagram describes the static type of the system in relation to processes, represents properties of objects and subjects, their co-operation. Also allows to analyzed the great number of the states of elements of the system.

1. Особливості функціонування інформаційно-аналітичної системи моніторингу освіти

Освітня галузь є складною багатокомпонентною системою, що становить основу інтелектуального, культурного, духовного, соціального, економічного розвитку суспільства і держави. Моніторинг забезпечує вмотивоване прийняття управлінських рішень для якісного виконання всіх завдань системи. За результатами аналізу структури та особливостей функціонування освітніх установ [1] зроблено висновок про необхідність створення інформаційно-аналітичної системи (ІАС) багаторівневого моніторингу якості освіти, що являє собою інтегроване мережеве програмне середовище, яке можна налаштовувати під потреби різних груп користувачів – учня/студента, вчителя/викладача, адміністраторів освіти різних рівнів (факультету, навчального закладу, району, області, регіону тощо).

ІАС призначена для автоматизації у режимі колективного використання наступних процесів: планування контрольних та моніторингових заходів, підготовки тестових матеріалів, вимірювання рівня знань учнів/студентів, моніторингу якості освіти на різних рівнях, візуалізації результатів у табличному та графічному вигляді [2]. Крім того програмне середовище виконуватиме: вимірювання знань впродовж заданого проміжку часу із заданою дискретністю на рівнях окремого учня/студента та учнівських/студентських колективів різного масштабу; аналіз якості знань на цих рівнях; також оцінювання якості роботи викладацьких колективів та органів управління освітою.

Запропоновані в [1] схеми взаємозв'язків та віддаленої взаємодії між об'єктами і суб'єктами ІАС становлять основу для продовження розробки програмного комплексу, що передбачає поділ всього об'єму робіт на окремі етапи [3]. Одним з таких етапів є процес моделювання сховища інформації. Адже моніторинг значної кількості об'єктів та суб'єктів системи створює великі потоки даних. Для їх ефективного опрацювання потрібно структурувати систему збереження інформації.

2. Опис діаграми класів ІАС багаторівневого моніторингу освіти

Прикладом ієрархічно організованого сховища є база даних (БД). Найбільш поширеним засобом абстрактного представлення структури БД є ER-модель (entity-relationship model) типу "сутність-зв'язок" [4]. Діаграми класів UML включають ER-діаграми, як частковий випадок. Але, якщо в класичних ER-діаграмах увага зосереджена

тільки на даних, то в діаграмах класів UML моделюється також і поведінка [5]. В реальній БД такі логічні операції зазвичай трансформуються в тригери або вбудовані процедури.

Розглянемо діаграму класів ІАС вищого навчального закладу як одну із моделей реалізації ІАС багаторівневого моніторингу якості освіти. Кожен клас має свої функціональні особливості, інформація про які зберігатиметься в БД системи. Діаграма класів UML (рис.1) – це графічне представлення набору елементів, зображена 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* [6] 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. На діаграмі рис.1 термін "Attributes" свідчить про те, що клас має певні властивості, які не оголошені. Порожній рядок в кінці блоку опису атрибутів теж вказує, що не всі властивості класу є оголошеними. *Операцією класу* називається іменована послуга, яку можна викликати у будь-якого об'єкта даного класу. Аналогічно до атрибутів, клас може мати будь-яку кількість операцій, або не мати їх зовсім На діаграмі рис.1 термін "Operations" свідчить про те, що клас надає певні послуги, які не оголошені. Порожній рядок в кінці блоку опису операції теж вказує, що не всі послуги класу є оголошеними. Повне оголошення всіх властивостей і операцій уточнюється в процесі проектування динаміки потоків даних.

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 ІАС, 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*.

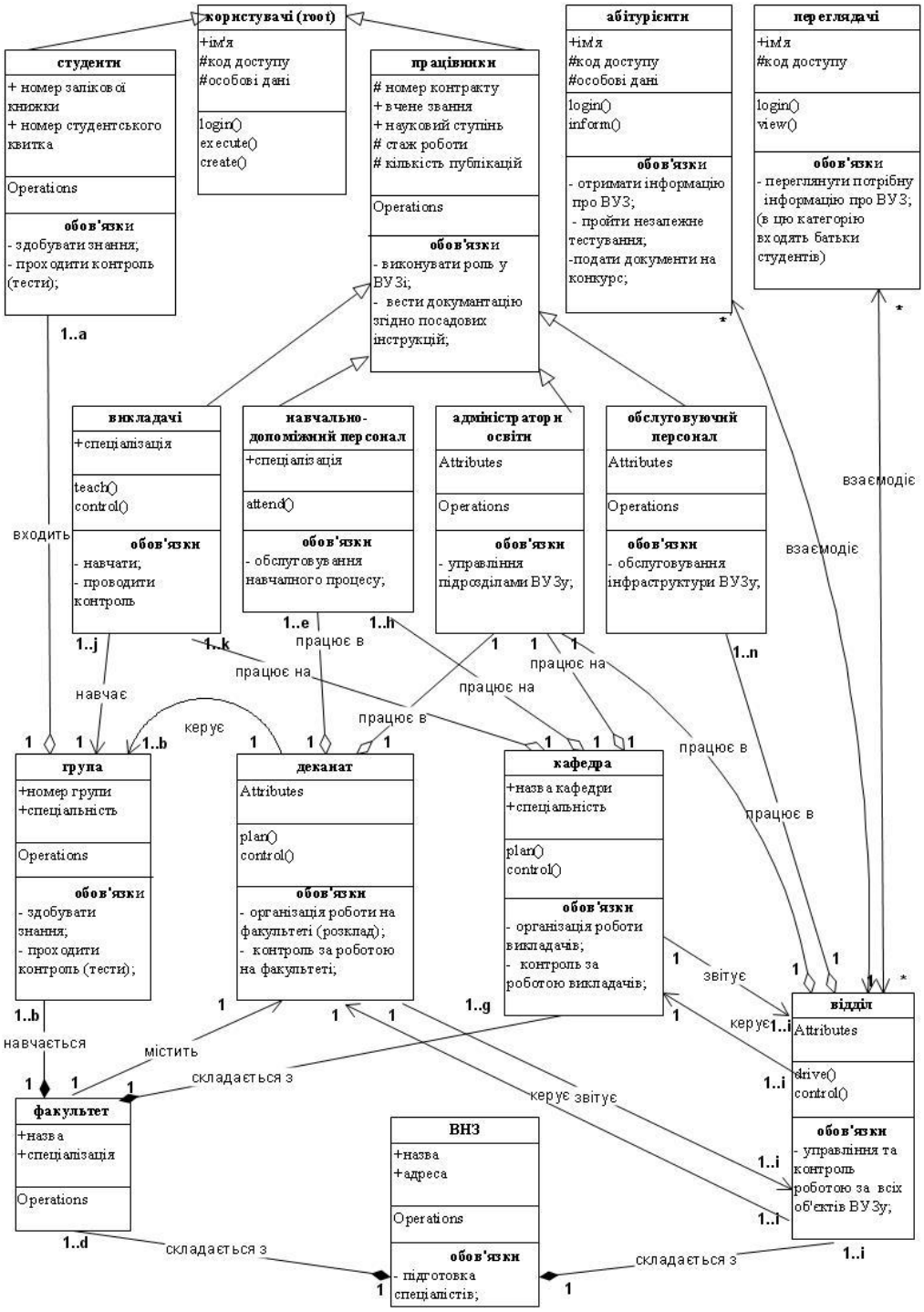

Рис. 1. Діаграма класів ІАС багаторівневого моніторингу ВНЗ

На діаграмі – це взаємодія 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. Найпоширенішим способом представлення кратності є зазначення конкретного числа або діапазону. Агрегатна асоціація між класами “студент” – “група” вказує, що в кожній 1 групі навчається певна кількість студентів (від 1 до a, де a – максимально можлива кількість студентів у групі, що регламентується правилами організації навчального процесу). Кожен ВНЗ складається з конкретної кількості факультетів та відділів відповідно до статуту освітньої установи. Аналогічні кратні зв’язки встановлені між усіма класами діаграми.

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”.

3. Аналіз множини станів класів

Кожен клас діаграми виконуватиме обмежене число функцій, множина яких залежить від його стану. Стан характеризується набором атрибутів об’єкта і операцій, які їх змінюють. Множина функцій, дозволених в даний момент для виконання, називається операційним полем об’єкта (класу). Зміна операційного поля відбувається лише після певної дії зв’язаних об’єктів. Слід зауважити, що множина станів об’єкта є замкнутою, і будь-які комбінації дій зв’язаних об’єктів переводять клас в один з відомих станів. Список усіх станів об’єкту буде сформованим лише після створення повного переліку атрибутів об’єкту, котрі зберігатимуться в БД. Комбінуючи об’єкти в різних станах можна описати множину станів ІАС вцілому. Ця множина також має бути замкненою для запобігання втрати цілісності та для стабільної роботи ІАС. Для прикладу, розглянемо множину станів класу “викладачі”. Базовим станом вважатимемо стан готовності до виконання завдань [8], він характеризується найширшим операційним полем, а також значеннями атрибутів в початковому положенні (значення за замовчуванням в момент запуску системи). З цього стану клас “викладачі” може перейти в стан виконання (перехід 1), зображеного на рис. 2.

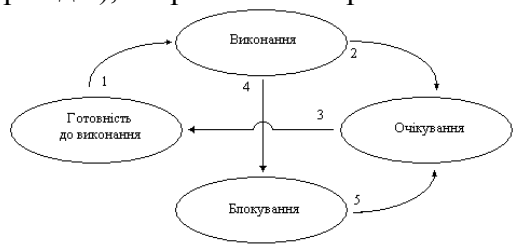

Рис.2 Схема переходів між станами

Він характеризується зменшеним до кількох зв’язаних функцій, операційним полем, та блокуванням всіх атрибутів, котрі не використовуються у вище названих функціях. Такими функціями виступатимуть операції по керуванню БД запитань, відповідей і тестів [1], що представлена дією create() базового класу root. На момент виконання цієї операції, будуть заблоковані атрибути: стаж роботи, кількість публікацій, номер контракту та інші, які не використовуються в процесі створення тестів. З стану виконання, за умови успішного завершення операції, клас переходить в стан очікування (перехід 2), де всі атрибути отримують початкові значення. З стану очікування клас переводиться в стан готовності з збільшенням операційного поля (перехід 3). Якщо за якихось причин не можливо завершити операцію (create() може не виконатись через відсутність зв’язку з БД) клас переходить в стан блокування (перехід 4), в якому є можливість викликати процедури та функції обробки помилок ІАС. Також в цьому стані є доступними лише атрибути, котрі відповідають за збереження інформації про тип помилки (її код), момент виникнення, ім’я операції, яка

створила помилку та операції яка її оброблятиме. Слід зауважити, що інформація про помилки буде закладена як атрибути батьківського класу root. З стану блокування клас переходить в стан очікування, в якому всі атрибути отримують початкові значення. Кожен перехід системи в стан блокування, автоматично створює колізію або тупикову ситуацію. В силу обмеженості апаратних та програмних інструментів, що обробляють колізії, не завжди можна вивести клас з стану блокування. Тому в моделі системи доцільно розглядати методи уникання подібних ситуацій. В сучасних операційних системах для уникнення колізій використовують мережі Петрі або семафорні примітиви [8]. Обидва варіанти базуються на основі властивостей напрямлених графів. Щоб отримати такий граф, потрібно спростити діаграму класів, видаливши з неї усі описи об'єктів. Цей граф дозволяє виявити задачу, яка оброблятиметься в циклі, і знайти способи уникнення тупикового стану.

4. Концепції подальшого розвитку ІАС багаторівневого моніторингу

Запропонована діаграма класів ІАС багаторівневого моніторингу ВНЗ відображає властивості об'єктів та суб'єктів, їх взаємодію і дозволяє аналізувати множину станів елементів системи. Однак вона описує статичний вид системи відносно процесів 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 [6]. Кожна динамічна діаграма UML доповнює статичну діаграму класів і представляє окремий аспект роботи всієї системи. Зокрема, діаграми взаємодії зображаються частковими випадками: діаграмами послідовностей, що відображають часову послідовність повідомлень, якими можуть обмінюватися класи; та діаграмами кооперацій – структурно організованою схемою обміну повідомленнями між класами. Діаграми станів представляють автомат, що містить стани, переходи, події і види операцій об'єктів. Діаграми діяльності зображають переходи потоку керування від однієї операції до іншої. Так як UML – мова, котра містить набір графічних інструментів для візуалізації з чіткою семантикою, то проектні схеми, створені в ній не тільки спрощують розуміння структурної організації програмного продукту, а й кожна модель, створена одним розробником однозначно інтерпретується іншими. Це допоможе створити прототип ІАС, експериментальне дослідження якого повинне підтвердити доцільність проектних рішень.

ЛИТЕРАТУРА

1. Чекурін В., Острей С., Острей О. Модель функціонування інформаційно-аналітичної системи багаторівневого моніторингу якості освіти // Фізико-математичне моделювання та інформаційні технології. – Львів. – 2007. – №6. – С. 66-76.
2. Чекурін В.Ф., Острей С.В., Острей О.Р. Моделювання програмної системи для багаторівневого моніторингу якості освіти // Інформаційні технології та інформаційна безпека в науці, техніці та навчанні "ІНФОТЕХ-2007". Частина 2: Матеріали між нар. наук.-практ. конф., м.Севастополь, 10-16 вересня 2007р. – Севастополь: Вид-во СевНТУ, 2007. – С.147-148.
3. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения. – СПб.: Питер, 2002. – 498 с.:ил.
4. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. Полный курс.: Пер.с англ. – М.: Издательский дом "Вильямс", 2004. – 1088с.: ил.
5. Кузнецов С.Д. Концептуальное проектирование реляционных баз данных с использованием языка UML // Новые публикации, 15.03.2008. – www/CITForum.ru.
6. Буч Г., Рамбо Дж., Якобсон А Язык UML: Руководство пользователя. – М.: ДМК, 2000. – 356 с.: ил.
7. Мюллер Р.Дж. Базы данных и UML. Проектирование / Первод. с англ. Е. Молодцова. – М.: Издательство "Лори",2002. – 432с.: ил.
8. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. – С.-Пб.:Питер, 2001. – 740 с.: ил.