DOI 10.31891/2307-5732-2019-277-5-248-251
УДК 004.273

Л.О. КОВТУН, Р. ФРАНЧУК
Хмельницький національний університет
В.М. ТКАЧУК
Вінницький технічний коледж

ВИБІР АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ НАВЧАЛЬНОЇ СИСТЕМИ

Інформаційна насиченість сучасного світу вимагає спеціальної підготовки та певної адаптації навчального матеріалу перед його поданням учням та студентам для того, щоб у візуально доступному для сприйняття вигляді надати основні або необхідні відомості, які будуть зрозумілими, легкодоступними та легкозасвоюваними. Доцільність використання візуалізації навчальної інформації зумовлена необхідністю врахування когнітивних особливостей сучасного покоління учнів та студентів, а також потребою в компактному поданні навчального матеріалу у вигляді, найбільш зручному для його сприйняття, розуміння, засвоєння і запам'ятовування. Використання різних операційних систем в освітньому процесі натикається на певні труднощі, головною з яких є відсутність або не достатній функціонал відповідного програмного забезпечення. Тому актуальним постає питання щодо такого програмного забезпечення, яке могло б функціонувати на різних платформах. У статті показано вибір моделі архітектури програмного забезпечення, що зосереджено на такому технотренді для візуалізації інформації як веб-технології.

***Ключові слова**: архітектура програмного забезпечення, трирівнева архітектура, веб-технології.*

L.O. KOVTUN, R. FRANCHUK
Khmelnytskyi National University
V.M. TKACHUK
Vinnytsia Technical College

SELECTION OF SOFTWARE ARCHITECTURE INFORMATION TRAINING SYSTEM

The rapid penetration of information and communication technologies into human life and the overload of information flows require modern education to adopt new technologies, changes in teaching methods, ways of presenting educational information and the introduction of new learning technologies that would be effective in the present. The informational saturation of the modern world requires special training and some adaptation of the teaching material before it is presented to students and students in order to provide students with essential or necessary information in a visually perceptible form that is clear, easily accessible and easy to digest. The expediency of using visualization of educational information is conditioned by the need to take into account the cognitive characteristics of the current generation of students and students, as well as the need for a compact presentation of educational material in the form most suitable for its perception, understanding, assimilation and memorization. The use of different operating systems in the educational process encounters certain difficulties, the main of which is the lack or lack of sufficient functionality of the corresponding software. Therefore, the question arises as to the kind of software that could function on different platforms. The article illustrates the choice of a software architecture model that focuses on such technology rendering to visualize information as web technology.

***Keywords**: software architecture, three-tier architecture, web application.*

Зародження та розвиток інформаційного суспільства зумовлює вимоги до інформатизації освіти. Це дозволяє розв'язувати головну задачу – підвищення якості освіти на основі використання сучасних інформаційних і комунікаційних технологій. Задоволення інформаційних потреб, розвиток творчого та інтелектуального потенціалу студентів і адекватне використання інформаційних ресурсів в різних сферах людської діяльності можливе при формуванні в освітньому процесі умінь роботи з електронними засобами обробки і передачі інформації.

Згідно з вимогами до інформаційно-освітнього середовища заклад вищої освіти повинен включати в себе комплекс інформаційних освітніх ресурсів, в тому числі цифрових освітніх ресурсів, сукупність технологічних середовищ, інформаційних і комунікаційних технологій, систему сучасних педагогічних технологій, що забезпечують навчання в сучасному інформаційному освітньому середовищі.

Інформаційна навчальна система не може існувати без програмного забезпечення, яке повинно задовольняти інформаційні потреби кожного користувача системи від викладача до студента. Тому вдалий вибір програмного забезпечення та його архітектури є важливим етапом проектування програмного забезпечення інформаційної навчальної системи.

Архітектура програмного забезпечення (ПЗ) (англ. software architecture) – спосіб структурування програмної або обчислювальної системи, абстракція елементів системи на певній фазі її роботи. Система може складатись з кількох рівнів абстракції і мати багато фаз роботи, кожна з яких може мати окрему архітектуру [1].

Дослідженню архітектури ПЗ потрібно приділити значну вагу, оскільки саме на етапі проектування та аналізу розроблюваного ПЗ стає можливим визначити як найкраще розбити систему на частини, яким чином ці частини визначаються та взаємодіють одна з одною, як між ними передається інформація, як ці частини розвиваються поодинці і як все вище описане найкраще записати, використовуючи формальну чи неформальну нотації. Після етапу дослідження, аналізу та формулювання вимог до архітектури ПЗ розробнику потрібно перейти до проектування архітектури ПЗ, задачею якого є перетворення вимог до

системи у вимоги до ПЗ і побудова на їх основі архітектури системи.

Побудова архітектури системи здійснюється шляхом визначення цілей системи, її вхідних і вихідних даних, декомпозиції системи на підсистеми, компоненти або модулі та розроблення її загальної структури. Існує ряд методів, що використовуються для проектування архітектури системи, кожний з яких пропонує свій шлях побудови архітектури. Ці методи містять визначення концептуальної, об'єктної й інших моделей за допомогою відповідних конструктивних елементів (блок-схем, графів, структурних діаграм тощо).

Архітектури залежно від кількості рівнів, компонентів та способів їх подання та об'єднання в систему можна поділити на чотирирівневі та трирівневі. Один зі шляхів архітектурного проектування, який має назву загальносистемний, – традиційний неформальний підхід до визначення архітектури системи, її компонентів, способів їхнього подання й об'єднання в систему. Така архітектура є чотирирівневою і містить у собі системні компоненти, які здійснюють взаємодію з периферійними пристроями комп'ютерів та використовуються під час побудови операційних систем; загальносистемні компоненти, які забезпечують взаємодію з універсальними сервісними системами середовища роботи прикладної системи, такими як операційні системи, СКБД, системи баз знань, системи керування мережами і т.п.; специфічні компоненти певної прикладної області, що входять до складу компонентів програмної системи і призначені для розв'язання задач в межах означеної області; прикладні програмні системи, що призначені для виконання завдань з обробки інформації, які постають перед окремими групами споживачів інформації з різних предметних областей (офісні системи, системи бухгалтерського обліку й ін.) і можуть використовувати компоненти нижчих рівнів [2–4].

Компоненти кожного з виділених рівнів використовуються, як правило, на своєму або вищому рівні. Кожен рівень відбиває відповідний набір знань, умінь і навичок фахівців, що створюють або використовують компоненти. Цей набір визначає відповідний розподіл фахівців програмної інженерії на аналітиків, системників, прикладників, програмістів й ін. В ході проектування архітектури програмна система розглядається як композиція компонентів третього рівня з доступом до компонентів першого і другого рівнів. Тобто архітектурне проектування – це розроблення компонентів третього рівня, визначення вхідних і вихідних даних рівнів ієрархії компонентів і їхніх зв'язків. Результат проектування — архітектура й інфраструктура, що містять у собі набір об'єктів, з яких можна формувати деякий конкретний вид архітектурної схеми для конкретного середовища виконання системи, а також набір елементів керування і контролю. Проектування архітектури системи завершується створенням опису, в якому відображені зафіксовані проектні рішення, логічна і фізична структура системи, а також способи взаємодії об'єктів.

Для реалізації веб-застосунків використовується архітектура клієнт/сервер, яка передбачає обмін даними з сервером бази даних, на якому у формі процедур розташовується основна частина бізнес-логіки, або з виділеним файловим сервером. Тобто програмний код додатку буде розміщений на віддаленому комп'ютері – сервері**.** Доступ до серверу має клієнт **–** це комп'ютер, який використовується користувачем для доступу до серверу. Цей доступ відбувається передачею даних використовуючи браузер, за допомогою якого відбувається надсилання запиту на віддалений сервер, де розміщені файли цього сайту. На стороні сервера відбувається обробка цього запиту і сформується відповідь, яку отримає клієнт, а саме користувач. За допомогою браузера, користувач може переглянути відповідь у вигляді веб-сторінки або іншого файлу.

Перевагами такого архітектурного стилю клієнт/сервер є велика безпека (всі дані зберігаються на сервері, який зазвичай забезпечує більший контроль безпеки, ніж клієнтські комп'ютери); централізований доступ до даних (адміністрування доступу до даних набагато простіше, ніж в інших архітектурах, оскільки дані зберігаються тільки на сервері); простота обслуговування (ролі і відповідальність обчислювальної системи розподілені між декількома серверами, що взаємодіють один з одним по мережі, тому клієнт не відчуватиме впливу подій, що відбуваються з сервером, таких як ремонт, оновлення або переміщення).

Недоліками архітектурного стилю клієнт/сервер є те, що несправність сервера зробить систему недоступною для користувачів, а при недостатньо потужному сервері з'являються сильні затримки в роботі системи.

З огляду на всі переваги та недоліки описаних архітектур для побудови інформаційної навчальної системи було прийнято рішення використовувати трирівневу архітектуру клієнт/сервер, схема роботи якої наведена на рис. 1.

Рис. 1. Схема роботи архітектури клієнт-сервер

Трирівнева клієнт-серверна архітектура передбачає відділення прикладного рівня від управління даними. Виокремлюється окремий програмний рівень, на якому зосереджується прикладна логіка застосування. Програми проміжного рівня можуть функціонувати під управлінням спеціальних серверів застосувань, але запуск таких програм може здійснюватися і під управлінням звичайного веб-сервера. Нарешті, управління даними здійснюється сервером даних. Наприклад, на клієнтському рівні для роботи з системою користувач використовує стандартне програмне забезпечення – звичайний браузер. Це позбавляє його необхідності завантажувати та інсталювати спеціальні програми (хоча інколи така необхідність все-таки виникає). Середній рівень (middleware) — рівень обробки

запиту: сервер викликає серверні програмні модулі, які забезпечують обробку запиту і в разі потреби звертаються до сервера даних. На рівні даних сервер даних здійснює операції з даними, що зберігаються в системі та складають її інформаційну основу. Зокрема, він може здійснити вибірку з інформаційної бази відповідно до запиту та передати її модулю проміжного рівня для подальшої обробки. Дані, з якими працює сервер даних, найчастіше організовані як реляційна база даних. Найчастіше сервер даних і серверні модулі проміжного рівня розміщуються на одному комп'ютері, хоч і являють собою окремі і логічно незалежні програмні модулі.

Тобто алгоритм роботи веб-сторінки навчальної інформаційної системи буде наступний:

- запит клієнтом надсилається на веб-сервер;
- запит клієнта отримується веб-сервером;
- запит піддається певній обробці з уточненнями;
- відповідний запит від веб-сервера надсилається до бази даних;
- база даних формує певний результат (робить вибірку по вказаних критеріях) і відправляє його на веб-сторінку або веб-застосунок;
- відповідь на основі отриманих результатів запиту формується у зручному для клієнта вигляді.

На основі викладених міркувань та аналізу алгоритму роботи веб-сторінки було створене ПЗ інформаційної навчальної системи, що базується на основі трирівневої архітектури. Діаграма станів входу користувача зображена на рис. 2.

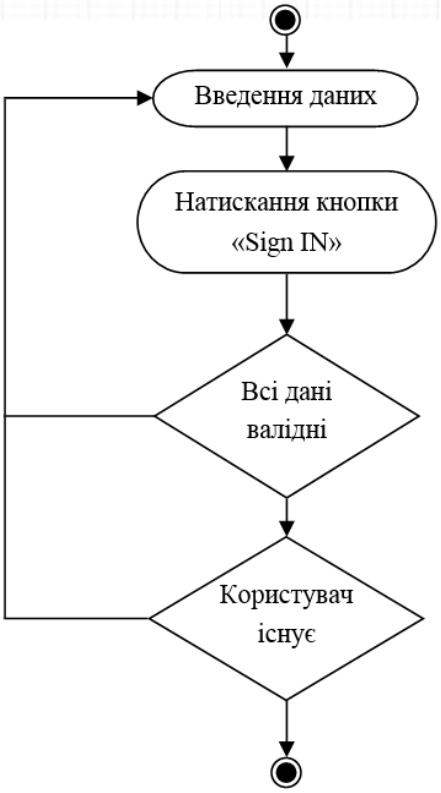

Рис. 2. Діаграма станів входу користувача

Як видно з діаграми, можна виділити наступні стани входу користувача: введення даних користувачем шляхом натиснення кнопки «SIGN IN». Наступним кроком є перевірка даних на валідність. Якщо перевірка не пройшла, то алгоритм почнеться заново з введення даних. Якщо дані валідні, то відбувається перевірка на існування користувача у системі. Якщо користувач у системі знайдений, то алгоритм починається з початку. В іншому випадку алгоритм має успішне завершення.

На рис. 3 зображено діаграму реєстрації на курс.

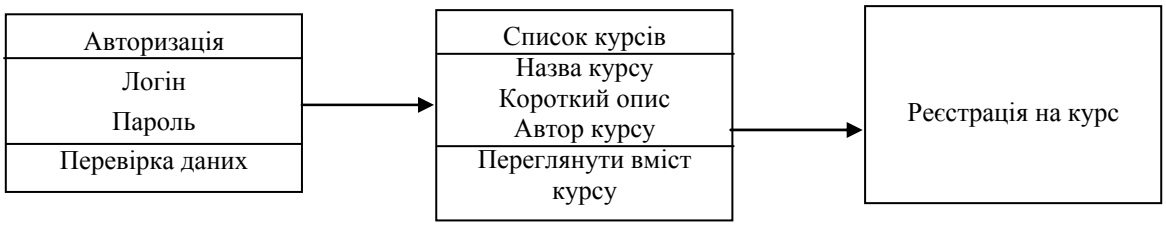

Рис. 3. Діаграма реєстрації користувача на курс

Для того, щоб розмежувати ролі користувачів у системі (наприклад, викладач, студент, адміністратор), у базі даних створена додаткова таблиця «ролі», яка пов’язана з таблицею «користувачі». Ці ролі надаються адміністратором після реєстрації користувача. Таблиці пов’язані між собою за ключовим полем «UserID». Організація бази даних, що містить відомості про користувачів та їх ролі у системі, наведена на рис. 4.

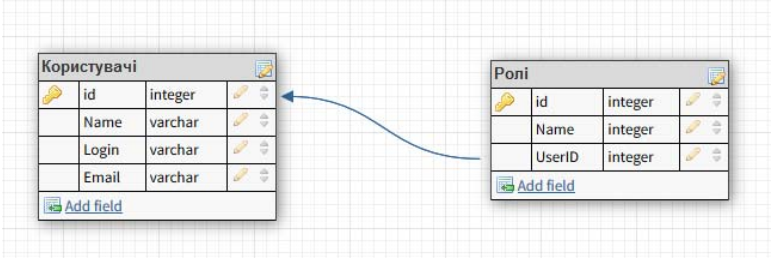

Рис. 4. Організація бази даних, що містить дані про користувачів та їх ролі в системі

Кожен користувач з роллю «викладач» матиме можливість доповнити або видалити непотрібні на його думку дані з курсу. Для кожного курсу у базі даних створена таблиця «курси», з якою пов’язана

таблиця «розділи». До таблиці «курси» доступ матимуть користувачі. Але залежно від ролі це буде або звичайний перегляд, або можливість додавання змін до курсу. Організація бази даних, що містить курси, відомості про авторів, розділи курсів, тести із запитаннями до них та варіанти відповідей на запитання зображені на рис. 5.

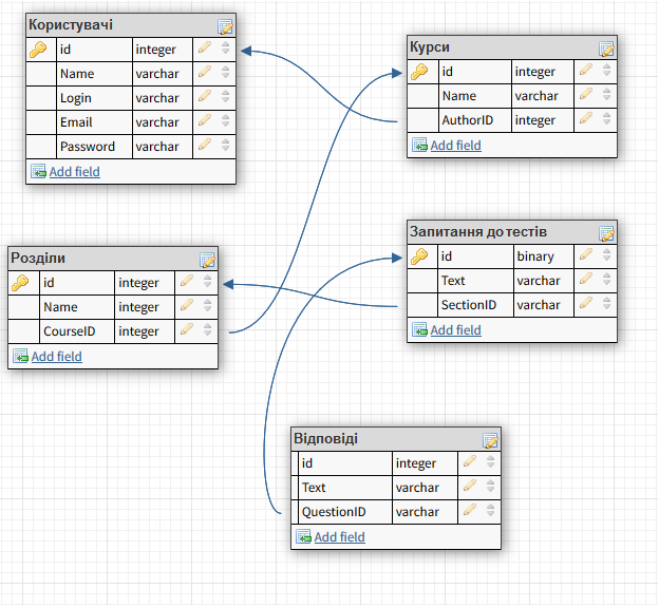

Рис. 5. Організація бази даних, що містить курси, відомості про їх авторів, розділи, запитання до них та варіанти відповідей на запитання

Отже, використання трирівневої архітектури для побудови інформаційної навчальної системи дозволило забезпечити виконання поставленого завдання – створити інформаційну навчальну систему, що розділяє користувачів за їх ролями в інформаційній навчальній системі на тих, хто може доповнювати даними систему та їх оновлювати (викладач), а також на тих, хто цими даними буде користуватись (студенти).

Висновки. Під час створення інформаційної навчальної системи дуже важливим є складання правильної і стабільної функціональної спеціалізації. Враховуючи зручність використання для побудови системи було вирішено використовувати трирівневу архітектуру побудови. За рахунок використання даної архітектури вдається розмежувати програмні рівні один від одного. Це є дуже важливим при подальшому супроводі і вдосконаленні системи і в рази спрощує її підтримку, спрощує пошук помилок (у випадку, коли такі були виявлені в системі).

Література

1. Пройдаков Е. М. Англо-Український тлумачний словник з обчислювальної техніки, Інтернету і програмування / Е. М. Пройдаков, Л. А. Теплицький. – Київ : СофтПрес, 2005. – 552 с. – ISBN 966-530-070-9.

2. Лавріщева К. М. Програмна інженерія / К. М. Лавріщева. – Київ, 2008. – 319 с.

3. Чорна О.В. Теорія та методика електронного навчання / О.В. Чорна, Н.А. Хараджян, С.В. Шокалюк, Н.В. Моісеєнко – Кривий Ріг : Видавничий відділ КМІ, 2013. – Том IV. – С. 272–284.

4. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSSиHTML5 / Р. Никсон. – СПб : Питер, 2016. – 816 с.

References

1. Proidakov E. M. Anhlo-Ukrainskyi tlumachnyi slovnyk z obchysliuvalnoi tekhniky, Internetu i prohramuvannia / E. M. Proidakov, L. A. Teplytskyi. – Kyiv : SoftPres, 2005. – 552 s. – ISBN 966-530-070-9.

2. Lavrishcheva K. M. Prohramna inzheneriia / K. M. Lavrishcheva. – Kyiv, 2008. – 319 s.

3. Chorna O.V. Teoriia ta metodyka elektronnoho navchannia / O.V. Chorna, N.A. Kharadzhian, S.V. Shokaliuk, N.V. Moiseienko – Kryvyi Rih : Vydavnychyi viddil KMI, 2013. – Tom IV. – S. 272–284.

4. Nikson R. Sozdaem dinamicheskie veb-sajty s pomoshyu PHP, MySQL, JavaScript, CSSiHTML5 / R. Nikson. – SPb : Piter, 2016. – 816 s.

Рецензія/Peer review : 20.06.2019 р. Надрукована/Printed : 23.07.2019 р.
Рецензент: д.т.н., проф. В.В. Мартинюк