при передачі мережевого трафіку дозволяє розвантажити магістраль і вузли мережі, розподілити навантаження між віддаленими серверами.

Розміщення серверів в безпосередній близькості від кінцевих користувачів може збільшити вихідну пропускну здатність всієї системи. Наприклад, наявність єдиного порту 100 Мбіт/с не означає дану швидкість на всіх ділянках мережі, так як вільна пропускна здатність магістрального каналу в момент передачі може бути всього 10 Мбіт/с. У випадку, коли використовуються 10 розподілених серверів, сумарна пропускна здатність може скласти 10×100 Мбіт/с.

При такому підході хмаркове сховище складатиметься з географічно розподілених багатофункціональних платформ, взаємодія яких дозволяє максимально ефективно обробляти і задовольняти запити користувачів для отримання контенту.

Проте, тут можливі дві моделі реалізації. У першому підході дані центрального сервера реплікуються на периферійні платформи. Кожна платформа підтримує в актуальному стані повну або часткову копію розповсюджуваних даних. В іншому випадку дані кешуються на сателітах і зберігаються там декілька днів.

Великі хмаркові сховища можуть складатися з величезної кількості розподілених вузлів і розміщувати свої сервери безпосередньо у мережі кожного локального інтернет--провайдера. Багато операторів роблять акцент на пропускній спроможності сполучних каналів і мінімальній кількості точок приєднання в регіоні присутності. Незалежно від використовуваної архітектури, головним призначенням подібних мереж є прискорення передачі як статичного контенту, так і безперервного потоку даних.

Для вивчення поведінки протоколів транспортного рівня широко використовується дискретно-часовий симулятор з відкритим вихідним кодом NS-3 (Network Simulator 3) [4]. Для дослідника він надає набір класів, які дозволяють моделювати протоколи і процеси, які відбуваються в комп'ютерних мережах. Також симулятор дозволяє моделювати процеси в реальному часі та інтегрувати його з дослідним стендом. Симулятор NS-3 має цілу множину готових тестів для всіх компонентів, що гарантує достовірність отриманих результатів.

Однак, нейромережевий симулятор містить тільки класи для базових протоколів передачі (TCP, UDP). При потребі досліднику потрібно на основі базових створювати власні класи протоколів, але відкритість архітектури симулятора дозволяє це зробити.

У процесі роботи над дослідженням характеристик протоколу UDT було розроблено класи для моделювання протоколу в середовищі NS-3 та проведено ряд модельних експериментів для дослідження характеристик протоколу при роботі з великими обсягами даних.

Список використаних джерел

1. User Datagram Protocol - https://tools.ietf.org/html/rfc768
2. W. Feng, P. Tinnakornsrisuphap, "The Failure of TCP in High-Performance Computational Grids". Supercomputing 2000.
3. E. He, R. Kettimut, S. Hegde, Y. Gu, M. Welzl, and W. E. Allcock, Survey of Transport Protocols Other Than Standard TCP, GGF White paper Draft, 2003.
4. The NS-3 Manual, The NS-3 Project, 2010. http://www.nsnam.org/docs/release/3.10/manual/singlehtml/index.html

УДК 004.045

ІНТЕЛЕКТУАЛЬНА СИСТЕМА ПОБУДОВИ ЗВІТІВ ДЛЯ СИСТЕМИ «HOSTEL MANAGEMENT SYSTEM»

Шпінталь М.Я.[1)], Микитюк В.П.[2)]

Тернопільський національний економічний університет

[1)] к.т.н., доцент; [2)] магістрант

І. Постановка проблеми

Для веб-орієнтованих інформаційних систем однією із функціональних особливостей є генерація різного роду звітів на основі інформації, яка зберігається в базі даних. При реалізації інтелектуальної підсистеми генерації звітної документації, як правило виникає проблема раціонального вибору

бібліотеки генерації звітів, або створення якісно нових методів їх побудови, які б дозволяли ефективніше і раціональніше використовувати різноманітні типи ресурсів.

Метою побудови звітів є надання користувачам для прийняття рішень повної, правдивої та неупередженої інформації про стан, результати діяльності. Сьогодні відомо багато засобів та програмного забезпечення для формування звітів. Більшість із цих засобів орієнтовані під конкретні системи управління базами даних, мають обмежений набір готових шаблонів звітів, надають складні механізми для побудови звітів у режимі реального часу. Основна проблема таких засобів полягає в тому, що кінцевим користувачам необхідно мати додаткові навички в галузі баз даних. Багато уваги необхідно приділяти системам, які формалізовані за допомогою великої кількості відношень, а ступінь зв'язків між ними є складним для розуміння.

Важливим аспектом життєвого циклу програмного забезпечення є ступінь адаптації до зміни апаратного забезпечення, а також до оновлення програмного забезпечення із легкою «міграцією» на інші програмно-технічні платформи. Виходячи із проведеного аналізу, актуальною науково-технічною задачею є підвищення ефективності роботи програмного забезпечення для генерації звітів з метою економії ресурсів.

II. Мета роботи

Метою є розробка універсального засобу формування звітів на основі аналізу властивостей конкретних бізнес-сутностей певної інформаційної моделі найвищого рівня, представленої у вигляді набору об'єктно-орієнтованих класів.

III. Принцип роботи системи

В основі роботи інтелектуальної системи є аналіз властивостей конкретних бізнес-сутностей інформаційної системи та надання користувачам засобів для швидкого створення звітів. Особливістю такого підходу є відсутність необхідності компетентного володіння предметною областю, а також навичок практичного використання баз даних на рівні системного інструментарію конкретної СКБД. Для аналізу властивостей бізнес сутностей необхідно здійснити їх формалізацію, виходячи із принципів об'єктно-орієнтованого проектування. Базовою категорію ООП є поняття об'єкта, як окремого екземпляра класу. Клас визначає абстрактні характеристики деякої сутності, включаючи характеристики самої сутності (її атрибути або властивості) та дії, які вона здатна виконувати (її поведінки, методи або можливості). Під властивістю (атрибутом) будемо розуміти пропозиційну функцію, визначену на довільному типі (даних). Виходячи із наведених вище припущення, можна записати наступні формалізовані предикатні відношення (1)-(4):

$$Dc = \langle Idc, CS, TCS, PDC, LDC, IdSA \rangle, \quad (1)$$

де Dc - сутність, яка інтерпретує конкретну бізнес-сутність певної інформаційної моделі; Idc - ідентифікатор сутності; CS - назва сутності; TCS - тип сутності; PDC - програмний опис сутності; PLC - лінгвістичний опис сутності, $IdSA$ - ідентифікатор предметної області.

$$A = \langle Ida, AS, TAS, PLA, LDA, Idc \rangle, \quad (2)$$

де A - атрибут, який описує властивості конкретної сутності; Ida - ідентифікатор атрибуту; AS - назва атрибуту; TAS - тип атрибута; PLA - програмний опис атрибуту; LDA - лінгвістичний опис атрибуту, Idc - ідентифікатор сутності, до якої належить визначений атрибут.

$$T = \langle IdT, NT, PT, D \rangle, \quad (3)$$

де T - відношення, яке описує певний тип даних; IdT - ідентифікатор відношення; NT - назва типу даних; PT - програмний опис типу; D - діапазон можливих значень.

Наступною сутністю, яку необхідно формалізувати є опис операцій, які визначені над конкретними об'єктами, тобто опис методів. Таке представлення опишемо за допомогою наступного відношення:

$$O = \langle Ido, NO, TO, PO, LO, A, T \rangle, \quad (4)$$

де O - метод, який описує поведінку певного атрибуту, а отже який може впливати на саму сутність; Ido - ідентифікатор методу; NO - назва методу (функції); TO - тип методу; PO - програмний опис методу; LO - лінгвістичний опис методу, A, T - множина атрибутів, та типів даних, які використовуються для реалізації відповідного методу.

Формалізовані відношення (1)-(4) дозволять здійснити інтелектуальний опис функціональності предметної області, яка реалізована в конкретній інформаційній системі. Тоді для генерації різного роду звітів необхідно також побудувати множину допустимих операції, які можна виконувати над даними формалізованими представленнями. Формалізований опис таких представлень здійснюємо за допомогою такого апарату як «алгебра кортежів». Використання такого підходу легко інтерпретується в процесі програмно-технічної реалізації системи. На основі створеного користувачем звіту та аналізу метаданих конкретних бізнес-сутностей формується дерево виразу з якого пізніше створюється SQL запит (SQL query), який готовий для виконання в СКБД.

IV. Проектування та реалізація web-орієнтованої системи

Для реалізації було обрано мову програмування C# (.NET 4.0+). Реалізація користувацької частини виконана для веб-орієнтованої системи "Hostel Management System" з використанням технології ASP.NET MVC. Після аналізу метаданих і визначення можливих операцій над типами даних користувачу доступний інтерфейс для побудови певного звіту (рисунок 1).

Рисунок 1 – Фрагмент користувацького інтерфейсу для побудови звітів

Висновок

На основі проведених досліджень, створено інтелектуальну систему формування звітів, яка базується на аналізі метаданих властивостей бізнес-сутностей найвищого рівня. Розроблену програмну систему рекомендується використовувати в різних інформаційних системах, написаних з допомогою технології ASP.NET MVC.

Список використаних джерел

1. Кузнецов С. Д. Основы баз данных / С. Д. Кузнецов – 2–е изд. – М.: Интернет – Университет Информационных Технологий; БИНОМ. Лаборатория знаний. – 2007. – 484 с.
2. Дюк, В. Data Mining: учебный курс / В. Дюк, А. Самойленко – СПб. : «Питер». – 2001. Dyuck, V. Data Mining: uchebniy kurs / V. Dyuck, A. Samoylenko – SPb. : «Piter». – 2001.
3. А. Фридмен. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов. М.: Вильямс, 2013р. – 488 с.