УДК 004.62

М.Т. Фісун, С.С. Клюс

ПОБУДОВА СИСТЕМИ ВІЗУАЛІЗАЦІЇ ДАНИХ З ВИКОРИСТАННЯМ ДОКУМЕНТО-ОРІЄНТОВАНОЇ СКБД MONGODB

Вступ. Зростання об'єму даних, орієнтованих на користувачів, викликало швидке збільшення об'єму і типів даних, які створюються, відображаються, аналізуються і архівуються. Крім того, змінюється кількість нових наборів джерел даних, у тому числі датчики, системи глобального позиціонування (GPS), автоматизовані трекери і системи моніторингу, які створюють великі масиви даних. Ці великі обсяги наборів даних, які часто називають «великими даними»(англ. Big Data), створюють нові виклики і можливості навколо зберігання, аналізу та архівування.

Паралельно зі швидким зростанням обсягів даних, дані також стають все більш частково-структурованими і розсіяними. Це означає, що традиційні методи управління даними, засновані на попередньому визначенні схеми даних і посилань між даними, вже не відповідають сучасним вимогам обробки і аналізу великих різнотипних слабко структурованих даних. Ці види СКБД називаються NoSQL [1-3]. Вони не є повним запереченням мови SQL і реляційної моделі, підхід NoSQL виходить з того, що SQL – це важливий і дуже корисний інструмент, але при цьому він не може вважатися універсальним. Однією з проблем, на яку вказують для класичних реляційних БД, є проблема при роботі з даними дуже великого об'єму, зі слабко структурованими або неструктурованими даними, особливо у системах з високим навантаженням. Основна мета підходу – розширити можливості БД там, де SQL недостатньо гнучкий, і не витісняти його там, де він справляється зі своїми завданнями дуже ефективно.

Документо-орієнтована система керування базами даних (ДОСКБД) – це система, яка зберігає колекцію текстових даних та надає змогу створювання запитів і оновлення об'єктів. Зазвичай, база даних включає в себе безліч документів, що співвідносяться за предметом, походженням або застосуванням. Зміст кожного документа може бути вільним текстом, частково структурованим текстом, який включає кілька чітко визначених атрибутів (наприклад, назва, автор, дата), або добре структурованим текстом, наприклад, може бути закодованим з використанням мови XML. Іноді документи також можуть містити мультимедійні компоненти.

Постановка задачі. ДОСКБД можуть застосовуватись для збереження великого об'єму даних: змін курсів валют, акцій та цінних паперів, трекінгу використання сайту (аналізу журналів Інтернет-серверів з великою кількістю користувачів), які можуть використовуватись для розробки системи моніторингу (візуалізації даних) реального часу. Таким чином, ДОСКБД надають нові інформаційні технології для видавництв, цифрових бібліотек, електронного уряду та електронного бізнесу в цілому. Оскільки нові технології обробки великих масивів слабко структурованої інформації ще мало досліджені, обрана тема даної статті представляється *актуальною*.

Метою дослідження є визначення переваг і недоліків документо-орієнтованої СКБД MongoDB на прикладі створення системи візуалізації даних реального часу.

Аналіз розвитку нереляційних СКБД та їх особливостей. Існує багато причин використовувати нереляційні СКБД, але можна виділити дві основні причини:

– Продуктивність розробки застосувань. Багато зусиль в області розробки програмного забезпечення витрачається на відображення моделі та структури даних, які знаходяться в оперативній пам'яті комп'ютера, у реляційні бази даних. NoSQL сховища даних можуть забезпечити модель даних, яка краще відповідатиме потребам розробників застосувань, тим самим спрощуючи взаємодію з СКБД і, в результаті чого, зменшуючи кількість написаного коду, час на налагодження і впровадження.

– Великі масштаби даних. Організації вважають доцільним зберігати великі об'єми даних і обробляти їх більш швидко. Зробити це виявляється досить дорого, якщо взагалі можливо, з використанням реляційних СКБД. Основною причиною є те, що реляційна база даних призначена для запуску на одному комп'ютері, але, як правило, є більш економним зберігати великі обсяги даних і запускати обчислення на кластерах, які складаються з набагато менших і дешевших машин. Багато нереляційних СКБД спеціально спроектовані та розроблені для роботи на кластерах, саме тому вони краще підходять для збереження великих об'ємів даних.

В літературі частіше за все виділяють такі види не реляційних СКБД:

– *Стовпчико-орієнтовані СКБД.* Структура даних в них контрастує з кортежним (рядковим) форматом реляційних СКБД. Це дозволяє уникати використання місця при зберіганні null-даних, просто не зберігаючи стовпці, якщо не існує даних для цього стовпця. Стовпці не потребують ніякого апріорного визначення. Крім того, вони здатні зберігати будь-які типи даних, оскільки дані можуть бути

збережені у вигляді масиву байтів. Така структура даних, звана як Bigtable, використовується, зокрема, компанією Google.

– *СКБД типу «ключ-значення»*. Вони є простою хеш-таблицею, тому весь доступ до даних здійснюється через первинний ключ. У якості значень даних використовуються blob-об'єкти (англ. Binary Large Objects – великі об'єкти у бінарному форматі). Оскільки СКБД типу «ключ-значення» завжди використовують первинний ключ для доступу до даних, вони, як правило, мають високу продуктивність і легко масштабуються.

– *Бази даних на основі графів.* Вони дозволяють зберігати об'єкти предметної області і зв'язки між цими об'єктами. Об'єкти також називають вузлами, які мають властивості. Вузол є представленням екземпляра об'єкта в застосуванні. Зв'язки між об'єктами, відомі як ребра, які можуть мати властивості. Ребра мають направлення; вузли організовані зв'язками, які дозволять знайти певні закономірності між вузлами. Організація графа дозволяє зберегти дані один раз і потім інтерпретувати по-різному на основі зв'язків між об'єктами. Така структура даних має багато загального з представленням знань на основаів семантичних мереж.

– *Багатовимірні бази даних.* Використовуютьтся для інтерактивного аналітичного опрацювання агрегованих даних, при цьому агрегація може бути по декількох рівнях ієрархії, наприклад, рік, квартал, місяц. Гіперкуб даних можна розглядати як множину відношень реляційної бази даних за значеннями кожного з вимірів. Отже, носієм багатовимірної моделі даних є відношення реляційної бази даних, зображені як зафіксовані виміри. Отже, між реляйною моделлю та багатомірною існує взаємне відображення, а значить, можна дані, представлені в одній моделі, перетворити в іншу модель. Тому побудовані на багатомірній моделі технології OLAP (On Line Analysis processing) [4] реалізовані в поширених СКБД реляйного типу MS SQL Server, Oracle та інших. Багатовимірна модель розширює реляційну: множина операцій, окрім традиційних реляційних, містить операції зрізу, згортки, деталізації, обертання.

– *Об'єктні СКБД.* В них дані моделюються у вигляді об'єктів, їх атрибутів, методів і класів. Одним з відомих представників цього типу систем зберігання й обробки даних є СКБД CACHÉ [5]. Крім об'єктної моделі вона підтримує також реляційну і багатовимірну, при цьому представлення даних в одній із моделей автоматично перетворюється в інші моделі.

– *документо-орієнтовані СКБД.* Документи є основною концепцією в документо-орієнтованих СКБД. База даних зберігає і вибирає документи, які можуть бути XML, JSON, BSON, і так далі. Документи – це ієрархічні структури даних, які можуть складатися з карт, колекцій та скалярних значень. Документи, що зберігаються, не обов'язково повинні бути з однаковою структурою.

Використання терміну «NoSQL», в теперішньому розумінні пов'язують з Йоханом Оскарссоном (2009 р.), а першими прообразами програмних продуктів були такі, як BigTable (пропріетарна високопродуктивна база даних, побудована на основі Google File System (GFS), Chubby Lock Service і деяких інших продуктах Google.

Компанія Google за останні кілька років, побудувала масштабну інфраструктуру для своєї пошукової системи і інших програм, включаючи Google Maps, Google Earth, Gmail, Google Finance і Google Apps. Була опублікована і представлена серія робіт, яка пояснює деякі з ключових елементів інфраструктури створених систем. Найбільш важливі з цих публікацій представлені в [6-9].

Після схвалення NoSQL двома провідними веб-гігантами – Google і Amazon – з'явилося кілька нових продуктів. Багато розробників почали використовувати методи та ідеї NoSQL в своїх прикладних програмах. Менш ніж за 5 років NoSQL і пов'язані з ним поняття для управління великим обсягами даних одержали широке поширення і використання у багатьох відомих компаніях, включаючи Facebook, Netflix, Yahoo, EBay, Hulu, IBM та ін.

В роботі [10] наведено такі три відмінності систем NoSQL від реляційних СКБД

1. *Неструктурованість.* У нереляційних базах даних, на відміну від реляційних, структура даних не регламентована (або слабо типізована, якщо проводити аналогії з мовами програмування) – в окремому рядку або документі можна додати довільне поле без попереднього декларативного змінення структури всієї таблиці.

Перевагою відсутності схеми є більш ефективна робота з розсіяними (sparse) даними. Якщо в одному документі є поле birthday, а в другому – немає, значить ніякого порожнього поля birthday для другого створено не буде. І в стовпчико-орієнтованих базах даних NoSQL, в яких використовуються знайомі поняття таблиць/колонок, в силу відсутності схеми, атрибути (колонки) не оголошуються декларативно і можуть мінятися/додаватися під час сесії роботи користува з базою. Це дозволяє зокрема використовувати динамічні колонки для реалізації списків.

У неструктурованої схеми є свої недоліки: накладні витрати в коді прикладної програми при зміні моделі даних; відсутність обмежень, що забеспечують ілісність бази даних (not null, unique, check constraint і т.д.); додаткові складнощі в розумінні і контролі структури даних при паралельній роботі з базою даних.

Проблеми паралелізму існують і в реляційних базах даних (втрата результатів оновлення, залежность від незафіксованих результатів, неузгодженона обробка результатів) [4], але у неструктурованих БД вони більш складні.

2. *Представлення даних у вигляді агрегатів.* На відміну від реляційної моделі, яка зберігає логічну модель предметної сфери в різних нормалізованих таблицях, NoSQL-сховища оперують з цими сутностями як з цілісними об'єктами. С цієї точки зору вони ближче до об'єктних баз даних. Наприклад, в об'єктній СКБД CACHÉ [5] такі агрегати називаються глобалами. Оскільки в агрегаті зберігається вся інформація для певної задачі, щоб звести до мінімуму операції «join» між об'єктами, то наслідком цього може бути дублювання інформації в інших агрегатах. В цьому полягає головне правило проектування структури даних в NoSQL базах даних – вони повинні відповідати вимогам застосування і бути максимально оптимізованими під найбільш часті запити. Наприклад, якщо платежі регулярно витягуються разом із замовленням – має сенс їх включати в загальний об'єкт, якщо ж багато запитів працюють тільки з платежами – значить, краще їх винести в окрему сутність. До речи, в реляційних базах даних теж приходиться йти на денормалізацію БД під найбільш часті запити [11]. Це компроміс, на який доводиться йти, щоб запобігти значного уповільнення роботи бази. Плюси і мінуси обох підходів згруповані у таблиці 1.

Таблиця 1.

Переваги і недоліки різних підходів зберігання даних

Структури й організація даних	Переваги	Недоліки
Нормалізовані таблиці реляційної моделі даних	Цілісність інформації при оновленні (міняємо запис в одній таблиці, а не у декількох).	Неефективна у розподіленому середовищі.
	Орієнтованість на широкий спектр запитів до даних	Низька швидкість читання при використанні сполучень (joins)
	Математичний базис мов маніпулювання даними	Невідповідність реляційної моделі фізичній структурі даних
Дані у вигляді агрегатів в NoSQL-базах даних	Найкращий спосіб досягти високої швидкості читання у розподіленому середовищі.	Оптимізація тільки під певний тип запитів.
	Можливість зберігати фізичні об'єкти у тому вигляді, в якому з ними легше працювати застосування(легше кодувати і менше помилок при перетворенні)	Труднощі щодо зебеспечення цілісності ненормалізованих даних
	Вбудована підтримка атомарності на рівні записів	Відсутність математичного базису для мови маніпулювання даними

3. *Слабкі ACID-властивості.* ACID (англ. Atomicity, Consistency, Isolation, Durability) – це набір властивостей, що гарантують надійну роботу транзакцій бази даних: атомарність, узгодженість, ізольованість, довговічність [4]. Узгодженість даних була найважливішою з точки зору архітекторів і розробників баз даних. Всі реляційні бази забезпечували той чи інший рівень ізоляції – або за рахунок блокувань при зміні і блокуючого читання, або за рахунок undo-логів. З появою величезних масивів інформації і розподілених систем стало ясно, що забезпечення для них транзакційного набору операцій з одного боку і отримання високої доступності і швидкого часу відгуку з іншого – неможливо.

Виходячи з вищенаведеного, виконано аналіз існуючих документо-орієнтованих (ДО) СКБД та вибір базової для системи візуалізації даних. На сьогоднішній день існує досить багато ДОСКБД. Вони випускаються під різними ліцензіями та реалізовані для використання на різних платформах, хоча деякі можуть використовуватись на декількох платформах одночасно. Незважаючі на деякі відмінності, слід зазначити, що кожна реалізація документної бази даних базується на основному понятті – «документ», яка описує один запис у базі даних. При цьому допускаються різні способи організації зберігання списків документів.

Для системи візуалізації даних важливими при виборі документо-орієнтованої СКБД є наявність REST- та HTTP-API – які дозволяють взаємодіяти з базою даних через протокол передачі даних HTTP, що досить зручно і дозволяє робити запити та отримувати дані, використовуючи HTTP запити: HTTP GET, POST, UPDATE та DELETE.

Досить часто при подальшому розвитку програмного продукту можливо створення самостійного

застосування, тому важливим є клієнтський API, який дозволить взаємодіяти з базою даних на більш високому рівні абстракції.

Незважаючи не те, що кожна реалізації документо-орієнтованої бази даних відрізняється у деталях, загалом, всі вони припускають, що документи інкапсулюють і кодують дані (інформацію) у деяких стандартних форматах. Формати кодування включють XML, YAML, JSON(JavaScript Object Notation), і BSON, а також бінарний формат, такий як PDF і документи Microsoft Office (MS Word, Excel і т. д.) [10].

Наприклад, документ може виглядати так:

```
{
    FirstName:»Romanov»,
    Address:»15 Okt Pr.»,
    Hobby:»sailing»
}
Інший документ може бути:
{
     FirstName:»Bulba»,
     Address:»6 Krylova Str.»,
     Children:[
        {Name:»Leonid»,Age:12},
        {Name:»Eugen», Age:10},
        {Name:»Tamara», Age:7},
        {Name:»Elena», Age:3}
     ]
}
```

Важливою характеристикою для обрання конкретної ДОСКБД є можливість горизонтальної масштабованості, тобто підключення великої кількості клієнтів.

Неодмінно треба звернути увагу на ціну та ліцензію документо-орієнтованої СКБД. Оскільки немає сенсу купувати дорогу систему для малого проекту, який не буде користуватися привілеями платних рішень.

Важливим фактором є наявність документації. За наявності вичерпної документації досить легко розпочати використання системи та вивчати особливості роботи з СКБД.

На додаток до вище перерахованих властивостей слід додати: наявність гнучкої мови для формування запитів, наявність динамічних запитів, наявність візуальних застосувань керування базою даних, швидке додавання нових документів та швідка вибірка незалежно від масштабу.

Необхідно також враховувати кількість активних користувачів, форумів, блогів та інших Internet-ресурсів обраної СКБД, оскільки за наявності чисельної спільноти легше знайти відповіді на виникаючі у процесі роботи з системою питання.

У таблиці 2 наведено порівняння характеристик найбільш поширених документо-орієнтованих СКБД.

Для реалізації системи візуалізації даних було обрано СКБД MongoDB з відкритим вихідним кодом, яка не потребує опису схеми таблиць. MongoDB займає нішу між швидкими і масштабованими системами, що оперують даними у форматі ключ/значення, і реляційними СКБД, функціональними і зручними у формуванні запитів. Код MongoDB написаний мовою C++ і поширюється в рамках ліцензії AGPLv3. GNU Affero General Public License (загальна суспільна ліцензія Affero GNU) або ‘GNU AGPL’ — вільна ліцензія на програмне забезпечення, видана Фондом Вільних Програм, та орієнтована на серверні веб-програми та веб-застосування. GNU AGPL подібна до GNU GPL за винятком того, що має додаткову секцію, яка надає право отримати сирцевий код зміненої програми користувачам, які використовують її через мережу.

У рамках даної роботи виконана розробка системи візуалізації даних реального часу. Головною задачею системи буде обробка подій, прив’язаних до дати та часу. Система буде відповідати за збереження їх у документо-орієнтовану базу даних MongoDB. На рис. 1 представлено функціональну модель системи та декомпозицію її окремих модулів (рис. 2 і 3) за стандартом IDEF0 [12].

В системі реалізовано можливість приймати події у стандартизованому форматі BSON та зберігати їх у колекціях СКБД MongoDB, визначаючи необхідні колекції, спираючись на тип події. Якщо колекція для визначеного типу не існує, необхідно створити її та колекцію для розрахованих метрик за типом події.

Метрики у даному контексті є одним із способів обчислення властивостей системи, базуючись на збережених подіях, що представляють інтерес. Згрупувавши і звівши їх до одного значення, можна отримати певні характеристики системи. Наприклад, якщо існує набір подій «request», можна обчислити число запитів в кожній п’ятихвилинний інтервал часу, протягом останнього тижня. Цей показник, «sum(request)» свідчить про інтенсивність запитів з плином часу.

Таблиця 2.

Порівняння характеристик трьох ДОСКБД

СКБД NoSQL / Критерій	CouchDB	MongoDB	OrientDB
REST-API HTTP-API	Присутні	Присутні	HTTP
Клієнтський API	JavaScript, PHP, Ruby, Python і Erlang	Java, JavaScript, C++, C#, PHP, Python, Perl, Ruby	Java
Ціна	Безкоштовна	Безкоштовна	Безкоштовна
Популярність	Висока	Дуже висока	Середня
Гнучка мова запитів	Запити на мовах JavaScript та Erlang	JavaScript та вбудована гнучка мова запитів	Підтримує мову запитів SQL та Gremlin
Динамічні запити	Ні	Так	Так
Підтримка ACIDвластивостей	Гарантує, при збереженні документів через механізм MVCC, як PostgreSQL	Оновлює документі на місці без реальної підтримки паралелізму, що надає додаткову швидкість запису	Повна підтримка ACID-властивостей
Рівень горизонтальної масштабованості	Високий	Дуже високий завдяки відсутності паралелізму	Низький

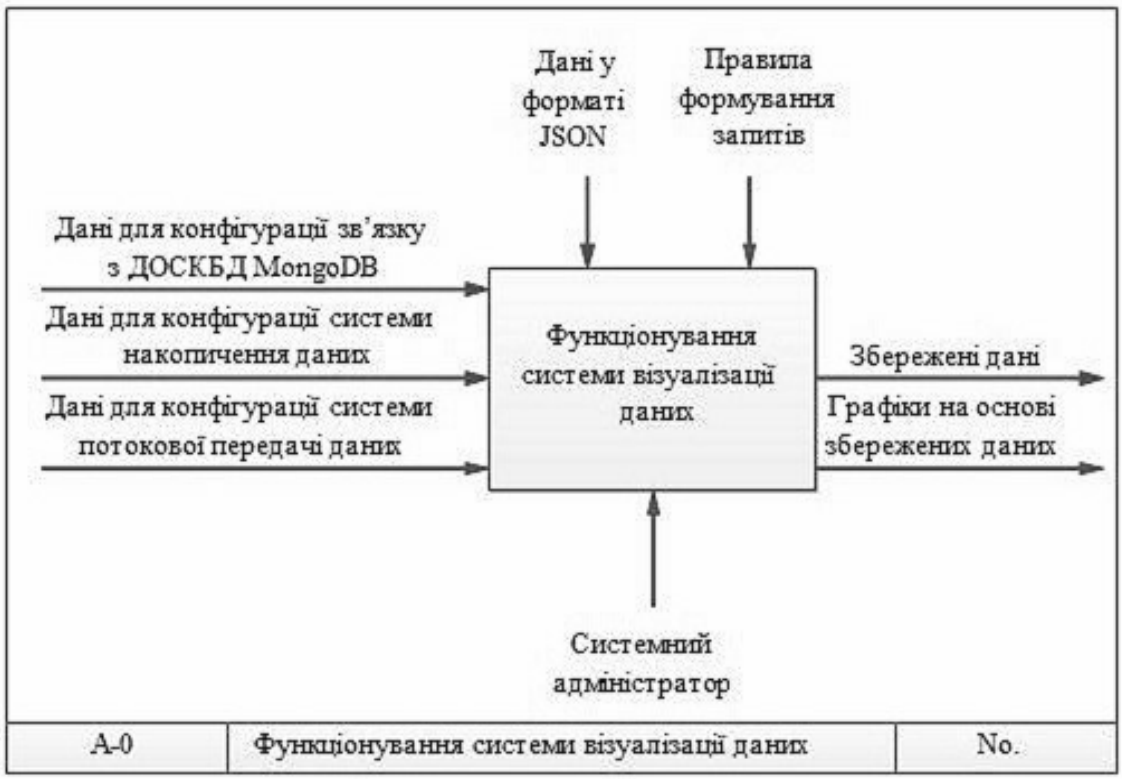

Рис. 1. Функціональна модель системи

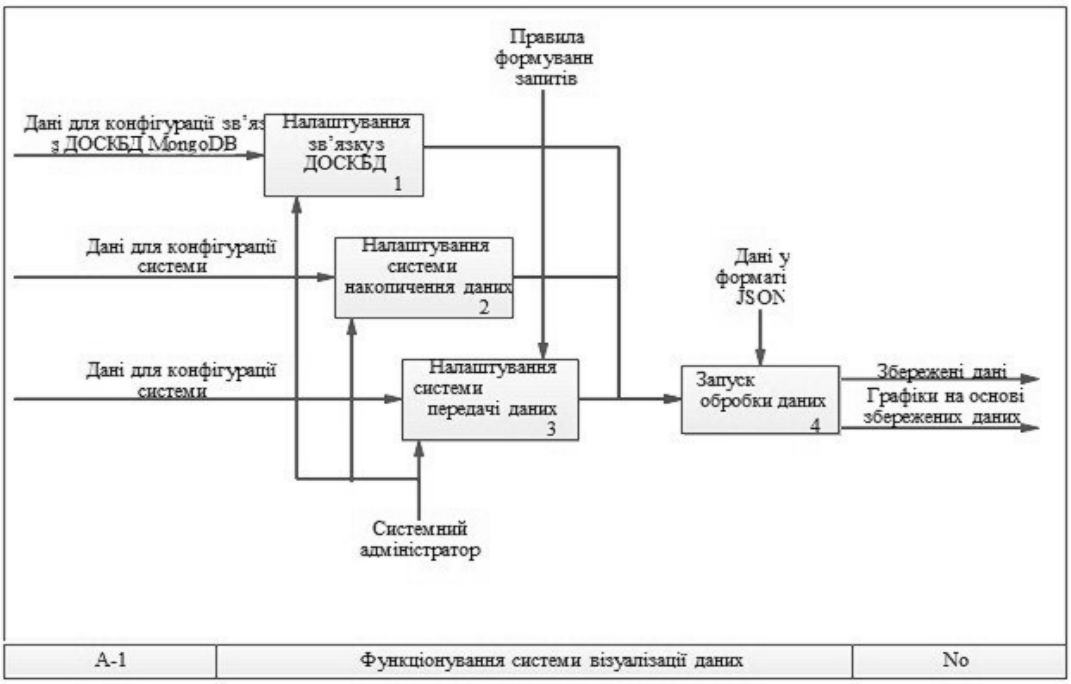

Рис. 2. Декомпозиція функціональної моделі системи на першому рівні

Розглянемо більш детально та проведемо декомпозицію процесу запуску серверу обробки даних на рис. 3.

Рис. 3Декомпозиція процесу запуску серверу обробки даних

На рис. 4 наведено діаграму класів [13] програмного забезпечення системи візуалізації.

Головним способом взаємодії із застосуваннями є протокол передачі даних HTTP.

В системі візуалізації є можливість робити запити подій і метрик, через HTTP GET запит або за допомогою WebSockets. Однією з головних вимог є відображення даних, отриманих за допомогою запитів подій та метрик, у реальному часі.

Для реалізації системи обрано мову програмування JavaScript та серверну реалізацію мови Node.js, оскільки вона добре інтегрується з MongoDB та може використовуватись у запитах.

Node.js – платформа з відкритим вихідним кодом для програмування серверної частини застосувань на мові JavaScript [14].

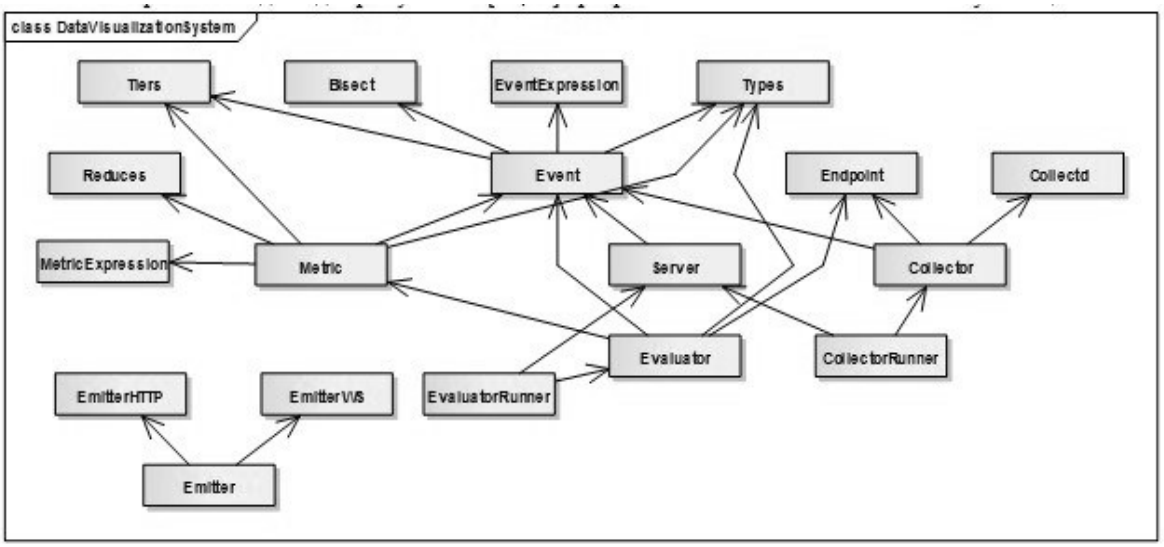

Рис. 4. Діаграма класів розробленої системи

Для потокової передачі даних була обрана технологія WebSockets [15], оскільки цю технологію досить легко використовувати у прикладних програмах, застосовуючи мову JavaScript.

Однією з головних вимог до розроблюваної системи є відображення даних, отриманих за допомогою запитів подій та метрик, у реальному часі.

Основні результати і висновки. Практичним результатом виконаної роботи є створена система візуалізації даних реального часу, призначена для консолідації й обробки інформації про події у високонавантажених веб-системах. Система використовує документо-орієнтовану СКБД MongoDB для зберігання отриманих даних та JavaScript для обробки подій на стороні сервера, який запускається із застосуванням платформи Node.js.

Як приклад повсякденного застосування системи візуалізації реального часу можна привести можливість відправки зрізів даних про обсяги і специфіку трафіку кожні 5 хвилин з подальшим аналізом всіх накопичених даних в наочному вигляді.

Інший типовий варіант використання – протоколювання волатильності курсу певної валюти в режимі реального часу. При цьому слід мати на увазі, що моделі числових послідовностей дозволяють не тільки здійснювати моніторинг хронології поведінки певного показника, але також передбачати його імовірнісну поведінку в майбутньому на підставі вже накопиченої інформації та автоматичного виявлення трендів. Перевага системи складається в формуванні візуальної загальної картини роботи системи в режимі реального часу.

Нажаль, чорно-білий формат представлення графічного матеріалу не дозволяє наглядно представити результати приклади роботи системи візуалізаії даних.

ЛИТЕРАТУРА

1. Shashank Tiwari. Professional NoSQL. Indianapolis. John Wiley & Sons, Inc. – 2011. – 480 p.
2. Eelco Plugge, Peter Membrey and Tim Hawkins. The Definitive Guide to MongoDB. The NoSQL Database for Cloud and Desktop Computing. SpringerVerlag, New York. Apress, Inc. – 2010. – 328 p. – ISBN: 978-1-4302-3051-9.
3. Document-oriented database. Режим доступу: URL: http://en.wikipedia.org/wiki/Document-oriented_database. – Загол. з екрану.
4. Дейт К. Дж. Введение в системы баз данных, 7-е издание. : Пер. с англ. – М: Издательство «Вильямс», 2001. – 1072 с.
5. Кречетов Н.Е., Петухова Е.А., Скворцов В.И., Умников А.В., Щукин Б.А. Постреляционная технология CACHÉ для реализации объектных приложений. Учебное пособие. М.:МИФИ, 2001. – 152 с.
6. Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. “The Google File System”; pub. 19th ACM Symposium on Operating Systems Principles, Lake George, NY, October 2003. URL: http://labs.google.com/papers/gfs.html.
7. Jeffrey Dean and Sanjay Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters”; pub. OSDI’04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December 2004. URL: http://labs.google.com/papers/mapreduce.html.
8. Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar

Chandra, Andrew Fikes, and Robert E. Gruber. “Bigtable: A Distributed Storage System for Structured Data”; pub. OSDI’06: Seventh Symposium on Operating System Design and Implementation, Seattle, WA, November 2006. URL: http://labs.google.com/papers/bigtable.html.
9. Mike Burrows. “The Chubby Lock Service for Loosely-Coupled Distributed Systems”; pub. OSDI’06: Seventh Symposium on Operating System Design and Implementation, Seattle, WA, November 2006. URL: http://labs.google.com/papers/chubby.html.
10. NoSQL базы данных: понимаем суть: URL: http://habrahabr.ru/post/152477/. – Загол. з екрану.
11. Кунгурцев А.Б., Зиноватная С.Л. Изменение структуры реляционной базы данных для применения денормализации отношений // Вестник Херсонского национального технического университета. – Херсон, 2007. – №4(27). – С. 293 – 296.
12. Применение CASE–средства BPwin 4.0 для информационного моделирования в системах обработки данных [Текст] / Горин С.В. ; Тандоев А.Ю // СУБД /. – М., 1995. – №3. – С. 125–127.
13. Крэг Ларман. Применение UML 2.0 и шаблонов проектирования [Електронний ресурс] – 3–е изд. – М.: Вильямс, 2006. – 736 с. – Режим доступа: URL : http://www.williamspublishing.com/Books/978–5–8459–1185–8.html – Загол. з екрану.
14. Pedro Teixeira. Professional Node.js: Building Javascript Based Scalable Software. Indianapolis, IN. John Wiley & Sons, Inc. – 2012. – 412 p.
15. WebSocket. Режим доступу: URL: http://ru.wikipedia.org/wiki/WebSocket. – Загол. з екрану.

ФІСУН М.Т. – д.т.н., проф., завідувач кафедри інтелектуальних інформаційних систем, Чорноморський державний університет ім. П. Могили.

Наукові інтереси: бази даних, інтелектуальні інформаційні системи, реінжинірінг, комп’ютерна лінгвістика.

КЛЮС С.С. – студент магістратури кафедри інтелектуальних інформаційних систем, Чорноморський державний університет ім. П. Могили

Наукові інтереси: бази даних, інтелектуальні інформаційні системи.