Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Комп'ютерних наук
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)
магістр
(освітній ступінь (освітньо-кваліфікаційний рівень))

на тему: **Методи та засоби синхронізації баз даних із врахуванням цінності інформації**

Виконав: студент (ка) 6 курсу, групи САм-61
спеціальності (напряму підготовки) 124 «Системний аналіз»
(шифр і назва спеціальності (напряму підготовки))

	(підпис)	Постолюк А. М.. (прізвище та ініціали)
Керівник	(підпис)	Гром'як Р. С. (прізвище та ініціали)
Нормоконтроль	(підпис)	Мацюк О. В. (прізвище та ініціали)
Рецензент	(підпис)	Тиш Є. В. (прізвище та ініціали)

м. Тернопіль – 2019

АНОТАЦІЯ

Методи та засоби синхронізації баз даних із врахуванням цінності інформації // Дипломна робота освітнього рівня "Магістр" // Постолюк Андрій Миколайович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група САм-61 // Тернопіль, 2019 // С.131, рис. – 26, табл. – 5, додат. – 4 , бібліогр. джерел – 50.

Методи дослідження базуються на принципах системного аналізу, узагальнення, статистичній обробки даних, теорії ефективності функціонування баз даних, теорії ймовірностей, методах математичного моделювання.

Запропоновано метод визначення цінності інформації з використанням функції чутливості, який використано в системі синхронізації розподілених баз даних.

Об'єктом дослідження є синхронізація розподілених бази даних.

Предметом дослідження є методи та комп’ютерні засоби, що визначають процеси синхронізації баз даних із врахуванням якісно-кількісних характеристик.

Практичне значення одержаних результатів. Розроблено програмну систему для синхронізації баз даних користувачів мультиплатформеного веб-сайту.

Ключові слова: розподілена база даних, реплікація, синхронізація, якісно-кількісні характеристики.

ANNOTATION

Methods and means of synchronization of databases with regard to the value of information // Diploma work of educational level "Master" // Postolyuk Andriy Nikolaevich // Ternopil National Technical University named after Ivan Pulyuy, Faculty of Computer-Information Systems and Software Engineering, Department of Computer Science , Sam-61 group // Ternopil, 2019 // C.¬131¬¬¬, fig. - 26, tab. - 5, add. - 4, bibliography. sources - 50.

Research methods are based on the principles of system analysis, generalization, statistical processing of data, theory of efficiency of functioning of databases, theory of probabilities, methods of mathematical modeling.

A method of determining the value of information using the sensitivity function, which is used in the system of synchronization of distributed databases, is proposed.

The object of the study is synchronization of distributed databases.

The subject of the study is methods and computer tools that determine the processes of database synchronization, taking into account qualitative and quantitative characteristics.

The practical significance of the results obtained. A software system has been developed to synchronize the databases of users of the multi-platform website.

Keywords: distributed database, replication, synchronization, qualitative and quantitative characteristics.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

РБД – розподілена база даних

СУБД – система управління базами даних

SQL – Structured query language — мова структурованих запитів

UDP – User Datagram Protocol, протокол дейтаграм користувача

ОС – операційна система

ЗМІСТ

ВСТУП

Передача даних між вузлами розподіленої системи постає перед торговельними мережами практично щодня. Зокрема, таке завдання може зустрітися при передачі даних про наявність залишків товарів на відповідних торговельних точках, відмітки про доставку товарів, їх затримку або у разі, коли управляючий відділ веде набір довідників, кожен з яких обов'язково має бути присутнім в базі даних окремих торговельних точок, що звітують перед ним. І, хоча вже сформувалися найбільш часто використовувані підходи до реалізації алгоритмів реплікації, всі вони не позбавлені деяких недоліків – а саме, жоден з існуючих методів реплікації не враховує якісно-кількісні характеристики інформації, насамперед цінність, ціну та старіння інформації.

Система реплікації розподіленої баз даних повинна відповідати наступним вимогам: мінімізація впливу на вже існуючі програмні комплекси і бази даних; будь-які зміни в базі даних мають виявлятися; система має бути ефективною, такою, що масштабується і мати прогнозовані витрати при розширенні; записи, що зберігаються, не повинні дублюватися частково або повністю; система не повинна зберігати довгі ланцюжки змін в базі; вона має бути ефективною навіть на дуже великій кількості записів; система має бути достатньо автономна; система має однаково добре функціонувати на абсолютно різних платформах; рахування якісно-кількісних характеристик інформації, що передається, для побудови оптимальної черги транзакцій та підвищеня ефективності функціонування системи реплікацій.

Відзначимо, що в реальному житті цієї вимоги досягти достатньо складно, оскільки окрім специфіки реалізації внутрішньої структури існує ще специфіка мови доступу до бази даних. І, хоча теоретично всі SQL-сумісні бази даних повинні відповідати деякому єдиному стандарту, на практиці це далеко не так.

Організація синхронізації даних таким чином є досить складним завданням. Хоча подібні проблеми виникають досить часто, універсального рішення цієї задачі в даний час практично не існує. Майже всі готові реплікатори накладають істотні обмеження на структуру і способи накопичення і зміни даних в таблицях бази даних. У кожній СУБД пропонується, як правило, декілька рішень задачі синхронізації даних. Це пов'язано з різноманіттям цілей використання СУБД при розробці розподіленої БД.

Урахування якісно-кількісних характеристик при проведенні реплікації розподілених баз даних, є досить актуальною задачею та розробляється на підставі виробничої необхідності, що виникла у роботі торговельних мереж.

Метою досліджень є розробка методів організації реплікацій в розподілених базах даних з урахуванням якісно-кількісних характеристик інформації, що передається.

Об'єкт дослідження – синхронізація розподілених бази даних.

Предмет дослідження – методи та комп'ютерні засоби, що визначають процеси синхронізації баз даних із врахуванням якісно-кількісних характеристик.

Методи дослідження: базуються на принципах системного аналізу, узагальнення, статистичній обробки даних, теорії ефективності функціонування баз даних, теорії ймовірностей, методах математичного моделювання.

Наукова новизна одержаних результатів.

Запропоновано метод визначення цінності інформації з використанням функції чутливості, який використано в системі синхронізації розподілених баз даних.

Практичне значення одержаних результатів.

Розроблено програмну систему для синхронізації баз даних користувачів мультиплатформеного веб-сайту.

РОЗДІЛ 1
ТЕХНОЛОГІЇ СИНХРОНІЗАЦІЇ ТА РЕПЛІКАЦІЇ РОЗПОДІЛЕНИХ БАЗ ДАНИХ

1.1 Особливості організації розподілених баз даних

Під розподіленою базою даних (Distributed DataBase – DDB) звичайно мають на увазі базу даних, що включає фрагменти з декількох баз даних, які розташовуються на різних вузлах мережі комп'ютерів, і, можливо, управляються різними системами керування базами даних. В ідеальній ситуації розподілена база даних виглядає з погляду користувачів і прикладних програм як звичайна локальна база даних. У цьому змісті слово "розподілена" відбиває спосіб організації бази даних, але не її зовнішню характеристику ("розподіленість" бази даних невидима ззовні) [1].

Розподілена база даних передбачує зберігання й виконання функцій керування даними в декількох вузлах і передачу даних між цими вузлами в процесі виконання запитів. Розбивка даних у розподіленій базі даних може досягатися шляхом зберігання різних таблиць на різних комп'ютерах або навіть зберіганні різних частин і фрагментів однієї таблиці на різних комп'ютерах. Для користувача або прикладної програми не має значення, яким образом розподілені дані між комп'ютерами. Робота з розподіленою базою даних здійснюється так само, як і із централізованої, тобто розміщення бази даних повинне бути прозорим [2].

На рисунку 1.1 наведений приклад розподіленої бази даних. При розподіленій обробці робота з базою (подання даних, їхня обробка та інше) ведеться на комп'ютері клієнта, а підтримка бази в актуальному стані – на сервері. При цьому такі бази даних звичайно розташовуються на декількох серверах – різних вузлах комп'ютерної мережі, а деякі дані можуть дублюватися.

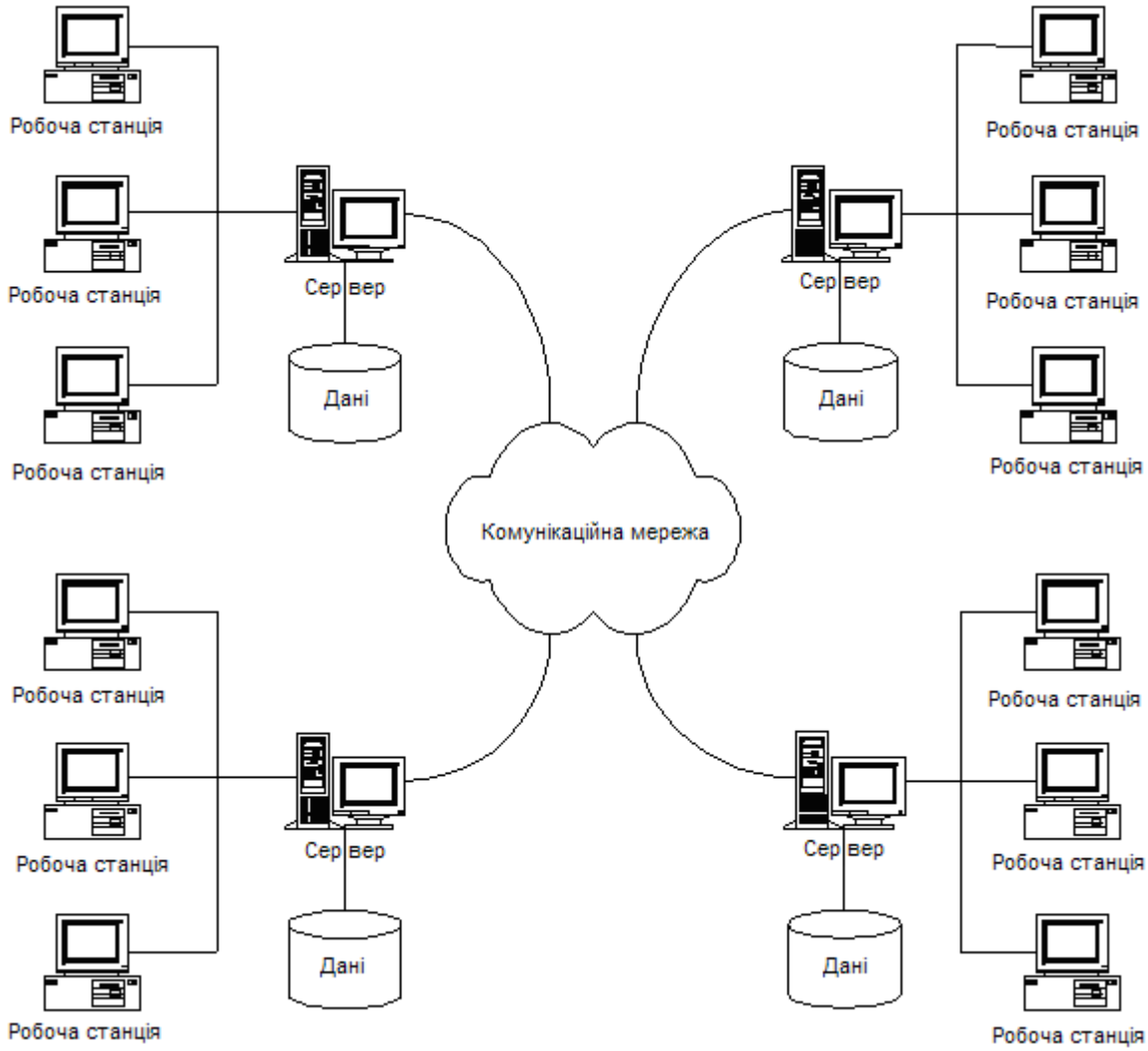


Рисунок 1.1. Типова розподілена база даних [2]

Створення розподілених баз даних викликано спробою одночасного рішення двох завдань: інтеграції й децентралізації. Інтеграція має на увазі централізоване керування й ведення баз даних. Децентралізація забезпечує зберігання даних там, де вони з'явилися й обробляються. При цьому знижується вартість системи й збільшується ступінь її надійності, а також підвищується швидкість обробки даних.

Звичайний сервер зберігає в себе всі дані й обслуговує всі клієнтські запити. Схема взаємодії між сервером і клієнтами зображена на рисунку 1.2. Серверні дані розташовуються на одному або декількох фізичних дисках. Щоб зробити запит, клієнт установлює з’єднання із сервером. Сервер аналізує інструкції, виконує їх, отримує дані й повертає результати запиту.

У міру зростання навантаження продуктивність сервера знижується. Щоб уникнути цього, використовують додаткові ресурси, наприклад, нарощують пам'ять, ставлять додаткові процесори й навіть мережні плати. Це ефективна стратегія, якщо клієнти розташовані в безпосередній близькості від сервера, наприклад, кілька серверів додатків взаємодіють із однією системою керування базами даних. Але в тих архітектурах, де сервер і клієнти віддалені один від одного, продуктивність обернено пропорційна відстані.

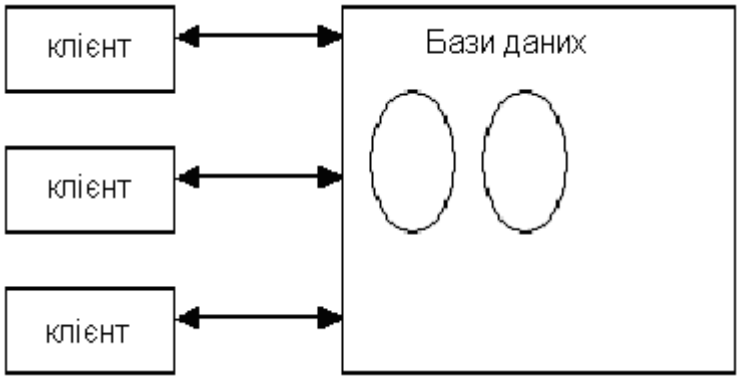


Рисунок 1.2. Схема взаємодії сервера бази даних з клієнтами

Рішенням цієї проблеми є розподілені бази даних (РБД), які сегментують збережену інформацію й переміщують окремі її блоки ближче до потрібних клієнтів. Способів організації таких баз даних багато. Можна розмістити таблиці на різних комп'ютерах або використовувати кілька ідентичних сховищ. У другому випадку сервери взаємодіють один з одним для підтримки синхронізації. Якщо на одному із серверів відбувається відновлення даних, воно поширюється й на всі інші сервери.

До недоліків розподілених баз даних можна віднести те, що зростає складність керування ними. Але все ж таки переваг більше. Головне з них – підвищення продуктивності. Дані швидше обробляються декількома серверами, крім того, дані розташовуються ближче до тих користувачів, які частіше з ними працюють [4].

Система стає більше стійкої, якщо вона здатна витримати збій одного зі своїх компонентів. У розподіленій базі даних із симетричною схемою зберігання зникнення одного із серверів приводить до вповільнення роботи користувачів, що перебувають ближче до цього сервера, але в цілому система залишається працездатної. До того ж, вона легко масштабується, тому що її не потрібно зупиняти при додаванні ще одного сервера.

У несиметричній системі можна оптимізувати схему розташування даних. Чим ближче користувачі перебувають до потрібних їм даних, тим менше на їхню роботу впливають мережні затримки. У результаті серверам доводиться обробляти менші обсяги даних. Це також сприяє підвищенню безпеки даних, оскільки їх можна фізично зберігати в тих системах, де користувачі мають право працювати з відповідними даними. У цілому, однак, застосування розподілених баз даних пов'язане з досить високим ризиком. Потрібно забезпечити дотримання мір безпеки відразу на декількох вузлах, що не так просто реалізувати.

Розподілені бази даних важко проектувати й обслуговувати. Порядок роботи в системі може згодом помінятися, що спричинить зміну схеми зберігання даних. Як клієнти, так і сервери повинні вміти обробляти запити до даних, які не розташовані в найближчій системі. Погано спроектована розподілена база даних може демонструвати меншу продуктивність, ніж одиночний сервер.

Розподілена база даних складається із трьох основних частин: клієнтів, модуля обробки транзакцій і сховища даних. Сервер звичайно бере на себе завдання обробки транзакцій і зберігання даних, хоча у повністю розподіленій базі даних за рішення цих завдань відповідають різні апаратні компоненти. Загальну схему цього процесу представлено на рисунку 1.3. Тут три клієнти взаємодіють із двома модулями обробки транзакцій, які, у свою чергу, працюють із двома сховищами. Клієнти посилають свої запити модулям, а ті визначають, у якому зі сховищ перебувають необхідні дані.

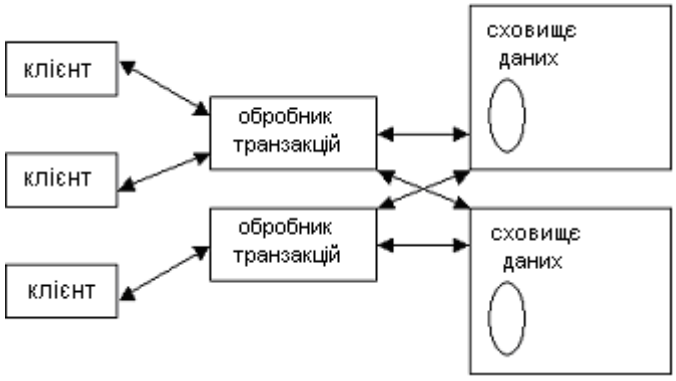


Рисунок 1.3. Організація роботи розподілених серверів [6]

В ідеалі клієнти не знають, є система розподіленою чи ні. Вони лише посилають їй запити, а система повертає клієнтам результати цих запитів. Як вона це робить, клієнтів не цікавить. На практиці розподілені бази даних демонструють різну "прозорість". У крайньому випадку розподілена база даних зберігається на декількох незалежних серверах, а клієнтському додатку доводиться обирати сервер залежно від того, яку інформацію потрібно одержати. Це означає, що таблиці, які перебувають на різних серверах, не мають ніяких внутрішніх зв'язків. Тому така організація розподіленої бази даних лише зрідка виявляється корисною.

Система керування розподіленими базами даних надає клієнтам уніфікований інтерфейс доступу до даних, завдяки якому виникає ілюзія єдиного сервера.

1.2 Способи виявлення змін

Алгоритмічне виявлення змін. Щоб система реплікації працювала коректно, необхідний коректно працюючий алгоритм визначення даних, які необхідно передати між двома вузлами. Цей алгоритм є однією з ключових характеристик систем реплікації. Розглянемо різні алгоритми виявлення (або захоплення) змін, що мають місце у системах реплікації. Тут розглядаються алгоритми, які в результаті своєї роботи гарантовано синхронізують дві бази

даних (за умови штатної роботи системи і відсутності змін під час сеансу обміну даними).

Повна передача таблиць є виродженим випадком виявлення змін, коли реально не відбувається ніякого захоплення змін, а таблиця передається цілком вже під час сеансу реплікації. Гіпотетично, такий підхід може бути застосовний тільки для випадків, коли вузли зв'язані між собою дуже швидким каналом передачі даних. На практиці він практично не використовується із-за надто неефективного використання ресурсів. Повна передача всієї таблиці має сенс тільки на початковому етапі взаємодії між вузлами, тобто на етапі створення видаленої копії, що зображено на рисунку 1.4.

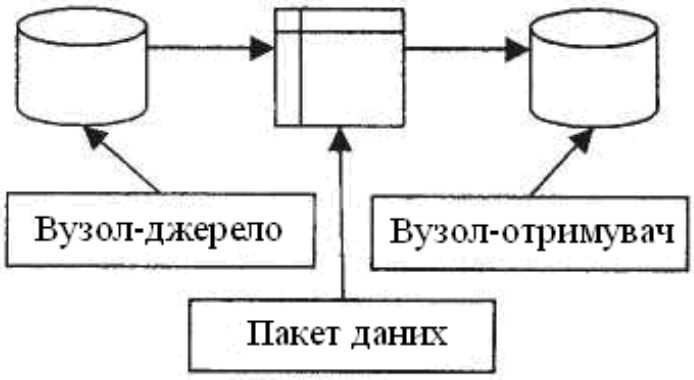


Рисунок 1.4. Метод повної передачі таблиць

Переваги:

1. Простота технічної реалізації є основною перевагою.

Недоліки:

1. Великі витрати ресурсів, які практично не дозволяють використовувати цей метод на практиці.

2.Неможливість проведення аналізу кількісно-якісних характеристик передаваємої інформації

Виявлення змін за допомогою подвійного копіювання таблиць працює так, що після проведення кожного сеансу реплікації для кожної робочої таблиці

створюється її копія (або у тій же базі, або взагалі на іншому носієві). Ця копія є незмінною до проведення наступної реплікації. Отже, всі зміни, занесені до робочих таблиць, доцільно порівняти з початковою таблицею а також з її копією. Приклад роботи цього алгоритму поданий на рисунку 1.5.

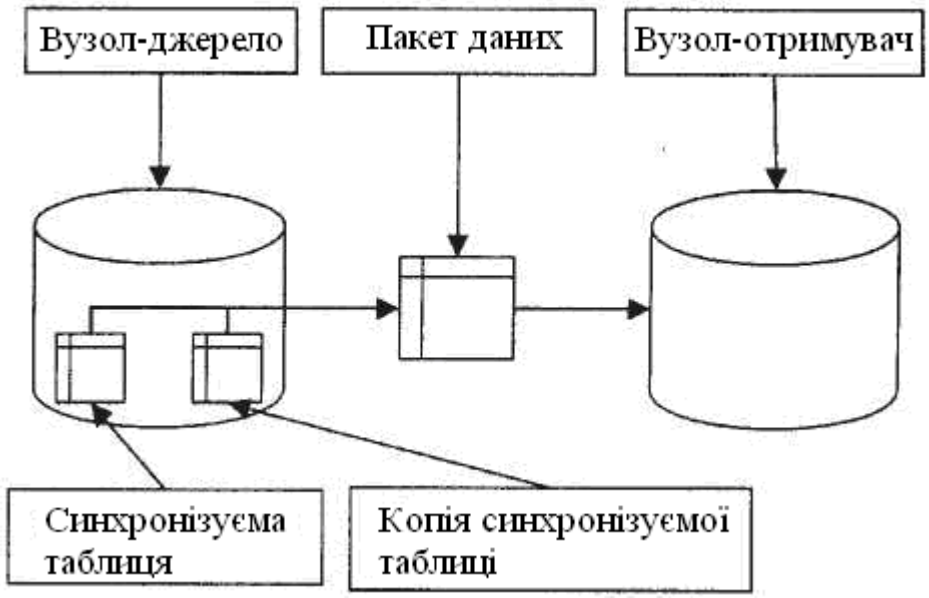


Рисунок1.5. Метод подвійного копіювання таблиць

Переваги:

1. Не вимагається зміни існуючого програмного коду і структури бази даних.

2. За допомогою такого підходу будуть виявлені всі зміни, незалежно від того, якими засобами вони були внесені.

Недоліки:

1. Для кожної відстежуваної таблиці зберігається її повна копія, що в загальному випадку, подвоює об'єм даних.

2. В разі збою на вузлі-джерелі, відновлення повинно відбуватися по не цілком виправданому алгоритму — повна пересилка таблиці з подальшим аналізом даних, які дійсно вимагають оновлення.

3. Не враховується якісно-кількісні характеристики передачі для проведення оптимальної реплікації.

Проміжний рівень доступу до бази даних є додатковим програмним прошарком між рівнем логіки управління і рівнем доступу до бази даних. Таким чином, разом із записом змін у відстежувану таблицю, цей проміжний рівень здійснює запис і в чергу змін. На рисунку 1.6 показана одна з можливих реалізацій такого методу.

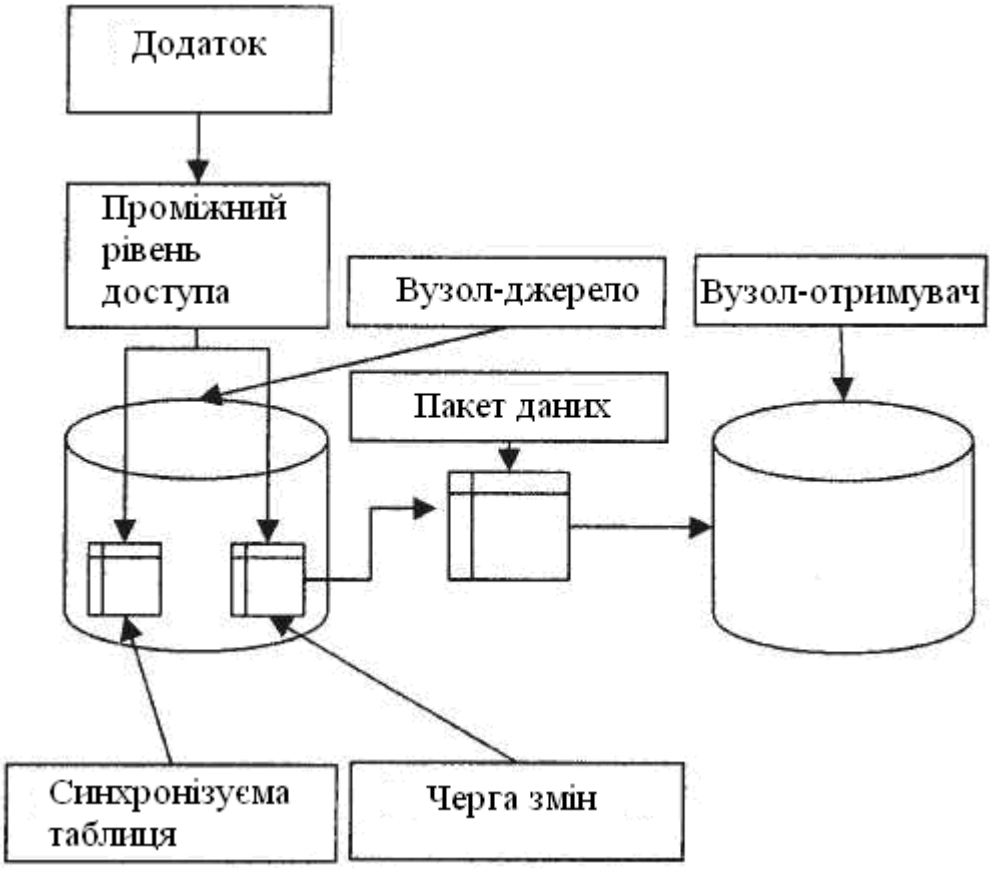


Рисунок1.6. Принцип роботи системи з проміжним рівнем доступу до бази даних

Переваги:

1. Не вимагається внесення змін до існуючої схеми бази даних, а такі умови часто ставляться перед розробниками.

2. Має технологічну можливість формування черги змін із урахуванням якісно-кількісних характеристик інформації.

Недоліки:

1. Можуть потрібно значні витрати при впровадженні проміжного рівня

доступу до бази даних, якщо навіть початкове застосування було правильно спроектоване.

2. Не відстежуються зміни, вироблювані програмними засобами, що не використовують цей проміжний рівень доступу. Необхідність в цьому виникає, коли, наприклад, довідкова інформація міняється як з основного застосування, так і з безлічі допоміжних програмних засобів.

Виявлення змін за допомогою журналу транзакцій. Система реплікації відстежує зміни в журналі транзакцій бази даних і або формує нову чергу змін, або використовує сам журнал транзакцій для формування списку змін. На рисунку 1.7 приведена схема роботи цього алгоритму.

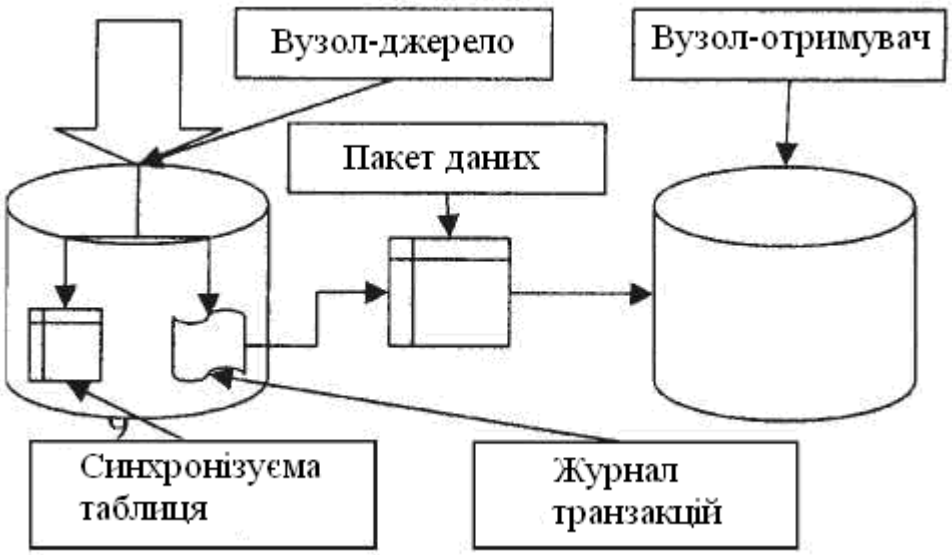


Рисунок 1.7. Виявлення змін за допомогою журналу транзакцій

Переваги:

1. Не вимагається внесення змін до існуючої схеми бази даних.

2. Відсутня залежність від програмного засобу, використаного для внесення змін.

Недоліки:

1. Різна архітектура журналу транзакцій. Архітектура цього журналу різна не лише у СУБД різних постачальників, але і у СУБД різних версій від одного постачальника.

2. Неможливість оцінки якісно-кількісних характеристик інформації для оптимальної реплікації.

Виявлення змін за допомогою подій або тригерів. Система відстежує кожну зміну, вироблювану в базі даних за допомогою подій, які прив'язані до кожного виду команд (вставити, змінити, видалити). На рисунку 1.8 наведений приклад реалізації цього методу.

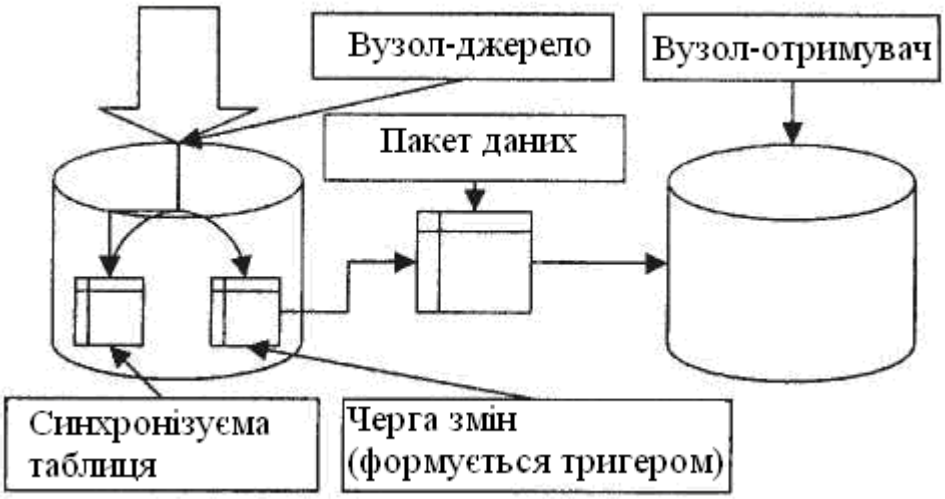


Рисунок 1.8. Метод, заснований на тригерах

Переваги:

1. Робота з програмами сторонніх розробників. Це означає, що зміна, зроблена в таблиці, що модифікується, будь-яким програмним засобом, буде врахована.

Недоліки:

1. Потрібна підтримка механізму тригерів (подій), що не завжди реалізоване в СУБД для персонального використання.

2. Потрібна модифікація існуючої структури бази даних, що особливо складно, якщо механізм тригерів вже був задіяний для інших цілей (доводиться

об'єднувати код старих тригерів з новим).

3. Не можна проводити автоматичну повну синхронізацію даних (передачу таблиць цілком у випадку, якщо таблиці на вузлі, що синхронізується, порожні).

4. Можливе сильне розростання існуючої бази даних за рахунок зберігання додаткових даних.

Виявлення змін за допомогою контрольних таблиць на перший погляд схоже на метод подвійного копіювання таблиць. Для кожної реплікуємої таблиці заводиться додаткова (контрольна) таблиця, в якій для кожного запису в таблиці указується деяка контрольна інформація, що включає час зміни даних, а також джерело оновлення. Час зміни для кожного запису оновлюється автоматично за допомогою тригерів. Абсолютно необхідним для цієї операції є тригер на оновлення. Тригери на створення і на видалення не є обов'язковими, проте їх наявність дозволить уникнути використання додаткових засобів для очищення контрольних таблиць від інформації про видалені записи, і дасть можливість отримувати точніші дані про час внесення змін. Цей метод, насправді, є поліпшенням методу тригерів або подій. Він володіє практично тими ж перевагами і недоліками, що і метод тригерів. Плюсом цього методу є те, що він не створює необмежено зростаючої черги подій.

1.3 Імітаційне моделювання при визначенні цінності інформації

Суть методу полягає в такому: за допомогою імітаційної моделі перебираються можливі стани системи в деякий момент часу. Для кожного стану виробляється оцінка значення критерію ефективності функціонування системи в наступний момент часу. Для цього використовується модель керованого об'єкта й алгоритм прийняття рішень з керування об'єктом. Керування об'єктом здійснюється на підставі значень вихідних величин системи. Значення критерію ефективності системи визначається для двох випадків:

а) рішення про керування об'єктом приймається за наявності інформації, що міститься у вихідних величинах, про стан об'єкта;

б) у разі відсутності такої інформації.

Цінність інформації визначається як різниця між значеннями критерію ефективності функціонування системи для двох зазначених випадків.

Реалізація методу здійснюється табличним способом. Для цього область припустимих значень вихідних величин об'єкта керування розбивається на деяку скінчену множину в загальному випадку нерівних під-областей. Цінність інформації визначається на межах під-областей. Як цінність інформації, що відповідає кожній з під-областей, береться середнє значення про цінності інформації на межах цих під-областей. При керуванні в реальному режимі часу таблиця цінностей інформації для кожної з під-областей зберігається в оперативній пам'яті керуючої ЕОМ. Значення цінності інформації з цієї таблиці вибираються табличним автоматом залежно від значення самої інформації.

Нехай система керування описується лінійними різницевими рівняннями у векторно-матричному записі:

$$\begin{cases} \mathrm{X}_{k+1} = \Theta_{xk} + \Psi_{uk} + \Omega_{fk}, \\ \mathrm{Y}_k = \tilde{A}x_k + Eh_k; \end{cases} \qquad (1.1)$$

де $X_k = [x_{k1}, x_{k2}, x_{k3}, \ldots, x_{kn}]$ – вектор стану; $u_k = [u_{k1}, u_{k2}, u_{k3}, \ldots, u_{kn}]$ – вектор керування; $f_k = [f_{k1}, f_{k2}, f_{k3}, \ldots, f_{kn}]$ – вектор зовнішніх збурень; $h_k = [h_{k1}, h_{k2}, h_{k3}, \ldots, h_{kn}]$ – вектор перешкод; $y_k = [y_{k1}, y_{k2}, y_{k3}, \ldots, y_{kn}]$ – вектор виходу; Θ - власна параметрична матриця системи; Ψ - вхідна матриця системи; Ω – матриця координат зовнішніх збурень; Γ – матриця виходу; E – матриця координат перешкод; $t_k=k\Delta t$; Δt – величина кроку дискретизації; k – номер моменту часу $k=\overline{1,K}$; $t_0=0$, $t_k=K$; X_{ki}, U_{ki}, F_{ki}, H_{ki}, Y_{ki} – області припустимих значень і-х компонентів відповідних векторів у k-й момент часу.

Областями припустимих значень векторів x_k і y_k будемо вважати області, при яких функціонування системи з погляду досягнення поставленої мети ще можливе. Якість функціонування системи характеризується критерієм ефективності[14]:

$$I_k = \rho(x_k, u_k),\ k=\overline{1,K}, \tag{1.2}$$

де k – номер моменту часу, у який здійснюється деякий крок керування системою.

Відповідно до мети керування значення вектора x_k на ділянці часу $\overline{1,K}$ вибираються з усіх варіантів можливих значень x_k, щоб критерій I_k, досягав екстремуму. Для цього на кожному кроці керування системою повинні бути обрані відповідні значення вектора керування x_k на підставі вектора виходу y_k.

Нехай є імітаційна модель системи керування, що дозволяє моделювати процес функціонування системи за кроками. Припустімо, що на деякому кроці керування (у k-1 й момент часу) на підставі вектора виходу y_{k-1}, було обране керування u_k, що привело систему в стан x_k з визначеним значенням критерію I_{k+1}. На наступному кроці (у k-й момент часу) вибирається керування u_k за умови, що $y_k \neq Ø$, що приведе систему в стан x_{k+1} з новим значенням I_{k+1}. Вираз $y_k \neq Ø$ означає, що в систему надходить інформація про дійсні значення всіх компонентів вектора виходу y_k. Таким чином, на кожному кроці керування критерій одержує збільшення $\Delta I_{k+1} = I_{k+1}-I_k$. Якщо в k-й момент часу відсутня інформація про дійсні значення вектора виходу, тобто $y_k = Ø$, то керування u_k, вибиратимемо відповідно до алгоритму прийняття рішень у системі на підставі y_{k-1} або y_{k+1} і u_{k-1} або u_{k-1} і всієї передісторії зміни вектора y_k і т. Прийняте у цьому випадку значення y_k, на основі якого вибиратиметься керування, будемо називати прогнозованим значенням y_{nk}, на відміну від дійсного значення y_k.

Різниця між дійсним і прогнозованим значеннями векторів y_k у k-й момент часу називатимемо збільшенням вектора y_k:

$$\Delta y_k = y_k - y_{nk.} \tag{1.3}$$

Зазначене стосується будь-якого і-го компонента вектора y_k:

$$\Delta y_k = y_{ki} - y_{nki} \ . \tag{1.4}$$

При цьому обране керування u_k може привести систему до необхідного значення збільшення ΔI_{k+1} тільки в тому випадку, коли дійсне значення вектора y_k дорівнює прогнозованому y_{nk}. При $y_k \neq y_{nk}$ критерій I_{k+1} матиме збільшення ΔI_{k+1}, що відрізняється від ΔI_k.

Цінність інформації вектора виходу в к-й момент часу визначається виразом

$$Ц(y_k) = \Delta I_{k+1} - \Delta I'_{k+1}, \tag{1.5}$$

тут

$$\Delta I_{k+1} = I_{k+1} - I_k, \tag{1.6}$$

$$\Delta I'_{k+1} = I'_{k+1} - I'_k, \tag{1.7}$$

де I_k - значення критерію ефективності системи в к-й момент часу; I_{k+1} - значення критерію ефективності системи в $k+1$-й момент часу за наявності інформації про всі компоненти вектора y_k; I'_{k+1} - значення критерію ефективності системи в $k+1$-й момент часу за відсутності інформації про всі компоненти вектора y_k.

Підставивши (1.6) і (1.7) у (1.8), отримаємо:

$$Ц(y_k) = I_{k+1} - I'_{k+1}; \quad (1.8)$$

Іншими словами, у виразі (1.8) визначається цінність інформації про збільшення вектора y_k в k-й момент часу. Однак нас цікавить цінність інформації, що міститься в кожному і-му компонентові вектора y_k.

Здійснюючи імітаційне моделювання функціонування системи, зафіксуємо на деякому k-му кроці значення всіх, крім і-го, компонентів вектора y_k на рівні їх прогнозованих значень, тобто:

$$y_k = [y_{nk1}, y_{nk2}, y_{nk3}, \ldots, y_{nkp}]. \quad (1.9)$$

Тоді цінність інформації, що міститься в і-му компонентові вектора y_k, можна визначити за формулою

$$Ц(y_{ki}) = I_{nk+1} - I'_{nk+1}, \quad (1.10)$$

де I_{nk+1} – значення критерію ефективності системи в k+1-й момент час при прогнозованому значенні всіх, крім *і*-го, компонентів вектора y_k, при довільному значенні з області припустимих значень ΔY_{ki} збільшення Δy_{ki} за умови наявності в системі інформації про дійсне значення і-го компонента вектора $y_к$; I'_{nk+1} - значення критерію ефективності системи в k+1-й момент часу при прогнозованих значеннях усіх компонентів вектора $y_к$ і за умови відсутності в системі інформації про дійсне значення і-го компонента вектора y_k .

Збільшення Δy_{ki} може змінюватися в межах

$$\Delta y_{min\ ki} < \Delta y_{ki} < \Delta y_{max\ ki}, \quad (3.11)$$

$$\Delta y_{min\ ki} = y_{min\ ki} - y_{max\ ki}, \quad (3.12)$$

$$\Delta y_{min\ ki} = y_{max\ ki} - y_{min\ ki}, \quad (3.13)$$

де $y_{min\ ki}$ та $y_{max\ ki}$ – відповідно мінімальне та максимальне значення області Y_k припустимих значень і-го компонента в k-й момент часу, тобто:

$$Y_{kl} = \overline{y_{\min ki}, y_{\max ki}}\ . \qquad (3.14)$$

Фіксація усіх, крім і-го, компонентів вектора y_k відбувається з метою виключення впливу цих компонентів на різницю значень критерію ефективності системи.

1.4 Постановка задачі дослідження

У теперішній час широке застосування набули методи та засоби організації процесу синхронізації та реплікації баз даних, які постійно розвиваються. Виходячи з цього зростають і вимоги до них. Урахування якісно-кількісних характеристик при проведенні реплікації розподілених баз даних, є досить актуальною задачею та розробляється на підставі необхідності, що виникає у роботі торгівельних мереж. Основним призначенням є оптимізація проведення реплікацій між вузлами розподіленої бази даних автоматизованої інформаційної системи управління торговельною мережею.

Із поставленої задачі виділимо основні завдання для магістерської роботи:

1) Аналіз особливостей організації процесу синхронізації баз даних.

2) Дослідження відомих методів реплікації баз даних та аналіз їх переваг та недоліків.

3) Розробка методів визначення цінності поточної інформації з використанням функції чутливості.

4) Практична реалізація системи синхронізації баз даних користувачів мультиплатформеного веб-сайту.

5) Апробація розроблених методів та засобів.

Висновки до розділу 1

В даному розділі розглянуто теоретичні основи та особливості організації розподілених баз даних, обґрунтовано способи виявлення змін, розкрито імітаційне моделювання при визначенні цінності інформації та поставлені задачі дослідження.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ ТА ОБГРУНТУВАННЯ РІШЕНЬ ПО СТВОРЕННЮ СИСТЕМИ СИНХРОНІЗАЦІЇ БАЗ ДАНИХ КОРИСТУВАЧІВ

2.1 Проблема реплікації систем розподілених баз даних

Проблема реплікації (передачі даних між вузлами розподіленої системи) постає перед торговельною мережею практично щодня. Зокрема, таке завдання може зустрітися при передачі даних про щоденні обсяги продаж, проведення звітного контролю на окремих вузлах торговельної мережі, або у разі, коли деякий управляючий орган веде набір довідників, кожен з яких обов'язково має бути присутнім в базі даних інших вузлів, що звітують перед ним. І, хоча вже сформувалися найбільш часто використовувані підходи до реалізації алгоритмів реплікації, всі вони не позбавлені недоліків, про які буде сказано нижче.

Ідея додати функції реплікації даних між видаленими вузлами розподіленої бази даних виникла в сімдесятих роках минулого століття. Одній з перших робіт по цій темі є [2]. Завдання синхронізації таблиць є окремим випадком реплікації даних, при якому потрібний, щоб у двох таблиць, що знаходяться на віддалених вузлах, було однакове наповнення. У даному підрозділі приводиться аналіз методів реплікації даних, що використовуються при реплікації у торгівельній мережі, з урахуванням критеріїв старіння, ціни та цінності інформації.

У [12] дається аналіз і опис основних характеристик існуючих систем реплікації даних. Ідеальна система повинна задовольняти наступні вимоги:

1. Мінімізація впливу на вже існуючі програмні комплекси і бази даних. Система реплікації не повинна вимагати від програміста звернення до сторонніх інтерфейсів і написання спеціальних засобів доступу до бази даних. Також недоцільно вимагати зміни існуючих структур баз даних.

2. Система повинна виявляти будь-які зміни в базі даних. Вони можуть бути зроблені як деяким програмним забезпеченням сторонніх виробників, так і за допомогою самої системи управління базою даних.

3. Система має бути ефективною, такою, що масштабується і мати прогнозовані витрати при розширенні. Система не повинна дублювати записи, що зберігаються, частково або повністю. Система не повинна зберігати довгі ланцюжки змін в базі. Вона має бути ефективною навіть на дуже великій кількості записів.

4. Система має бути достатньо автономна. Не можливо вимагати від окремих торговельних точок присутності адміністратора баз даних на кожному вузлі розподіленої мережі.

5. Система має однаково добре функціонувати на абсолютно різних платформах. Вона не повинна покладатися на таких, специфічні для кожної бази речі, як журнал транзакцій і тому подібне. Відзначимо, що в реальному житті цієї вимоги досягти достатньо складно, оскільки окрім специфіки реалізації внутрішньої структури існує ще специфіка мови доступу до бази даних. І, хоча теоретично всі SQL-сумісні бази даних повинні відповідати деякому єдиному стандарту, на практиці це далеко не так.

6. Відзначимо, що існуючі на даний момент системи реплікації частенько не задовольняють всім вказаним принципам, та практично усі вони не враховують якісно-кількісні характеристики інформації для побудови оптимальної черги транзакцій для реплікації.

2.2 Принципи встановлення з’єднання при реплікації

Системи реплікації, що використовуються у торговельній мережі, можна класифікувати по різних ознаках [17]. Однією з таких ознак є принцип встановлення з’єднання.

Системи з чергою повідомлень накопичують зміни у вигляді повідомлень, які передаються в порядку «перший прийшов, — перший пішов» у момент

сеансу реплікації на віддалений вузол. Середовищем передачі даних може бути як деяка мережа передачі даних, так і email, ftp і тому подібне. Такий спосіб передачі чимось схожий на взаємодію між комп’ютерами в Інтернеті за допомогою дейтаграм (UDP). Зміни, що передаються при реплікації, можуть накопичуватися або у вигляді знімків даних, або безпосередньо в журналі транзакцій (тобто для накопичення змін не потрібні ніяких спеціальних засобів формування цього списку). Проаналізуємо переваги і недоліки цього методу.

Переваги:

1. Відсутність інтенсивних обчислень у момент передачі, процес захоплення змін рівномірно розподілений по всьому часу редагування бази даних.

2. Відсутність необхідності інтенсивного двостороннього обміну даними і, як наслідок, простота реалізації. Один вузол відправляє повідомлення, а інший проводить їх аналіз.

Недоліки:

1. Можливе розростання цієї черги повідомлень, аж до перевищення нею розміру початкової таблиці. Навіть в тому випадку, якщо в базі зберігаються тільки підсумкові зміни даних між сеансами реплікації, об'єм пам'яті, займаний захопленими змінами, буде, як мінімум, збігатися з об'ємом відстежуваних даних. Якщо ж система спирається на журнал транзакцій, то, по-перше, це не позбавляє від інтенсивного використання ресурсів системи, але також вводить додаткову проблему. Очищення журналу транзакцій є рутинною операцією по поліпшенню продуктивності системи, проте в такому разі всі зміни будуть втрачені. Таким чином, всі люди, що працюють з базою, мають бути повідомлені про категоричну заборону таких дій.

2. Наявність проблеми відновлення видаленої копії після збою. У випадку якщо зміни були передані на видалений вузол, а потім на цьому вузлі стався збій, немає ніякої можливості автоматично відновити ідентичність баз. Щоб це було можливо, необхідно зберігати всі зміни вже

після їх відправки з самого початку здійснення реплікації між вузлами. Це, як правило, неможливо з міркувань витрат ресурсів.

3. Неможливість проведення аналізу кількісно-якісних характеристик передаваємої інформації.

Системи зі встановленням з'єднання орієнтовані на тісну взаємодію між вузлами в процесі реплікації. Цей спосіб, по суті, схожий на протокол TCP. Тобто, власне до моменту закінчення обміну даними відбувається багатократний обмін різною інформацією. Розглянемо цей спосіб передачі даних докладніше.

Переваги:

1. Значне скорочення витрат на зберігання службової інформації. При такому підході не відбувається накопичення даних про зміни, оскільки завжди порівнюються існуючі версії даних. Відзначимо, що ця перевага сходить нанівець, якщо синхронізуються таблиці, для яких однією з вимог є ведення історії змін.

2. Не потрібна модифікація існуючої бази даних, принаймні, не змінюються існуючі таблиці.

3. Не потрібна постійна присутність адміністратора бази даних і контроль системи накопичення змін. Якщо проблеми і виникають, то тільки в строго визначені моменти, при цьому відновлення системи (приведення системи в синхронний стан) відбувається швидко і без особливих витрат.

Недоліки:

1. Виявлення змін відбувається вже протягом сеансу обміну даними, що означає додаткове обчислювальне навантаження.

2. Більшість методів синхронізації вимагає наявність швидкого фізичного каналу передачі даних між вузлами системи. І, хоча гіпотетично, здійснювати реплікацію з використанням системи встановленням з'єднання можна і за відсутності швидкого фізичного каналу даних, на практиці більшість методів синхронізації не будуть достатньо ефективними.

2.3 Синхронізація та реплікація баз даних

Синхронізація баз даних – ліквідація відмінностей між двома копіями даних. Під реплікацією баз даних розуміється механізм синхронізації вмісту декількох копій об’єкта (наприклад, вмісту бази даних). При реплікації зміни, зроблені в одній копії об’єкта, можуть бути поширені в інші копії. Синхронізація може бути синхронною або асинхронною. У випадку синхронної реплікації, якщо дана репліка оновлюється, всі інші репліки того ж фрагмента даних також повинні бути оновлені в одній і тій же транзакції. Це означає, що існує лише одна версія даних. У більшості продуктів синхронна реплікація реалізується за допомогою тригерних процедур (можливо, прихованих і керованих системою). Але синхронна реплікація має той недолік, що вона створює додаткове навантаження при виконанні всіх транзакцій, в яких оновлюються репліки (крім того, можуть виникати проблеми, пов'язані з доступністю даних). У випадку асинхронної реплікації оновлення однієї репліки поширюється на інші через деякий час, а не в тій же транзакції. Таким чином, при асинхронній реплікації вводиться затримка, або час очікування, протягом якого окремі репліки можуть бути фактично неідентичних (тобто визначення репліки виявляється не зовсім відповідним, оскільки ми не маємо справу з точними і своєчасно створеними копіями) [6].

У більшості продуктів асинхронна реплікація реалізується за допомогою читання журналу транзакцій або постійної черги тих оновлень, які підлягають поширенню. Перевага асинхронної реплікації полягає в тому, що додаткові витрати реплікації не пов'язані з транзакціями оновлень, які можуть мати важливе значення для функціонування всього підприємства і пред’являти високі вимоги до продуктивності.

До недоліків цієї схеми відноситься те, що дані можуть виявитися несумісними (тобто несумісними з точки зору користувача). Іншими словами, надмірність може проявлятися на логічному рівні, а це, строго

кажучи, означає, що термін контрольована надмірність в такому випадку не застосуємо.

Розглянемо коротко проблему узгодженості (або, швидше, неузгодженості).Справа в тому, що репліки можуть ставати несумісними в результаті ситуацій, які важко (або навіть неможливо) уникнути і наслідки яких важко виправити. Зокрема, конфлікти можуть виникати з приводу того, в якому порядку повинні застосовуватися оновлення. Наприклад, припустимо, що в результаті виконання транзакції A відбувається вставка рядка в репліку X, після чого транзакція B видаляє цей рядок, а також припустимо, що Y - репліка X. Якщо поновлення поширюються на Y, але вводяться в репліку Y в зворотному порядку (наприклад, через різні затримки при передачі), то транзакція B не знаходить в Y рядок, що підлягає видаленню, і не виконує свою дію, після чого транзакція A вставляє цей рядок. Сумарний ефект полягає в тому, що репліка Y містить зазначений рядок, а репліка X - ні. В цілому завдання усунення конфліктних ситуацій і забезпечення узгодженості реплік є досить складними. Слід зазначити, що в співтоваристві користувачів комерційних баз даних термін реплікація став означати переважно (або навіть виключно) асинхронну реплікацію [6].

Основна відмінність між реплікацією і управлінням копіюванням полягає в наступному: якщо використовується реплікація, то оновлення однієї репліки в кінцевому рахунку поширюється на всі інші автоматично. У режимі керування копіюванням, навпаки, не існує такого автоматичного розповсюдження оновлень. Копії даних створюються і управляються за допомогою пакетного або фонового процесу, який відділений в часі від транзакцій оновлення.

Управління копіюванням в загальному більш ефективно в порівнянні з реплікацією, оскільки за один раз можна буде скопіювати великі обсяги даних. До недоліків можна віднести те, що більшу частину часу копії даних не ідентичні базовим даними, тому користувачі повинні враховувати, коли саме були синхронізовані ці дані.

У розподілених базах даних з підтримкою реплікації кожному логічному елементу даних відповідає декілька фізичних копій. Так, розмір зарплати деякого службовця (логічний елемент даних) може зберігатися на трьох вузлах (фізичні копії). У такого роду системах виникає проблема підтримкою узгодженості копій. Найбільш відомим критерієм узгодженості є критерій повної еквівалентності копій (one copy equivalence), який вимагає, аби після закінчення транзакції всі копії логічного елементу даних були ідентичні.

Якщо підтримується прозорість реплікування, то транзакція видаватиме операції зчитування-запису, що відносяться до логічного елементу даних x. Протокол управління репліками відповідає за відображення операцій над x в операції над фізичними копіями x (x_1, x_2..., x_n). Типовий протокол управління репліками, що дотримується критерію повної еквівалентності копій, відомий під назвою ROWA (Read-Once/Write-All – читання з однієї копії, запис у всі копії [7]). Протокол ROWA відображує читання x [Read(x)] в операцію читання який-небудь з фізичних копій x_i [Read(x_i)]. З якої саме копії проводитиметься читання, неважливо – це питання може вирішуватися з міркувань ефективності. Кожна операція запису в логічний елемент даних x відображується на безліч операцій запису у всі фізичні копії x.

Протокол ROWA простий і прямолінійний, але він вимагає доступності всіх копій елементу даних, аби транзакцію можна було термінувати. Збій на одному з вузлів приведе до блокування транзакції, що знижує доступність бази даних.

Було запропоновано декілька альтернативних алгоритмів, направлених на пом'якшення вимоги про те, що для завершення транзакції необхідно внести зміни в кожну копію елементу даних. Всі вони ослабляють ROWA, зіставляючи операцію запису з деякою підмножиною фізичних копій.

Ідея, згідно якої для завершення транзакції досить модифікувати деяку підмножину копій, лягла в основу механізму голосування на основі кворуму,

використовуваного в протоколах управління репліками. Алгоритм консенсусу більшості можна сформулювати небагато з іншої точки зору: кожній копії даних приписується одне і те ж число голосів, і транзакція, що змінює логічний елемент даних, може успішно завершитися, якщо вона набере більшість голосів. На тій же ідеї заснований ранній алгоритм голосування на основі кворуму (quorum-based voting algorithm) [3], який також присвоює копіям даних (можливо, не одне і те ж) число голосів. Для виконання будь-якої операції читання або запису елементу даних потрібно отримати кворум читання (read quorum) (Vr) або кворум запису (write quorum) (Vw). Якщо елемент даних має в сумі V голосів, то при визначенні кворуму повинні дотримуватися наступні правила.

1. $V_r + V_w > V$ *(дві транзакції не можуть одночасно читати і модифікувати один і той же елемент даних щоб уникнути конфлікту читання-запис);*

2. $V_w > V/2$ *(дві транзакції не можуть одночасно проводити запис одного і того ж елементу даних щоб уникнути конфлікту запис-запис).*

Проблема, властива цьому підходу, полягає в тому, що транзакція повинна набрати кворум навіть для читання. Через це істотно і невиправдано сповільнюється доступ до бази даних на читання. Був запропонований альтернативний протокол голосування на базі кворуму, де цей серйозний недолік долається [Abbadi et al., 1985]. Проте цей протокол виходить з абсолютно нереалістичних припущень про властивості комунікаційної системи. Базові вимоги полягають в тому, що всім вузлам негайно стає відомо про відмови, що наводять до змін в топології мережі, і кожен вузол має в своєму розпорядженні представлення тій частині мережі, де містяться вузли, з якими він взаємодіє. У загальному випадку неможливо гарантувати здійснимість цих вимог. Таким чином, протокол повної еквівалентності копій є обмежувальним з точки зору доступності системи, а протоколи, засновані на голосуванні, дуже складні, і з ними пов'язані високі накладні витрати. Тому в сучасних промислових розподілених СУБД жоден з цих методів не

використовується. Для практичного вживання були досліджені деякі гнучкіші технології реплікацій, де тип узгодження копій знаходиться під контролем користувача. На основі цього принципу вже створено або створюється декілька різновидів серверів реплікації. На жаль, в даний час не існує стрункої теорії, яка б дозволяла судити про узгодженість реплікованої бази даних в умовах, коли застосовуються відносно нестрогі політики реплікацій.

2.4 Імовірнісне виявлення змін процесу синхронізації

У даному випадку розглядуються методи виявлення змін, які не гарантують приведення баз в синхронізований стан впродовж одного сеансу реплікації. Проте алгоритми, використовувані в цих системах, дозволяють понизити витрати окремих типів ресурсів. Розглянемо ці алгоритми докладніше. Аналіз вірогідності зміни приводиться в [5]. Спочатку він був розроблений для побудови ефективних алгоритмів оновлення баз пошукових систем пошуковими роботами. Розгледимо можливість застосування такого алгоритму до баз даних. Ідея цього алгоритму полягає в тому, що проводиться аналіз частоти змін того або іншого об'єкту і на основі статистичних даних проводиться спроба побудови прогнозу частоти подальших змін в базі даних. Спочатку авторами в якості об'єкту синхронізації розглядувалися сторінки веб-сайтів. Проте той же підхід можна спробувати застосувати і до баз даних. Тоді об'єктами синхронізації будуть записи в таблиці або групи записів. Передбачається, що той або інший об'єкт синхронізації міняється, підкоряючись певній моделі, наприклад пуасонівскому закону. Ця модель може бути одна для всіх даних об'єктів або варіюватися у груп об'єктів або навіть у окремих об'єктів. Відповідно до цієї моделі передбачається, що в певний момент часу об'єкт змінився (хоча насправді цього може і не бути).

Об’єкт пересилається на інший вузол, де проводиться аналіз, чи дійсно він змінився, чи ні. На підставі цих даних може мінятися або сам закон, що описує

частоту змін, або якісь його параметри. При такому методі часто можливі такі ситуації, коли об'єкти, які насправді не змінювалися, були переслані на інший вузол. Використання цього методу може бути доцільним в дуже специфічному випадку, коли необхідно мінімізувати обчислювальні витрати на вузлі-джерелі, дані в реплікуємих таблицях міняються з якоюсь прогнозованою вірогідністю і стовідсоткова відповідність бази вузла-отримувача базі вузла-джерела не є важливою вимогою. Такі системи в реальному житті зустрічаються достатньо рідко. На рисунку 2.1 приведена спрощена схема роботи цього алгоритму.

Переваги:

1. Не потрібна модифікація бази даних вузла джерела. Вузол-джерело може взагалі не бути повідомлений про те, що він бере участь в деякій системі реплікації.

Недоліки:

1. Відсутність гарантії синхронізованого стану баз даних після закінчення сеансу реплікації.

2. Неврахування якісно-кількісних характеристик для оптимальної реплікації.

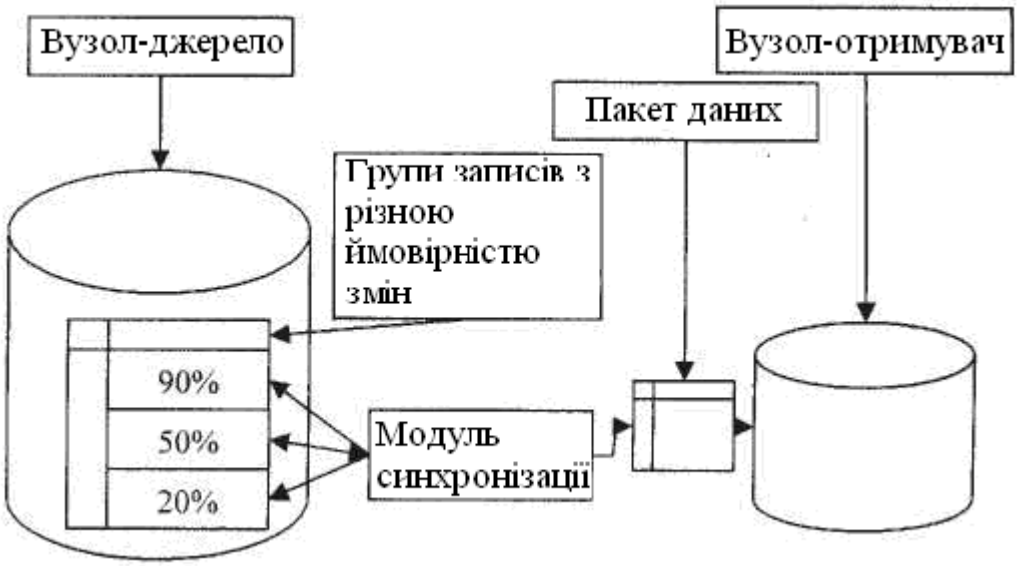


Рисунок 2.1. Аналіз вірогідності зміни груп записів

Аналіз змін представників груп, як і попередній метод, був запропонований в [9] як спосіб оптимізувати алгоритми синхронізації баз пошукових систем мережі Інтернет. Як і в попередньому методі, об'єктом синхронізації є сторінка веб-сайту. Розгледимо, чи можна як такий об'єкт використовувати просто запис в таблиці. Всі об’єкти об'єднуються в деяку єдину групу за певною ознакою. Для веб-сторінок цією ознакою є приналежність до одного сайту. Для записів бази даних необхідно виділяти цю ознаку, виходячи з припущення про одночасну зміну цих записів. Наприклад, в погано нормалізованих базах можлива ситуація, коли зміна одному запису спричиняє за собою зміну великого числа підлеглих записів. На етапі синхронізації для кожної групи вибирається представник, тобто об'єкт, що підлягає безумовній передачі по каналу зв'язку. Якщо цей об'єкт змінений, то вся група об'єктів визнається зміненою і підлягає безумовній передачі по каналу зв'язку.

Основною проблемою при реалізації цього алгоритму є правильний вибір групуючої ознаки. Використовувати його представляється можливим також в досить специфічних випадках: бази, в яких можливе групування об'єктів за такому ознакою, з'являються або в результаті поганого проектування, або у випадках, подібних до баз пошукових систем (коли присутній неявний зв'язок між різними об'єктами). Переваги і недоліки цього методу повністю збігаються з перевагами і недоліками методу з попереднього пункту.

На рисунку 2.2 показано два етапи роботи цього алгоритму.

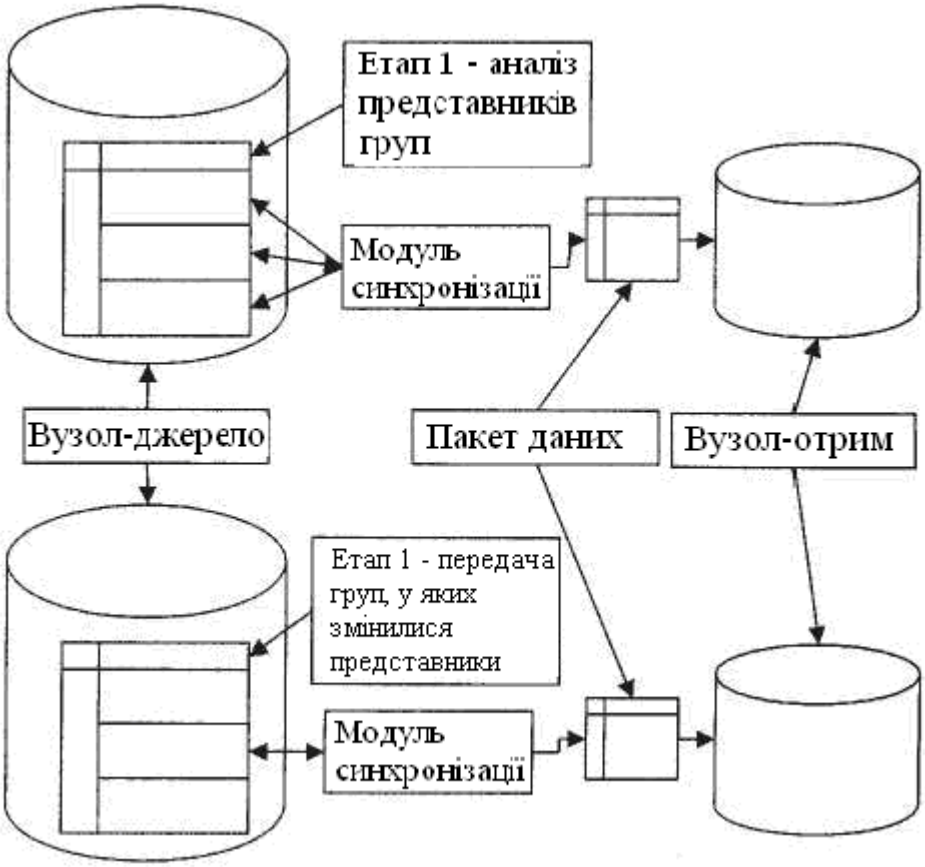


Рисунок 2.2. Метод аналізу представників груп

Виявлення змін за допомогою хеш-функцій. Те, як система реплікації проводить вибірку даних, що підлягають пересилці на видалений вузол, безпосередньо впливає на продуктивність системи. Можливі два способи виявлення змін: реплікація по поточному стану і дельта-реплікація. Реплікація по поточному стану — це такий спосіб реплікації, при якому постачальник даних ще до початку обміну даними (або після обміну якийсь однією міткою) знає, котрі дані необхідно поставити вузлу-отримувачеві. При дельта-реплікації вузол-постачальник і вузол-отримувач спільними зусиллями з'ясовують, які саме дані необхідно переслати. При такому способі реплікації відбувається значне навантаження на мережу, яка частенько не дозволяє проводити цю операцію ефективно. Покажемо, що існують алгоритми дельта-реплікації, що дозволяють здійснювати синхронізацію баз даних при знижених витратах мережевого трафіку.

Схожа проблема встає при необхідності синхронізації файлів. Синхронізувати об'єкти такого типа ефективніше було б за допомогою розбиття їх на деякі шматки і передачі по мережі тільки тих з них, які були змінені з моменту останнього сеансу передачі. Проте файлові системи (принаймні, більшість з них) не підтримують механізму подій (тригерів, в термінології баз даних). Зберігання метаданих для таких об'єктів також є проблемою при реалізації. На допомогу в даному випадку приходить механізм хеш-функцій. Хеш-функція ставить у відповідність вхідному ряду довільного розміру деякий вихідний ряд фіксованого розміру. При цьому, якщо хеш-функція повертають різні значення для двох вхідних рядків, то можна стверджувати, що ці рядки різні. Якщо ж хеш-функція двох вхідних рядків збігаються, то можна з великою вірогідністю стверджувати, що і самі вхідні рядки збігаються (вірогідність того, що збіжаться хэш-значения для двох різних функцій, наприклад для алгоритму МБ5, складає 2~64 [10]). У [4] приводиться алгоритм, використовуваний в утиліті RSync, що є утилітою для підтримки синхронізованого стану двох файлів. У цій утиліті алгоритм ототожнення на основі хеш-функцій використовується для зменшення мережевого трафіку. Розгледимо цей алгоритм трохи детальніше.

1. Вузол-отримувач розбиває файл на деяку кількість однакових інтервалів фіксованої довжини. Для кожного з інтервалів обчислюється «швидка контрольна сума» (деяка функція, яка обчислюється дуже швидко, проте з деякою ненульовою вірогідністю може давати однакові результати на різних даних) і хеш-значення. Ці дані відправляються вузлові-джерелу даних.

2. Вузол-джерело обчислює «швидкі контрольні суми» для всіх інтервалів вказаної довжини. У множини обчислених сум шукаються співпадання з сумами, отриманими від вузла-отримувача. Для знайдених пар збіг додатково перевірятимуться за допомогою строгішої хеш-функції.

3. Інтервали, для яких не було знайдено збігів, відправляються на вузол-отримувач.

В [6] автори застосовують цей алгоритм до проблеми синхронізації баз даних. База даних розглядується як лінійна структура, де операції проводяться з рядками таблиці цілком. Тобто або рядок (запис) передається на видалений вузол, або не існує. Таблиця розбивається на деякі інтервали, для кожного з яких обчислюється хеш-значення. Для тих інтервалів, для яких хеш-значення на вузлі-джерелі та на вузлі-отримувачі не збігаються, проводиться подрібнення інтервалів. Після чого для кожного з отриманих інтервалів операція повторюється. Процес продовжується до тих пір, доки величина інтервалів не досягне деякого порогового значення, або інтервал не дорівнюватиме одному запису.

Опишемо цей алгоритм докладніше:

1. Визначити множину загальної суті на вузлі-постачальникові і на вузлі-отримувачеві (суть може містити різні дані, але повинні мати однакові ідентифікатори).

2. Розбити безліч загальної суті на інтервали заданої величини (це є параметром алгоритму).

3. Обчислити хеш-функцій для заданих інтервалів і передати обчислені хеш-значення на видалений вузол.

4. На видаленому вузлі обчислити значення хеш-функцій і обчислені хеш-значення порівняти зі значеннями, отриманими від вузла-постачальника.

5. Інтервали, для яких обчислені локально і отримані хеш-значення не збігаються, визнаються різними, і вже для безлічі змінених інтервалів повторюється крок 3.

6. Цей процес повторюється до тих пір, поки розмір подрібненого інтервалу не дорівнюватиме одному запису або ж не досягне деякого порогового значення.

Розгледимо переваги та недоліки цього алгоритму.

Переваги:

1. Не вимагається зміни існуючої бази даних або програмного

забезпечення.

2. Не вимагається ведення деякого журналу змін.

3. Істотна економія мережевого трафіку, оскільки передаються лише ті дані, які потрібно синхронізувати.

Недоліки:

1. Підвищені вимоги до обчислювальних ресурсів.

2. Існує достатньо мала вірогідність несинхронізованого стану двох баз даних після завершення процесу реплікації.

3. Можливість ототожнення записів на підставі хеш-значень.

Описаний вище спосіб ототожнення записів викликає багато суперечок. У [10] приводиться аналіз можливості застосування такого способу ототожнення об'єктів для різних типів завдань. Покажемо, що для завдань реплікації такий спосіб ототожнення записів використовувати можна, розгледівши можливі проблеми його використання і способи їх усунення.

Невипадковість вхідних даних. Для того, щоб показати, що вірогідність збігу хеш-значень для вхідних даних дуже мала, багато учених (наприклад [8]) припускають, що вхідні дані являються довільними. У реальному житті це, природно, не так. Покажемо, чому для описаного вирішення поставленого завдання таке допущення не є проблематичним. У [11] наведені приклади атак на існуючі алгоритми. Під атакою хеш-алгоритма мається на увазі тривалий підбір вхідних даних, таких, хеш-значення яких збігатимуться. Для того, щоб хеш-значення різних вхідних даних збіглися, необхідно, щоб вхідні дані підбиралися по деякому певному закону (наприклад, мали однаковий префікс, що складається із строго певної 1024-байтової послідовності символів), а сам алгоритм ініціалізував певним значенням вектора, що ініціалізував (ініціалізація алгоритму певним значенням дозволяє, не міняючи алгоритму, повністю змінити безліч хеш-значень, що повертаються). Вірогідність того, що в реальній базі даних зустрінуться такі значення, близька до нульової з тієї причини, що зазвичай в базі даних зберігається інформація, що відображає дані з реального світу, а вхідні дані, що дають хеш-колізію, з цієї точки зору

абсолютно безглузді. Крім того, цілком можна поліпшити рівномірність розподілу інформації за вхідними даними, заздалегідь стискує їх за допомогою якого-небудь алгоритму стискування без втрат (ZIP, RAR і т.д.).

Старіння хеш-алгоритмів. Після появи нового хеш-алгоритму, що претендує на стійкість до колізій, сотні математиків починають займатися дослідженням цього алгоритму. Кінець кінцем, для будь-якого алгоритму знаходяться можливі колізії. Проте це не є проблемою. Сучасні середовища розробки дозволяють з легкістю описувати алгоритми, що використовують хешування, не указуючи явно спосіб хешування. Таким чином, оновлення хеш-алгоритмів може бути проведене простою зміною конфігурації системи без перекомпіляції початкового коду.

Проблема помилок, що складно виявляються. У випадку, якщо все-таки настала колізія, може статися ситуація що дві бази знаходитимуться в розсинхронізованому стані після завершення сеансу синхронізації. Звичайно, таку можливість необхідно в явному вигляді вказати в документації до системи реплікації. Проте існує ряд способів, що дозволяють з дуже великою вірогідністю привести бази даних в синхронізований стан вже під час наступного сеансу синхронізації. Це такі способи, як:

- зміна хеш-алгоритма після кожного сеансу передачі;
- ініціалізація хеш-алгоритму іншим, ключем, що довільно згенерував;
- зміна множини аналізованих інтервалів. Проблема збільшення вірогідності помилки функціонування системи. У [10] піднімається філософська проблема: чи можна використовувати вірогідність помилки апаратних засобів (наприклад: 0,000000005 для протоколу TCP-IP [7]) як верхню межу допустимої вірогідності помилок для програмних засобів (можна приблизно сказати, що для алгоритму MD5 вірогідність колізії складає 2"64). Можна [20] стверджувати, що при такій малій вірогідності подія колізії на реальних даних (не підібраних спеціально) ніколи не настане.

Існують різні способи синхронізації баз даних. Більшість з них вимагають зміни існуючих баз і/або програмного коду. Більшість алгоритмів, які не вимагають цього, є не ефективними з погляду продуктивності.

Описаний алгоритм порівняння на основі хеш-функцій виявляється особливо зручним в разі обмеженої можливості зміни бази даних і початкового коду, а також в разі наявності не дуже швидкого каналу, що сполучає два видалені вузли.

Висновки до розділу 2

Даний розділ присвячений проблема реплікації систем розподілених баз даних, висвітлені принципи становлення з’єднання при реплікації, розкрита синхронізація та реплікація баз даних, особливу увагу приділено ймовірнісному виявленню змін процесу синхронізації.

РОЗДІЛ 3
МЕТОДИ ВИЗНАЧЕННЯ ЯКІСНО-КІЛЬКІСНИХ ХАРАКТЕРИСТИК ІНФОРМАЦІЇ

3.1 Визначення цінності поточної інформації

У цьому підрозділі розглянемо принципи використання методів визначення цінності поточної інформації. Ці методи орієнтовані на застосування в системах управління різних типів і призначень.

Зараз різні автори розглядають цінність і старіння інформації з різних позицій, вкладаючи при цьому різні поняття в їх визначення. Часто справедливо підкреслюють, що цінність інформації нерозривно пов'язана з її старінням, проте облік цих факторів здійснюється зовсім по-різному. Наприклад, старіння інформації визначається як закон зміни цінності інформації в часі, що схематично зображено на рисунку 3.1.

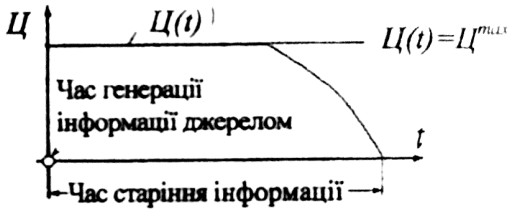


Рисунок 3.1. Зміна в часі характеристик цінності і старіння інформації

Таким чином, закон зміни старіння інформації інтерпретується як закон зміни цінності інформації в часі. Цим і визначається взаємозв'язок характеристик цінності і старіння. Підкреслимо істотний момент: взаємозв'язок цінності і старіння інформації в даному випадку визначено, але при цьому фактично зникають цінність і старіння інформації як самостійні і не залежні одне від одного поняття. Описаний вище підхід до визначення цінності і старіння інформації та їх взаємозв'язку (назвемо його першим, він типовий для великої кількості робіт) суперечить сформульованим за-

конам інформації. Відповідно до цих законів цінність інформації визначається залежно від мети її одержання, а старіння - як рівень відхилення матеріального явища від інформації, що відображає це матеріальне явище.

Отже, цінність однієї й тієї ж інформації для різних цілей використання буде різна, а характеристика "відходу" інформації від свого первісного (відображуваного) базису, тобто старіння, не залежить від цілей використання інформації, хоча під час її обробки з різними цілями кількісно старіння інформації може враховуватися по-різному. Це вказує на те, що визначення цінності і старіння інформації та їх взаємозв'язку відповідно до першого підходу ґрунтується на фактично "поєднаних" поняттях несправедливо Треба не поєднувати поняття цінності і старіння інформації, а розглядати їх як самостійні характеристики. Такий підхід (назвемо його другим) використовують Р.Л. Стратанович, А.А. Харкевич, А.Н. Єфімов та ін. [19]. Зазначимо, що другий підхід відповідає законами інформації. Однак він не встановлює що взаємозв'язок цінності інформації та її старіння визначається як рівень відходу матеріального явища від свого первісного значення.

Розглянемо визначення цінності і старіння інформації та їх взаємозв'язок у рамках другого підходу. Нехай дано певне матеріальне явище, що зображується кривою (Рисунок 3.2).

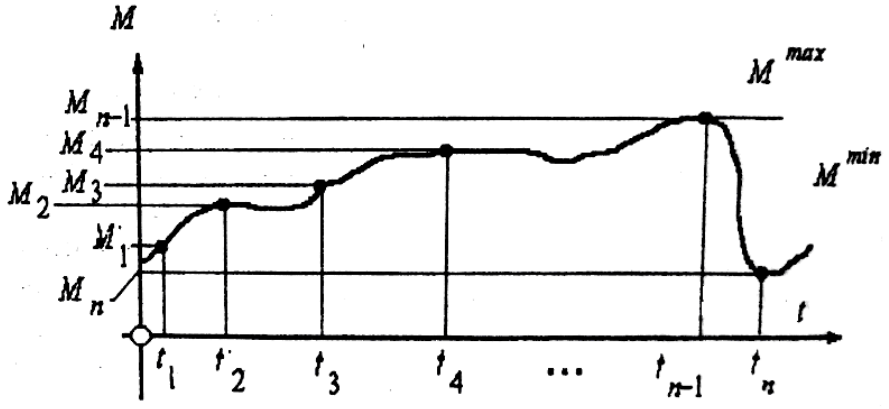


Рисунок 3.2. Зміна матеріального явища *M* в часі

У момент часу t_1, матеріальне явище мало значення M_t, що визначене (зареєстроване) з певною точністю. З часом t матеріальне явище M змінює своє значення і в моменти часу t_2, t_3, $t_4...t_n$ набуває відповідно значення M_2, M_3, $M_4,...,M_n$. Якщо використовувати інформацію про значення матеріального явища M_x в момент часу t_2, то відхід матеріального явища за час (t_2 -t_1,) становитиме M_2-M_1. За час $(t_3 - t_1)$ і (t_4-t_1) відхід матеріального явища від зареєстрованого в момент часу t_1, значення M_1, становитиме, відповідно, M_3-M_1, і M_4-M_1. Узявши момент часу t_1, за момент генерації інформації можна формально визначити старіння інформації, отриманої в момент часу t_1, у різні моменти часу[1]:

$$S_i=M(t)-M(t_i) \qquad (3.1)$$

де $Si(t)$- старіння інформації про матеріальне явище, зареєстроване в момент часу ti.

Визначивши значення $S_i(t)$ для різних значень t, можна побудувати криву старіння інформації, зареєстрованої в момент ti залежно від t.

Очевидно, що визначення старіння інформації відповідно до (3.1) відбиває лише принциповий підхід, який можна реалізувати різними формальними методами. Правильність підходу до визначення старіння інформації відповідно до (3.1) можна обґрунтувати так: фактично $S_i(t)$, адекватно відбиває зміна $M(t)$ від моменту t. Розробка формальних методів визначення $S_i(t)$ - окреме завдання, яке ми розглянемо далі. З урахуванням мети використання в кожному випадку характеристика $S_i(t)$ має бути доповнена значенням $T_{гран}$, - граничним часом старіння інформації. Граничний час старіння інформації - час від моменту $t_г$, за котрий $S_i(t)$ досягає свого граничного значення з урахуванням мети використання. Тому актуальним є завдання визначення $S_i(T_{гран})$ (граничного значення $S_i(t)$ для різних цілей використання, на підставі якого встановлюється значення $T_{гран}$.

Таким чином, визначення старіння інформації має передбачати визначення характеристик $S_i(t)$ і $S_i(T_{гран})$ з подальшим з'ясуванням $T_{гран}$.

3.2 Визначення цінності інформації в системах управління

Розглянемо основні проблеми визначення цінності інформації. Найважливішим питанням теорії інформації є встановлення міри кількості та якості інформації. Інформаційні міри визначаються за такими основними напрямками: структурним, статистичним, семантичним і прагматичним.

Структурна теорія розглядає дискретну побудову масивів інформації і їх вимірювання простим підрахунком інформаційних елементів (квантів) або комбінаторним методом, що передбачає найпростіше кодування масивів інформації.

Статистична теорія оперує поняттям ентропії як міри невизначеності, що враховує імовірність появи, а отже, й інформативність тих або інших повідомлень.

Семантичний напрямок враховує зміст і вміст повідомлень; прагматичний - цінність, корисність інформації для споживача. Прагматичний напрямок через свою конкретність найбільш складна і найменш вивчена галузь теорії інформації. Про це свідчить той факт, що значних загальних теоретичних результатів тут немає.

Основною метою прагматичних досліджень є оптимізація інформаційних систем. Це завдання має поки що постановочний характер і не може бути вирішене без розв'язання ряду інших складних проблем, найважливіша з яких - кількісне вимірювання цінності інформації.

Прагматичний аспект інформації у загальному випадку залежить від суб'єкта, об'єкта і мети. Тому цінність інформації можна класифікувати залежно від споживача інформації (суб'єкта), самої інформації (об'єкта), Цілей що ставляться під час використання інформації.

Донедавна прагматичний аспект поняття "інформація" мало обговорювався і майже не досліджувався. Це обумовлено тим, що теорія інформації була пов'язана тільки з технікою зв'язку, проблему цінності інформації можна було ігнорувати, оскільки канал зв'язку мав однаково добре працювати

незалежно від конкретного змісту інформації що підлягає передаванню цим каналом. І хоча питання про цінність інформації піднімалося ще в перших розробках, по-справжньому актуальним воно стало з розвитком прикладної теорії інформації та проникненням її ідей у різні галузі. Залежно від характеру розв'язуваних завдань поняття цінності інформації розглядалося в різних напрямках. Розвиток теорії цінності інформації був важливим кроком, завдяки якому у формалізм теорії інформації внесено конкретний фізичний і технічний зміст, що відбиває реальне призначення систем.

Характеристики цінності інформації можуть ефективно застосовуватися під час виконання таких завдань.

1.Ціннісно-динамічна фільтрація поточної інформації в системах управління.

Для функціонування системи управління в загальному випадку потрібна поточна інформація про значення вихідних величин, зовнішніх збурень і змінних параметрів системи, що надходить від датчиків і фільтрів. Для зручності викладу називатимемо цю інформацію просто поточною. Своєчасність обробки поточної інформації є необхідною умовою успішної роботи системи управління. Однак, оскільки системи обробки інформації та виконуючих пристроїв управління мають обмежену пропускну здатність, існує потреба виділити з усього потоку поточної інформації ту частину, що становить найбільшу цінність. Саме від того, наскільки вдасться перебороти складність інформаційного забезпечення управління, зокрема відокремити головні відомості від другорядних (малоцінних і застарілих) і тим самим мінімізувати потоки даних, значною мірою залежить якість управління. Тому в системах управління попередньо має вирішуватися завдання добору потрібної інформації в невпорядкованому потоці. Як критерії добору необхідно використовувати цінність інформації та її старіння.

Таким чином, завдання ціннісно-динамічної фільтрації поточної інформації в системах управління полягає, по-перше, у мінімізації потоків даних для подальшої обробки; по-друге, в оптимізації зміни керівних впливів.

Тобто після опитування датчиків і фільтрів на подальшу обробку для вироблення керівних рішень відправляється тільки та інформації, що має більшу цінність, ніж раніше задане граничне значення. Іншими словами, на обробку інформації, яка надійшла, і вироблення керівних впливів ресурси будуть витрачені тільки в тому випадку, якщо вплив цієї інформації на ефективність функціонування системи перевищить певний поріг. При цьому необхідно, щоб економія ресурсів на обробку первинної інформації та вироблення керівних впливів перевищували втрати ефективності системі через незбереження попереднього керівного впливу. Ц. втрати можуть заповнюватися на наступному кроці управління, якщо цінність інформації перевищить деякий поріг, необхідний для зміни керівних впливів

Технічними прикладами ціннісно-динамічної фільтрації є управління ракетами, літаками, багатотоннажними суднами, у яких зміна керівних впливів призводить до великих витрат ресурсів.

2. Ціннісно-динамічне ранжирування поточної інформації а системах управління.

Це одне із завдань управління спостереженнями, а саме - установлення черговості обробки повідомлень, що надходять, у режимі реального часу відповідно до характеристик цінності й старіння цих повідомлень. Реалізація такого завдання здійснюється шляхом включення в контур системи управління системи масового обслуговування з ціннісно-динамічною дисципліною вибору вимоги на обробку. При цьому оптимізується обробка поточної інформації за цінністю з обмеженнями щодо старіння. Цю процедуру можна застосовувати як наступний етап після ціннісно-динамічної фільтрації для ранжирування перед обробкою відфільтрованої за цінністю і старінням інформації.

Таке завдання виникає, наприклад, під час розробки автоматизованих диспетчерських систем великих аеропортів, що забезпечують безпеку польотів в умовах одночасного перебування в повітряному просторі над аеродромом декількох літаків.

3. Організація раціональної обробки інформації в інформаційно-обчислювальних комплексах (ЮК).

Ефективність ЮК істотно залежить від методів організації обробки інформаційних потоків. Основу цих методів становлять дисципліна і критерій або система критеріїв оцінки різних варіантів організації обробки. Найбільш ефективними у цьому випадку є дисципліна і критерії оцінки, що враховують характеристики цінності й старіння інформації.

4. Оптимізація одержання оцінок координат процесу, що використовуються для цілей управління.

Таке завдання виникає у випадках, коли витрати на одержання інформації порівнянні з її цінністю. її постановка стає актуальною через ускладнення датчиків первинної інформації та систем обробки цієї інформації. Під час вирішення цього завдання цільова функція, що оптимізується, разом із традиційними складовими, такими як точність оцінок координат і витрати на керування, містить члени, що враховують витрати на спостереження й обробку інформації.

Розглянуті завдання не вичерпують усіх можливих сфер застосування характеристик цінності інформації і є лише найбільш типовими.

3.3 Аналіз відомих методів реплікації та синхронізації баз даних

Для організації реплікації пропонується велика кількість методів синхронізації. Кожен з них має свої переваги і недоліки і, отже, сфера застосування. В даний час широко використовують наступні методи синхронізації даних в реплікуємих БД: по журналу транзакцій; по поточному стану таблиць; з використанням незалежних наборів таблиць. Розглянемо ці методи детальніше.

Синхронізація по журналу транзакцій [10]. Цей метод застосовується найчастіше. Результати операцій, що проводяться в БД (включення, видалення і зміна записів), зберігаються в деякому спеціально

організованому журналі (звичайно це окрема таблиця або набір таблиць в самій БД, хоча журнал може бути будь-яким). При виконанні транзакції в журнал поміщаються дані про здійснювані операції так, щоб в подальшому можна було відновити БД і повторити виконання операцій. Звичайне заповнення журналу проводиться тригерами, які пов'язані з таблицями, що беруть участь в реплікації. В разі включення або оновлення записів таблиці в журналі зберігаються значення полів таблиці, що змінилися, в разі видалення записів – лише ідентифікатори видалених записів. У журналі зберігаються також відмітка часу, коли було проведено зміну, і ідентифікатор користувача, який ініціював зміну даних. Інколи туди включається ідентифікатор поточного сервера БД, використовуваний для ідентифікації бази даних, в якій сталася зміна.

Переваги методу: немає необхідності в установці прямого з'єднання; можливість виконати відкіт бази даних на будь-який момент часу; порівняльна простота реалізації; висока швидкість синхронізації (вона назад пропорційна кількості змін в циклі синхронізації). Недоліки методу: великий об'єм даних, що зберігаються (він пропорційний кількості змін в циклі синхронізації); велике число колізій (можливих протиріч).

Синхронізація по поточному стану [10]. Синхронізація по поточному стану завжди виконується зі встановленням з'єднання. Тому застосувати цей метод без наявності якого-небудь каналу зв'язку між серверами не можна. Процес, що виконує синхронізацію, встановлює з'єднання з обома базами даних (якщо число баз даних більше двох, то синхронізація виконується попарно, – наприклад, "ланцюжком" або "зіркою") і сканує всі таблиці, що синхронізуються, у пошуках відмінностей. Процес, що управляє, "вирівнює" таблиці, наводячи усі бази даних до ідентичного вигляду. При цьому програмі реплікації доводиться приймати рішення: яка з баз даних містить вірніші дані (наприклад, чи слід видалити запис з першої БД або, навпаки, включити такий же запис в другу базу даних, де її доки немає). Цей метод має істотний недолік: тут швидкість синхронізації пропорційна кількості

записів в тиражованих таблицях БД. Проте на відміну від синхронізації по журналу, коли відсутня яка-небудь додаткова інформація, процес, що в даному випадку управляє, може видавати допоміжні запити до баз даних, що синхронізуються, для того, щоб по можливості вирішити всі колізії (наприклад, взнати, чи є в інших таблицях записи, що посилаються на дану). Задавши певні правила, можна забезпечити автоматичне усунення великої частини колізій. Перевага методу: хороші можливості по усуненню колізій. Недоліки методу: низька швидкість синхронізації, оскільки вона назад пропорційна кількості всіх записів в реплікуємих таблицях; необхідність забезпечити з'єднання процесу, що управляє, з обома базами даних.

Синхронізація незалежних наборів таблиць [10]. Хай є декілька БД – A, B, C... Передбачимо, що зміна даних проходить тільки в БД A. Якщо в базах даних є незалежні (що не перетинаються) набори таблиць, то в одному такому наборі дані можуть змінюватися лише в БД A, а в іншому – лише в базі даних B. Така структура може вважатися системою з однобічною синхронізацією, що проводиться, декільком наборам таблиць. У такій схемі колізії практично виключені, оскільки потоки оновлень бази не перетинаються. Реалізовувати таку схему можна практично будь-яким способом без особливих проблем. Процес реплікації можна класифікувати також по наступних ознаках:

По напряму [10]. Якщо дані змінюються лише в одній з баз даних, а в іншій БД дані лише зберігаються і не піддаються змінам (репліки "лише для читання"), то така реплікація називається однонаправленою, або однобічною. Якщо ж дані можуть змінюватися і вводитися на всіх БД, то така реплікація іменується мультинаправленной, або багатобічною.

За часом проведення сеансу [10]. Якщо дані мають бути синхронізовані негайно після змін, то така реплікація називається реплікацією реального часу. Якщо ж процес реплікації запускається по якій-небудь події або в результаті дій адміністратора БД, то така реплікація називається відкладеною реплікацією.

За способом передачі інформації під час процесу реплікації [10]. Якщо з'єднання серверів, де зберігаються бази даних, відбувається за допомогою програми, яка гарантує стійкий канал зв'язку для прямої зміни реплікуємих баз даних, то такий вигляд синхронізації називається прямим. Якщо ж канал не є стійким під час процесу синхронізації і дані доводиться передавати в пакетному режимі, то таку синхронізацію називають недетермінованою, або імовірнісною. Як видно з огляду методів реплікації, теоретично може існувати велика кількість варіантів організації реплікації баз даних. Розглянемо детальніше метод реплікації по журналу. Спробуємо на практиці з'ясувати всі його переваги і недоліки. Нижче аналізуються розроблені програми, що моделюють таку реплікацію, і результати тестування реалізованої схеми. Зокрема, оцінюється зміна продуктивності процедур зміни даних при включеному режимі реплікації по журналу.

Процес реплікації по журналу [10]. Синхронізація по журналу – це найбільш простій і ефективний метод реплікації. Для синхронізації бази даних на сервері повторюють всі зміни, виконані на іншому сервері БД, витягуючи їх по черзі з журналу. Звідси витікає, що немає необхідності у встановленні з'єднання з БД під час передачі змін, оскільки журнал, виведений в окремий файл, можна передати по будь-яких каналах зв'язки у будь-який час. Журнал можна переносити, що в принципі недосяжно при використанні інших методів. Треба лише ретельно стежити за тим, які записи журналу ще не передавалися.

Ця проблема ускладнюється за наявності декількох що синхронізуються БД: потрібно запам'ятовувати, який останній запис журналу був переданий в кожну з БД Недоліком використання журналу є також і те, що він займає великий об'єм. При досить частій зміні таблиць баз даних і великих проміжках часу між сеансами реплікації журнал може перевершувати за розміром самі дані, що зберігаються в БД. Зрозуміло, що після успішного вживання журналу на іншому сервері БД цей журнал можна очистити.

Для синхронізації по журналу характерне збільшення об'єму даних, що зберігаються, пропорційно кількості змін в циклі синхронізації. В той же час швидкість синхронізації назад пропорційна кількості змін в циклі синхронізації. Слід ще згадати, що на основі цього журналу можна побудувати службу відкоту стану БД практично на будь-який момент (якщо лише відповідні дані не були видалені з журналу). Також зручно удаватися до журналу в цілях аудиту зміни станів об'єктів БД користувачами.

Синхронізація по журналу реалізується досить просто, але вочевидь, що колізії при цьому неминучі, оскільки процес, провідний реплікацію, не має одночасного доступу до двох баз даних.

3.4 Метод визначення цінності інформації з використанням функції чутливості

Нехай є припустимі інтервали значень критерію ефективності системи на кожному кроці керування. Значення критерію, що належать такому інтервалові і є оптимальними для системи, називатимемо номінальними. Збільшенням критерію на деякому кроці керування вважатимемо відхилення значення критерію на цьому кроці від номінального. Значення компонентів вектора виходу, вектора зовнішніх збурень і параметричної матриці системи, при яких значення критерію ефективності дорівнює номінальному, також вважатимемо номінальними. Збільшенням деякого компонента вектора виходу, вектора зовнішніх збурень або параметричної матриці системи стосовно їх номінального значення вважатимемо відхилення цього компонента від її номінального значення.

В основі методу - положення загальної теорії чутливості. Відповідно до цієї теорії збільшення критерію ефективності функціонування системи стосовно його номінального значення, яке можна отримати на деякому кроці керування, дорівнює сумі добутків функцій чутливості зазначеного критерію

до компонентів вектора виходу на збільшення відповідних компонентів вектора виходу.

Як цінність інформації про збільшення стосовно номінального значення деякого компонента вектора виходу береться збільшення критерію ефективності системи стосовно його номінального значення, викликане збільшенням зазначеного компонента. У свою чергу, це збільшення критерію ефективності дорівнює добуткові його функції чутливості до зазначеного компонента вектора виходу на збільшення цього компонента. Усі викладки, що стосуються вектора виходу, стосуються також вектора зовнішніх збурень і параметричної матриці системи.

Реалізація методу з використанням функцій чутливості можлива табличним і формульним способами. За табличного способу області припустимих значень кожного компонента вектора виходу розбиваються на деяку скінчену кількість у загальному випадку нерівних під-областей. Цінність інформації про збільшення компонентів вектора виходу щодо номінальних значень цих компонентів визначається на межах під-областей. При цьому цінність інформації визначається за допомогою функцій чутливості. У процесі керування в реальному часі таблиця цінностей інформації про збільшення компонентів вектора виходу системи зберігається в оперативній пам'яті керуючої ЕОМ. Значення з цієї таблиці вибираються табличним автоматом залежно від значень збільшень компонент вектора виходу при функціонуванні системи в реальному часі. При формульному способі реалізації методу цінність поточної інформації визначається на кожному кроці для кожного компонента вектора виходу в режимі реального часу.

Відзначимо, що застосовувати табличний спосіб доцільно з таких причин:

- є можливість точного апріорного визначення значень цінності інформації;

- порівняно з формульним способом виконується менше обчислювальних операцій під час роботи в реальному часі.

Однак у разі великої кількості підобластей розбивки ємність оперативної пам'яті для табличного способу потрібна більша, ніж для формульного. Крім того, урахування параметра цінності інформації для підобласті деякого компонента вектора виходу, середнього від значень цінності на межах цієї підобласті, з'являються втрати точності за рахунок фактичної розбіжності для окремих значень компонента вектора виходу з однієї під-області

Для формульного способу необхідна значна швидкодія керуючої ЕОМ, оскільки обчислення значень цінності інформації, що надходить, має здійснюватися в реальному часі до прийняття рішень щодо керування об'єктом. При цьому можна визначити значення цінності інформації в деяких базових точках і її апроксимацію на наступних кроках керування не складною функціональною залежністю. Однак тут необхідно враховувати зниження точності обчислення значень цінності.

Таким чином, можна зробити висновок, що вибір табличного або формульного способу для реалізації методу визначення цінності поточної інформації з використанням функцій чутливості треба здійснювати з урахуванням вимог до точності і швидкодії системи керування об'єктом.

Розглянемо деякі положення загальної теорії чутливості. Нехай система керування описується диференціальними рівняннями в нормальній формі Коші

$$\dot{x} = a_{k1}x_1 + a_{k2}x_2 + K + a_{kn}x_n; k = \overline{1,n}\,, \qquad (3.2)$$

де $x_к$ - координати стану системи.

Позначимо через α_i, $(i = \overline{1,p})$ параметри системи, що змінюються в часі і входять у коефіцієнти рівняння (3.16). У загальному випадку параметри містять апріорну аналітичну або вимірну інформацію про систему, що враховується під час вибору вектора управління. Вхідні сигнали, коли вони є

відомими функціями, також можна вважати параметрами. Найбільш типовими параметрами в системах автоматичного управління є початкові умови, коефіцієнти підсилення, постійні часу, власні частоти, інтервал дискретності, помилки квантування і т. д.

З урахуванням параметрів а, рівняння (3.2) можна подати так [18]:

$$\dot{x}_k = \varphi_k(x_1, x_2, ..., x_n, \alpha_1, \alpha_2, K, \alpha); k = \overline{1, n}; \tag{3.3}$$

Розв'язок системи (3.3) $x_1(t), x_2(t), ..., x_n(t)$ визначає її вихідний рух. При невеликих змінах параметрів отримаємо:

$$x_k = \varphi'_k(x_1, x_2, ..., x_n, \alpha_1 + \Delta\alpha, \alpha_2 + \Delta\alpha, ..., \alpha_p + \Delta\alpha_p) \tag{3.4}$$

Розв'язок системи (3.4) $x'_1(t), x'_2(t), ..., x'_n(t)$ визначає її варійований рух.

Різниця

$$\Delta x_k(t) = x'_k(t) - x_k(t); k = \overline{1, n} \tag{3.5}$$

визначає додатковий рух системи. При цьому має місце таке співвідношення

$$\Delta x_k(t) = \frac{\partial x_k}{\partial \alpha_1}\Delta\alpha_1 + \frac{\partial x_k}{\partial \alpha_2}\Delta\alpha_2 + ... + \frac{\partial x_k}{\partial \alpha_2}\Delta\alpha_2 . \tag{3.6}$$

Позначимо

$$V_{ki}(t) = \frac{\partial x_k}{\partial \alpha_1}; k = \overline{1, n}; i = \overline{1, p}; \tag{3.7}$$

Тоді

$$\Delta x_k = V_{k1}(t)\Delta\alpha_1 + V_{k2}(t)\Delta\alpha_2 + ... + V_{kp}(t)\Delta\alpha_p; k = \overline{1, p}; \tag{3.8}$$

Величина $V_{ki}(t)$ є функцією чутливості k-го компонента вектора стану за i-м параметром системи.

Запропоновані методи доцільно застосовувати для визначення цінності поточної інформації, тобто в загальному випадку використання інформації про значення вихідних величин, зовнішніх збурень і змінних параметрів системи.

У працях К. Шеннона, що пов'язані з розробкою теорії інформації, не ставилось за мету визначення таких характеристик, як цінність і старіння інформації, але розробка саме теорії була викликана необхідністю кодування інформації, однією складовою (метою) цієї операції є захист інформації. Сучасний стан розвитку теорії захисту інформації дає змогу розглядати процеси обробки інформації з урахуванням її якісно-кількісних характеристик: цінності, старіння і захищеності - у нерозривній єдності.

Висновки до розділу 3

В даному розділі обґрунтовано визначення цінності поточної інформації та визначення цінності інформації в системах управління, здійснено аналіз відомих методів реплікації та синхронізації баз даних, для вирішення задач адаптовано метод визначення цінності інформації з використанням функції чутливості.

РОЗДІЛ 4
РЕАЛІЗАЦІЯ СИСТЕМИ ДЛЯ СИНХРОНІЗАЦІЇ БАЗ ДАНИХ КОРИСТУВАЧІВ ІЗ ВРАХУВАННЯМ ЯКІСНО-КІЛЬКІСНИХ ХАРАКТЕРИСТИК

4.1 Загальна структура системи

При роботі автоматизованої інформаційної системи торговельної мережі в режимі реального часу досить часто виникає необхідність реплікації даних у межах розподіленої бази даних. Потрібно відзначити, що на сьогоднішній день існує досить багато готових рішень, які являються в цілому працюючими. Але інформація, що передається між вузлами розподіленої системи, являється різнорідною та представляє собою різну цінність. Тому доцільно під час реплікації враховувати якісно-кількісні характеристики інформації, що передається для створення оптимальної черги транзакцій реплікації та виставленню пріоритетів для передаваємої інформації. В першу чергу необхідно враховувати такі характеристики інформації, як цінність, ціна та старіння.

Система розробляється на підставі виробничої необхідності, що виникла у роботі торговельної мережі. Основним призначенням є оптимізація проведення реплікацій між вузлами розподіленої бази даних автоматизованої інформаційної системи торговельної мережі.

Система повинна включати наступні функції:

- Визначати інтерфейси передачі даних для різних типів вузлів розподіленої бази.
- Визначати інтерфейси для якісно-кількісних характеристик передаваємої інформації.
- Надавати можливість гнучкого налаштування системних параметрів для оптимальної послідовності передаваємих транзакцій.
- Ведення історії здійснених реплікацій, зберігання усієї необхідної

інформації для подальшого аналізу.

- Можливість довільного вибору баз даних, між якими буде здійснюватися реплікація.
- Можливість реплікації даних між різнотипними вузлами розподіленої бази даних.
- Моніторинг виконання реплікацій в певний момент часу.

Для реалізації функцій системи необхідно реалізувати наступні модулі:

1. *Налаштування типів джерел даних* – визначення переліку СУБД, для яких буде застосовуватися реплікація. В даному модулі необхідно реалізувати зберігання наступної інформації:

- назва типу джерела даних;
- опис типу джерела даних;

2. *Налаштування джерел даних* – визначення типової структури даних для кожного типу СУБД. Наприклад, для СУБД Oracle необхідно вказати такі дані:

- мережеву адресу (ip-адресу чи мережеве ім’я сервера);
- порт для підключення до БД;
- протокол обміну даними;
- ім’я екземпляру БД;

3. Для кожного вузла розподіленої бази даних, що бере участь у реплікації, необхідно визначити структуру даних, що будуть брати участь у реплікації, питання авторизації до видаленої СУБД. Для цього у розроблюваній системі необхідно реалізувати модуль *Налаштування структур даних для вузла РБД.* В даному модулі необхідно зберігати інформацію:

- назва типу вузла;
- назва джерела даних;
- відомості про таблиці та загалом об’єкти БД, з якими буде здійснюватися реплікація;
- відомості про якісно-кількісні характеристики елементів БД для

проведення оптимальної реплікації.

4. Для забезпечення механізму оцінки якісно-кількісних характеристик інформації необхідно реалізувати *модуль визначення якісно-кількісних характеристик* інформації. Також до стандартного функціоналу системи необхідно додати такі засоби, як розклад проведень реплікацій.

Система реплікацій з урахуванням якісно-кількісних характеристик інформації не зберігає значний масив даних, тому налаштування доцільно зберігати у вигляді xml-файлів.

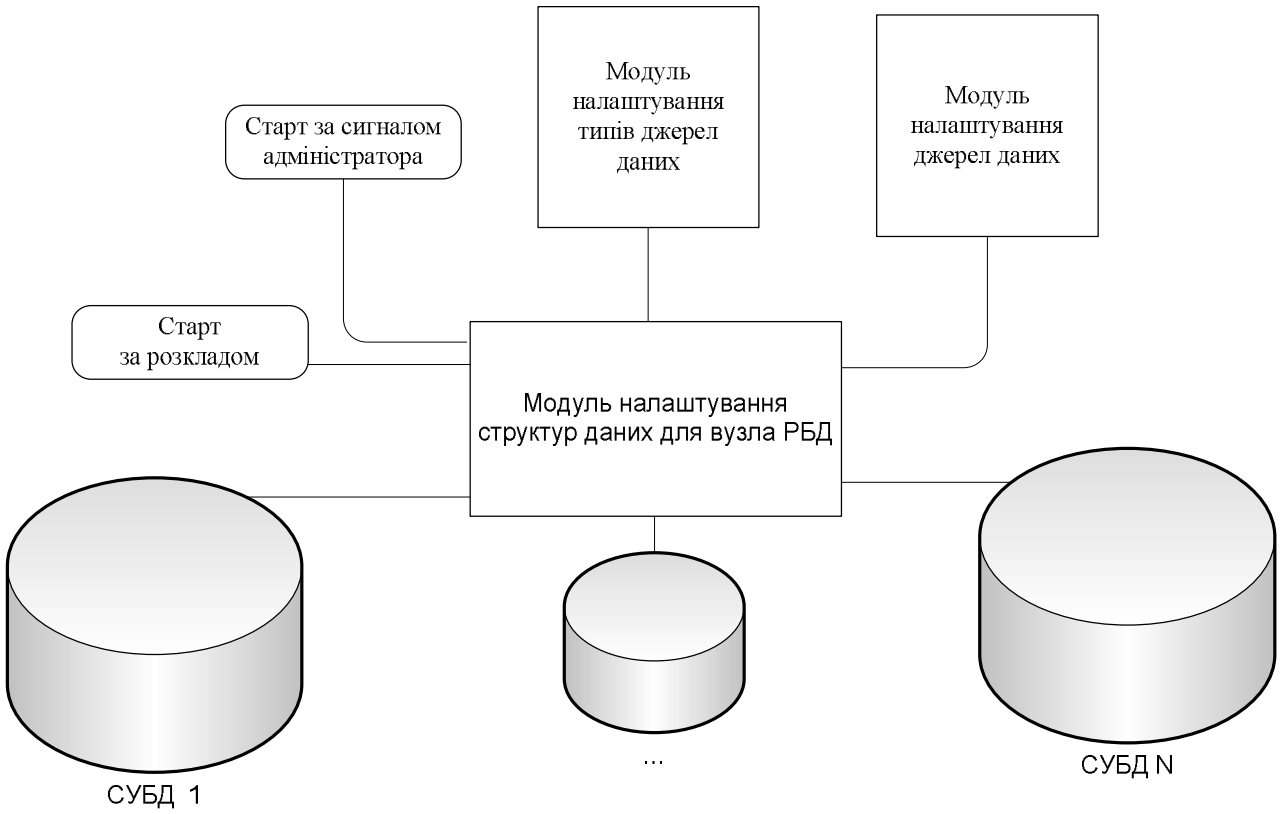


Рисунок 4.1. Схема інформаційних взаємозв'язків модулів програми

4.2 Ефективність методів визначення цінності та старіння інформації

Розглянемо принципи визначення цінності інформації. Цінність інформації слід визначати у зв'язку з метою, для якої вона використовується, а також вважати змінною в часі. Справді, якщо генерація інформації відбулася

задовго до моменту її використання, то можна вважати, що в момент часу *1*: цінність її менша максимальної. З іншого боку, коли обробляти і використовувати її вже пізно, цінність стає нульовою, тобто враховувати таку інформацію при управлінні об'єктом вже не можна. Отже, між моментом генерації інформації (повідомлення) і моментом, коли вона втрачає свою цінність, має існувати момент, коли вона здобуває максимум цінності.

Таким чином, виникають три питання:

- Який максимум інформації в абсолютних або відносних одиницях?
- Де він розташований на осі часу?
- Як змінюється цінність інформації по обидва боки від нього?

Значення максимуму цінності інформації можна визначити через збиток, що несе об'єкт, якщо інформація взагалі не буде оброблена і використана. Він може бути виражений як в абсолютних, так і у відносних одиницях. При цьому можна скористатися математичними моделями об'єктів і методами експертних оцінок. Сполучення обох методів дає досить достовірні дані при правильно побудованих шкалах і процедурах опитування. Для дуже широкого класу об'єктів може бути знайдена імовірність досягнення мети залежно від якого-небудь фактора (зокрема, від часу використання інформації, Рисунок 4.2) і побудованого інтегрального закону розподілу *F(t)* - імовірності досягнення мети залежно від моменту використання інформації.

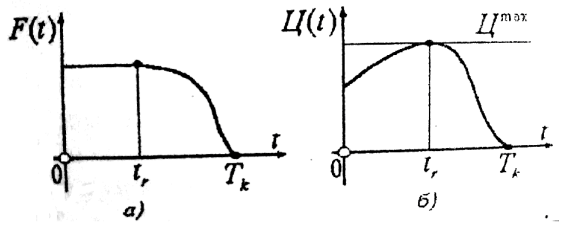


Рисунок 4.2. Криві: а) залежностей інтегрального закону розподілу імовірності досягнення мети; б) зміни цінності інформації в часі

$$\begin{cases} 1, при\_t < t_k \\ P(t), при\_t_k < t < T_k \end{cases} \qquad (4.1)$$

де t_k — момент максимуму цінності повідомлення; $T_к$ - момент часу, коли цінність повідомлення стає рівною нулеві;

$$P(t) = I_ф(t)/I_n; \quad (4.2)$$

$I_ф(t)$ і I_n - фактичне значення критерію ефективності системи при використанні інформації в момент часу t і планове, відповідно.

Відзначимо, що формула правдива лише для випадку максимізації критерію ефективності системи. Однак у випадку його мінімізації можна також одержати аналогічну залежність.

У момент часу t - 0 повідомлення генерується об'єктом. До моменту $t = t_k$ імовірність того, що повідомлення буде оброблене й використане та об'єкт від його запізнення не зазнає втрат, близька до одиниці. З моменту часу $t = t_{ki}$ імовірність досягнення мети починає падати і до моменту T_k практично дорівнює нулеві. Логічно припустити, що в останній момент $t = t_k$, коли імовірність досягнення мети об'єктом ще залишається близькою до одиниці, цінність повідомлення здобуває максимум, а до моменту $T_к$ вона стає рівною нулеві. Природно вважати, що на інтервалі $[t_k,T]$ цінність повідомлення падає за тим самим законом, що й імовірність досягнення мети об'єктом.

Таким чином, ми визначили максимум цінності повідомлення і закон її спадання на інтервалі $[t_k,T_к]$. Залишається з'ясувати, як змінюється $Ц(t)$ на інтервалі $[0,t_k]$. Припустити строгу формалізацію важко, однак правдоподібне розуміння, що цінність повідомлення має зростати від моменту t = 0 до моменту $t = t_к$. У першому наближенні можна прийняти, що оскільки $Ц(t)$ падає від максимуму до нуля за час $[t_к, T_к]$, то й на відрізку $[0,T_k]$ вона повинна змінюватися пропорційно відношенню цих відрізків часу.

Зміну цінності інформації в часі можна визначити за формулою[20]:

$$Ц(t) = \begin{cases} Ц^{\max} \dfrac{T_k - t_k}{T_k - t}, при\_0 \le t \le t_k \\ Ц^{\max} F(t), при\_t_k < t \le T_k \end{cases} . \qquad (4.3)$$

Формула (4.2), як і формула (4.3), дає лише принцип визначення цінності інформації. Проте абсолютно зрозуміло, що, реалізуючи обговорені вище моделі об'єктів, можна визначити цінність різних класів інформації. Підкреслимо, моделі повинні дозволяти визначення *Цmax, t_k , T_k max F(t)* .

Розглянемо питання взаємозв'язку старіння і цінності інформації, Рисунок 4.2.

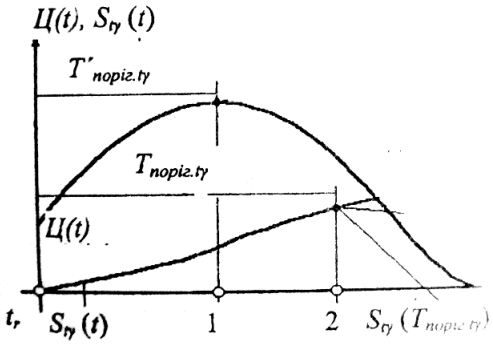


Рисунок 4.2. Взаємозв'язок цінності і старіння інформації під час обробки й використанні з визначеною метою

Взаємопов’язаність цінності інформації та її старіння полягає у встановленню вимог на раціональну організацію обробки і використання інформації. На рисунку 3.4 показано, що точка 1 на часовій осі є найбільш раціональною для реалізації мети використання. Це точка з максимальною цінністю інформації, і саме до цього моменту слід виконати всі необхідні операції інформаційного процесу (передача, обробка й ін.).

4.3 Тестування методів розв’язання задач інформаційної системи

При реалізації інформаційної системи реплікацій з урахуванням якісно-кількісних характеристик інформації необхідно визначитися з методом та алгоритмом проведення реплікації, описаним у пункті один другого розділу магістерської роботи. Для цього необхідно визначити такі параметри проведення реплікації, як:

1. Принципи встановлення з'єднання при реплікації. На мою думку, доцільно використовувати системи зі встановленням з'єднання, але дещо допрацьовану. А саме – черга передаваємих повідомлень має формуватися з урахуванням якісно-кількісних параметрів інформації. Такий принцип встановлення з’єднання надає наступні переваги:

- Значне скорочення витрат на зберігання службової інформації. При такому підході не відбувається накопичення даних про зміни, оскільки завжди порівнюються існуючі версії даних. Відзначимо, що ця перевага сходить нанівець, якщо синхронізуються таблиці, для яких однією з вимог є ведення історії змін.
- Не потрібна модифікація існуючої бази даних, принаймні, не змінюються існуючі таблиці.
- Не потрібна постійна присутність адміністратора бази даних і контроль системи накопичення змін. Якщо проблеми і виникають, то тільки в строго визначені моменти, при цьому відновлення системи (приведення системи в синхронний стан) відбувається швидко і без особливих витрат.

2. Способи виявлення змін. На мою думку, доцільно використовувати проміжний рівень доступу до бази даних, який по суті являється додатковим програмним прошарком між рівнем логіки управління і рівнем доступу до бази даних. Таким чином, разом із записом змін у відстежувану таблицю, цей проміжний рівень здійснює запис і в чергу змін. На рисунку 4.2 показана одна з можливих реалізацій такого методу.

Переваги:

– Не вимагається внесення змін до існуючої схеми бази даних, а такі умови часто ставляться перед розробниками.

– Має технологічну можливість формування черги змін із урахуванням якісно-кількісних характеристик інформації.

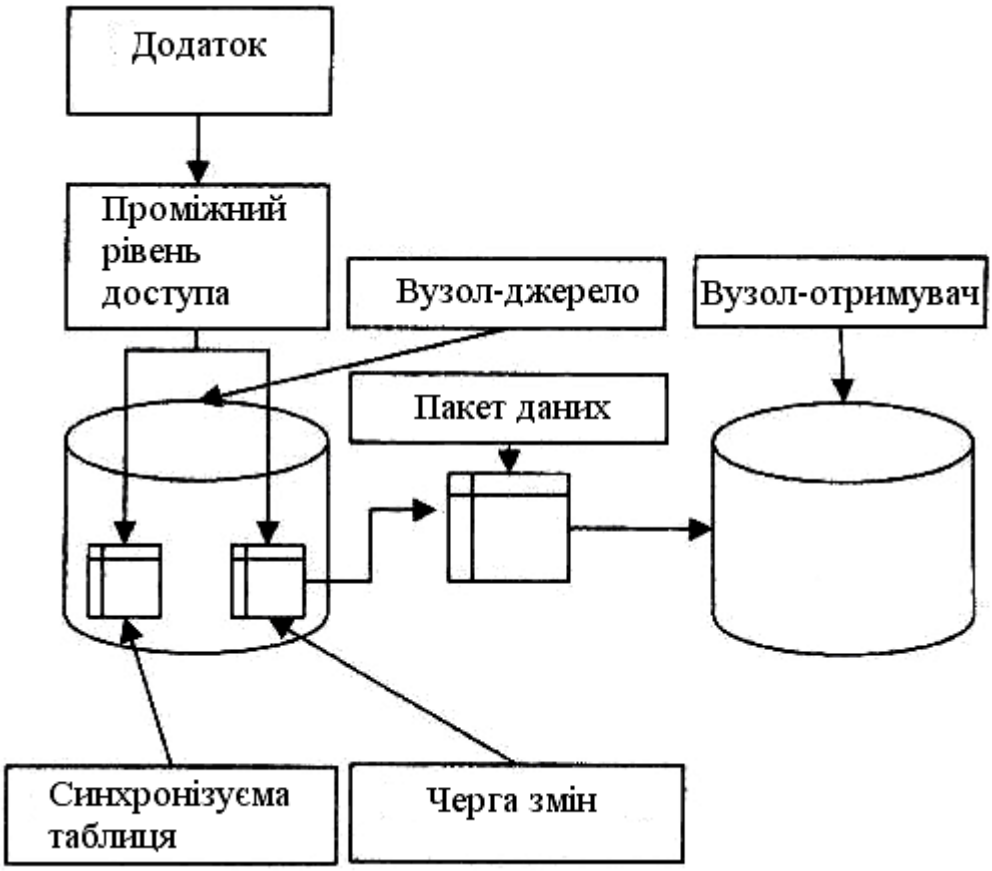


Рисунок 4.3. Принцип роботи системи з проміжним рівнем доступу до бази даних

Недоліки:

– Можуть потрібно значні витрати при впровадженні проміжного рівня доступу до бази даних, якщо навіть початкове застосування було правильно спроектоване.

– Не відстежуються зміни, вироблювані програмними засобами, що не використовують цей проміжний рівень доступу. Необхідність в цьому виникає, коли, наприклад, довідкова інформація міняється як з основного застосування, так і з безлічі допоміжних програмних засобів.

При формуванні черги змін, показаної на рисунку 4.2, необхідно враховувати якісно-кількісні характеристики інформації, а саме: цінність, ціна та старіння інформації. Найбільш важливою з них, на мою думку, являється цінність інформації.

Цінність і старіння інформації оцінюється з різних позицій, вкладаючи при цьому різні поняття в їх визначення. Часто справедливо підкреслюють, що цінність інформації нерозривно пов'язана з її старінням, проте облік цих факторів здійснюється зовсім по-різному. Закон зміни старіння інформації інтерпретується як закон зміни цінності інформації в часі. Цим і визначається взаємозв'язок характеристик цінності і старіння. Підкреслимо істотний момент: взаємозв'язок цінності і старіння інформації в даному випадку визначено, але при цьому фактично зникають цінність і старіння інформації як самостійні і не залежні одне від одного поняття. Даний підхід до визначення цінності і старіння інформації та їх взаємозв'язку суперечить сформульованим законам інформації. Відповідно до цих законів цінність інформації визначається залежно від мети її одержання, а старіння - як рівень відхилення матеріального явища від інформації, що відображає це матеріальне явище. Отже, цінність однієї й тієї ж інформації для різних цілей використання буде різна, а характеристика "відходу" інформації від свого первісного (відображуваного) базису, тобто старіння, не залежить від цілей використання інформації, хоча під час її обробки з різними цілями кількісно старіння інформації може враховуватися по-різному. Це вказує на те, що визначення цінності і старіння інформації та їх взаємозв'язку відповідно до першого підходу ґрунтується на фактично "поєднаних" поняттях несправедливо Треба не поєднувати поняття цінності і старіння інформації, а розглядати їх як самостійні характеристики. Однак він не встановлює що взаємозв'язок цінності інформації та її старіння визначається як рівень відходу матеріального явища від свого первісного значення.

У момент часу t_1, матеріальне явище мало значення M_t, що визначене (зареєстроване) з певною точністю. З часом t матеріальне явище M змінює своє значення і в моменти часу t_2, t_3, $t_4...t_n$ набуває відповідно значення M_2, M_3, $M_4,...,M_n$. Якщо використовувати інформацію про значення матеріального явища M_x в момент часу t_2, то відхід матеріального явища за час (t_2 -t_1,) становитиме M_2-M_1. За час $(t_3 - t_1)$ і (t_4-t_1) відхід матеріального явища від зареєстрованого в момент часу t_1, значення M_1, становитиме, відповідно, M_3-M_1, і M_4-M_1. Узявши момент часу t_1, за момент генерації інформації можна формально визначити старіння інформації, отриманої в момент часу t_1, у різні моменти часу.

$$S_i=M(t)-M(t_i) \tag{4.4}$$

де $Si(t)$- старіння інформації про матеріальне явище, зареєстроване в момент часу t_i.

Такий важливий показник як цінність інформації, на мою думку, у реалізуємій системі слід визначати з використанням імітаційних моделей.

Нехай система керування ріпліками описується лінійними різницевими рівняннями у векторно-матричному записі:

$$\begin{cases} \mathrm{X}_{k+1} = \Theta_{xk} + \Psi_{uk} + \Omega_{fk}, \\ \mathrm{Y}_k = \tilde{A}x_k + Eh_k; \end{cases} \tag{4.5}$$

де $x_k = [x_{k1}, x_{k2}, x_{k3},..., x_{kn}]$ – вектор стану; $u_k = [u_{k1}, u_{k2}, u_{k3},..., u_{kn}]$ – вектор керування; $f_k = [f_{k1}, f_{k2}, f_{k3},..., f_{kn}]$ – вектор зовнішніх збурень; $h_k = [h_{k1}, h_{k2}, h_{k3},..., h_{kn}]$ – вектор перешкод; $y_k = [y_{k1}, y_{k2}, y_{k3},..., y_{kn}]$ – вектор виходу; Θ - власна параметрична матриця системи; Ψ - вхідна матриця системи; Ω – матриця координат зовнішніх збурень; Γ – матриця виходу; E – матриця координат перешкод; $t_k=k\Delta t$; Δt – величина кроку дискретизації; k – номер моменту часу $k=\overline{1,K}$; $t_0=0$, $t_k=K$; X_{ki}, U_{ki}, F_{ki}, H_{ki}, Y_{ki} – області

припустимих значень і-х компонентів відповідних векторів у *k*-й момент часу.

Областями припустимих значень векторів x_k і y_k будемо вважати області, при яких функціонування системи з погляду досягнення поставленої мети ще можливе.

Якість функціонування системи характеризується критерієм ефективності:

$$I_k = \rho(x_k, u_k),\ k=\overline{1,K}, \qquad (4.6)$$

де k – номер моменту часу, у який здійснюється деякий крок керування системою.

У якості технології програмування вибрано фреймворк Microsoft .NET Framework 3.0. Один з ключових елементів архітектури Microsoft .NET - середовище .NET Framework. Вона відповідає за реалізацію нового режиму виконання (а значить, і розробки) програм на локальному комп'ютері. Йдеться про додаткове спільне середовище між програмами і ОС, з одного боку, і між програмними компонентами однієї програми - з іншого.

Структура .NET Framework представлена на рисунку 4.3. Як видно, йдеться фактично про єдине середовище виконання програм і підтримки їх розробки. Саме тут зібрані базові класи для всіх мов програмування, реалізовані у вигляді бібліотеки ядра System, а також великого числа (більше 20) спеціалізованих бібліотек з іменами System.Data, System.XML і так далі Над ними розташовується набір засобів формування виконуваних модулів різного типа (знову ж таки єдиний для різних мов).

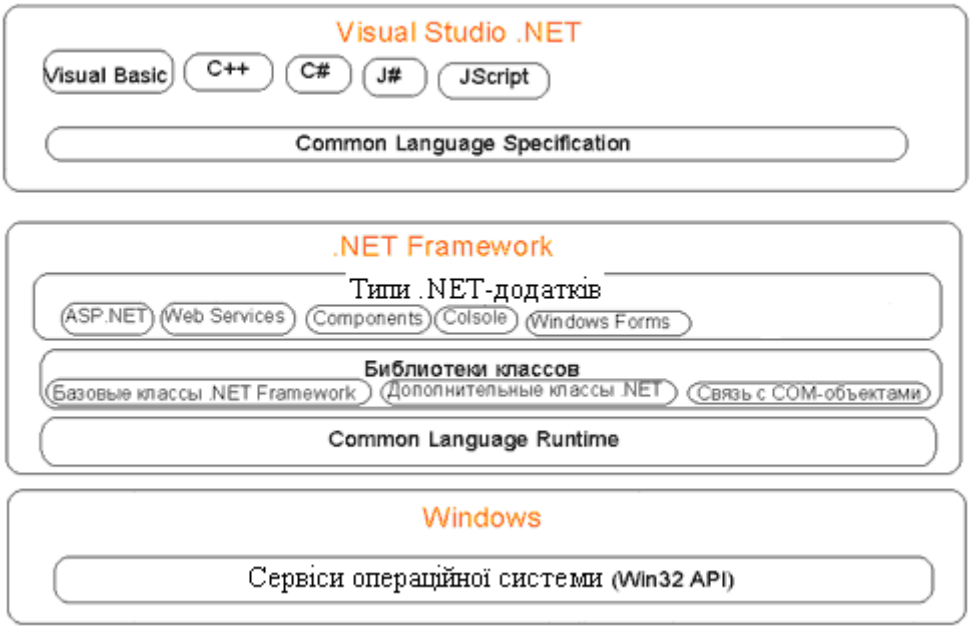


Рисунок 4.4. Структурна схема .NET Framework.

Власне, з точки зору розробки програм саме в цьому полягає новизна (але лише відносна, для Windows-програмування) даного рішення: допоміжні функції відокремлені від бізнес-логіки програм, що реалізовується за допомогою конкретних мов. Окрім очевидної переваги такої уніфікації функцій (навіщо писати окремі функції, що обчислюють синус, для різних мов?), це створює хороші передумови для поліпшення управління оперативною пам'яттю. Адже, як відомо, величезне число проблем надійності програм зв'язане з використанням різних механізмів динамічного розподілу простору в різних мовах.

Зверху розташовуються самі інструменти розробки, представлені в даному випадку системою Visual Studio .NET, у якій кожна з мов взаємодіє з .NET Framework через загальний мовний інтерфейс. З одного боку, такий механізм дозволяє досить просто підключати до даного середовища різні мови. В даний час про створення таких засобів (COBOL, FORTRAN, Perl і т. п.) оголосили вже більше 20 незалежних розробників. В той же час майже у всіх цих інструментів є альтернативні варіанти середовища, які працюють зовні VS.NET і безпосередньо взаємодіють з .NET Framework.

З іншого боку, така уніфікація автоматично нівелює функціональні можливості різних мов, в значній мірі зводячи проблеми вибору конкретного інструменту до прихильності конкретних людей тому або іншому синтаксису. Це сьогодні особливо добре видно на прикладі VB.NET і C#. Оскільки, кажучи про платформу .NET, нам ніяк не уникнути порівняння її з Java 2 Platform, то відмітимо, що відмінність між ними в тому, що в першому випадку є різні синтаксичні форми для єдиної .NET-мови. У якій мірі виправдана відмова від специфіки мов (яка виявлялася в різній їх ефективності для вирішення різних наочних завдань.

Вище ми говорили про компоненти NET Framework, що відносяться до процесу розробки програм. CLR (Common Language Runtime, загальна для мов середовище виконання) - це наріжний камінь у фундаменті організації обчислювальних процесів всієї концепції .NET. Саме тут повинні вирішуватися основні завдання підвищення надійності і безпеки програм, а також платформенній незалежності. CLR - це окрема велика тема, тому ми розглянемо її лише тезовий.

Всі виконувані модулі .NET-додатків (називатимемо їх CLR-модулями) реалізуються не у вигляді машинного коди (native, "рідного" для даного комп'ютера), а за допомогою так званих байт-кодів по специфікаціях проміжної мови Microsoft Intermediate Language (MSIL). Іншими словами, кожен сумісний з .NET-компілятор повинен перетворити вихідний код на мові високого рівня в двійковий MSIL-код, який вже потім виконуватиметься в середовищі CLR. Ідея і реалізація подібного підходу зовсім не нові, для Basic і Virtual Pascal це робилося ще в 1970-х, а про Java відомо і нинішньому поколінню розробників.

Проте, на відміну від Java, CLR виконуватиме код не в режимі класичного інтерпретатора, а шляхом попередньої компіляції в машинний код окремих фрагментів програми або цілого застосування (Рисунок 4.4). Перший варіант - основний, при цьому застосовується так званий компілятор Just-In-Time, який виконує перетворення MSIL в машинний код у міру

звернення до відповідних процедур (тобто невживані фрагменти програми зовсім не компілюються). Даний підхід також добре відомий.

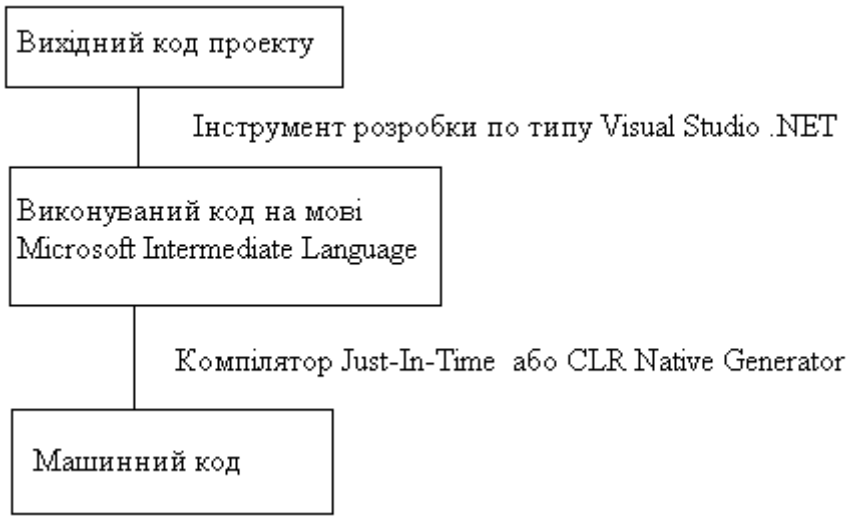


Рисунок4.5. Схема компіляції .NET-додатків.

Як відомо, режим інтерпретації має дві головні переваги в порівнянні з використанням машинних кодів: підвищення безпеки програм (точніше, захищеності системи в цілому від дії конкретних програм) і спрощення адаптації програм до конкретної апаратної платформи. З врахуванням цього розглянемо структуру CLR-модулів.

CLR-модулі складаються з виконуваної коди і метаданих. Метадані (наприклад, різні декларації полів, методів, властивостей і подій) широко застосовуються і в COM-технології, що і складає її основну відмінність від звичайних двійкових DLL. У CLR склад метаданих значно розширений, що дозволяє ефективніше контролювати версії, перевіряти надійність джерел вступу програм і тому подібне

Виконуваний код в основному представлений у вигляді "керованого коду" (можливі і фрагменти "некерованого коду", але вони будуть великою рідкістю). Це означає, що CLR не просто перетворить MSIL в машинні інструкції, а виконує ці дії з врахуванням певних зовнішніх установок. Наприклад, Модуль1 може задати свій власний набір прав, що надається що

викликається їм Модулю2, заборонивши, зокрема, будь-які операції зміни файлів.

Ця можливість широко використовується в розрахованій на багато користувачів інтернет-грі для програмістів "Тераріум", анонсованою на форумі PDC 2011. У ній кожен може написати програмні модулі, що реалізовують вибрані стратегії, які інші учасники завантажують на свої комп'ютери. Безпека тут забезпечується саме завдяки чіткому контролю за допустимими діями "чужорідних тіл". Загалом, в CLR ми бачимо реалізацію ідей інтернет-браузерів, які представляють проміжне середовище виконання програм, але лише із значно вищим рівнем керованості прав.

Що стосується платформеної незалежності, то CLR, здавалося б, має для цього всі передумови, оскільки передбачає наявність JIT-компілятора (як і в Java). Але я не розділяю оптимізму деяких експертів, які говорять про можливість появи найближчим часом подібних засобів, наприклад, для Linux. По-перше, CLR спочатку досить істотно задіює базові служби Windows. По-друге, Microsoft абсолютно інакше, ніж Java-співтовариство, трактує поняття мультиплатформеності: JIT-компілятори з'являться для різних типів апаратури (кишенькові ПК, стільникові телефони і т. п.), але працювати вони будуть лише в середовищі Windows .NET.

Як середовище розробки інформаційної системи синхронізації з урахуванням якісно-кількісних характеристик інформації було обрано Microsoft Visual Studio 2008, мова програмування – C#.

4.4 Програмна реалізації системи синхронізації баз даних користувачів мультиплатформеного веб-сайту

Опишемо внутрішнню структуру системи за допомогою UML-діаграм. Клас (class) - категорія речей, які мають загальні атрибути і операції. Класи - це будівельні блоки будь-якої об'єктно-орієнтованої системи. Вони є описом сукупності об'єктів із загальними атрибутами, операціями, стосунками і

семантикою. При проектуванні об'єктно-орієнтованих систем діаграми класів обов'язкові.

Класи використовуються в процесі аналізу предметної області для складання словника предметної області системи, що розробляється. Це можуть бути як абстрактні поняття предметної області, так і класи, на які спирається розробка і які описують програмні або апаратні сутності.

Діаграма класів - це набір статичних, декларативних елементів моделі. Діаграми класів можуть застосовуватися і при прямому проектуванні, тобто в процесі розробки нової системи, і при зворотному проектуванні - описі існуючих і використовуваних систем. Інформація з діаграми класів безпосередньо відображується у вихідний код застосування - в більшості існуючих інструментів UML-моделювання можлива кодогенерація для певної мови програмування. Таким чином, діаграма класів - кінцевий результат проектування і відправна точка процесу розробки. Діаграма класів для системи «реплікація інформації з урахуванням якісно-кількісних характеристик» зображена на рисунку 4.6.

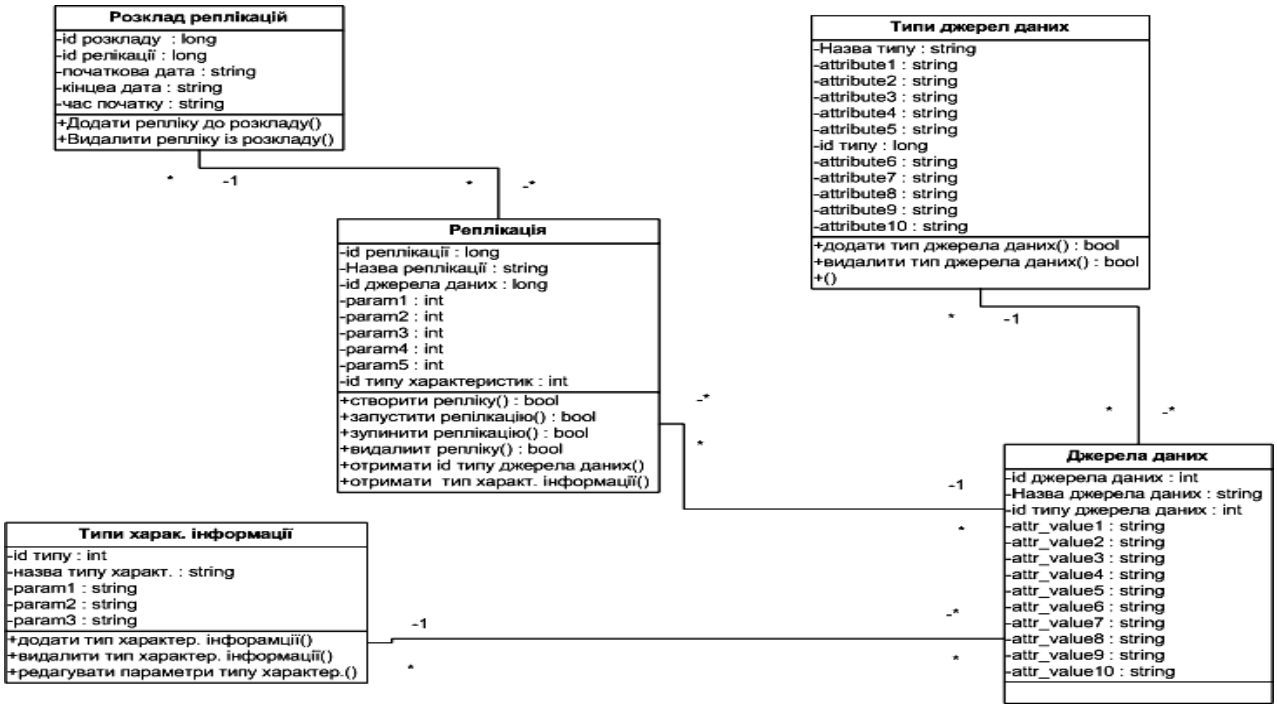


Рисунок 4.6. Діаграма класів

2. Діаграма послідовностей (sequence diagram) - відображує взаємодію об'єктів в динаміці. У UML взаємодія об'єктів розуміється як обмін інформацією між ними. При цьому інформація набирає вигляду повідомлень. Крім того, що повідомлення несе якусь інформацію, воно деяким чином також впливає на одержувача. Як бачимо, в цьому плані UML повністю відповідає основним принципам ООП, відповідно до яких інформаційна взаємодія між об'єктами зводиться до відправки і прийому повідомлень.

Діаграма послідовностей відноситься до діаграм взаємодії UML, що описують поведінкові аспекти системи, але розглядає взаємодію об'єктів в часі. Іншими словами, діаграма послідовностей відображує тимчасові особливості передачі і прийому повідомлень об'єктами.

Об'єкти позначаються прямокутниками з підкресленими іменами (аби відрізнити їх від класів), повідомлення (виклики методів) - лініями із стрілками, повертані результати - пунктирними лініями із стрілками. Прямокутники на вертикальних лініях під кожним з об'єктів показують "час життя" (фокус) об'єктів. Діаграма послідовностей для системи «реплікація інформації з урахуванням якісно-кількісних характеристик» зображена на рисунку 4.7

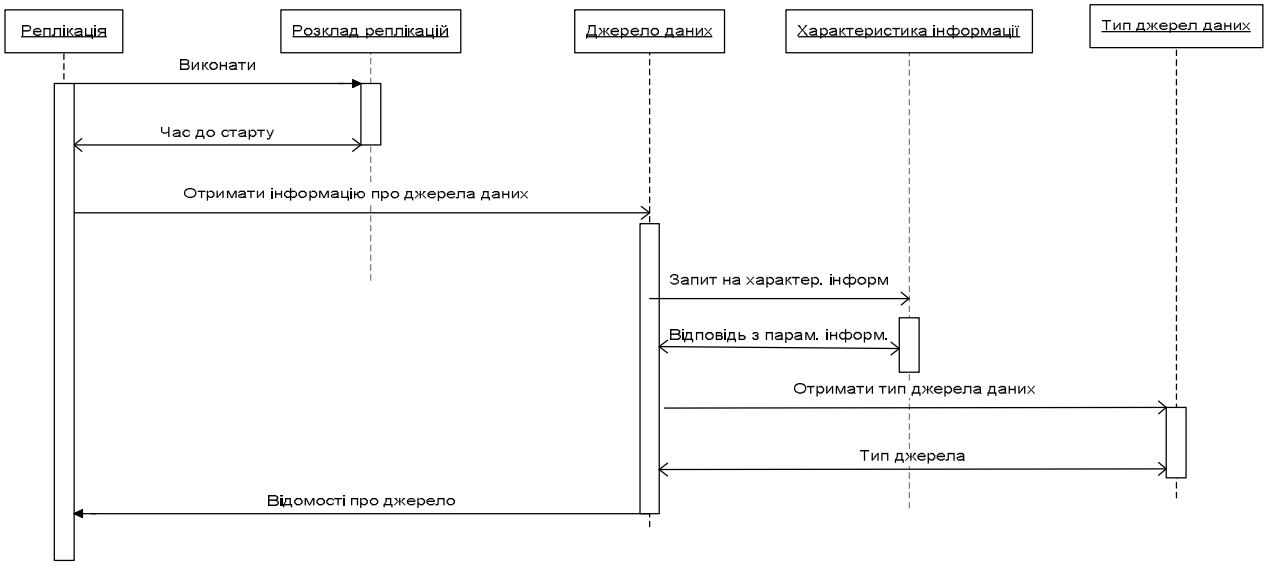


Рисунок 4.7. Діаграма послідовностей.

Діаграма взаємодії (кооперації, collaboration diagram). Діаграми послідовностей - це засіб документування поведінки системи, деталізації логіки сценаріїв використання; але є ще один спосіб - використовувати діаграми взаємодії. Діаграма взаємодії показує потік повідомлень між об'єктами системи і основні асоціації між ними і по суті, є альтернативою діаграми послідовностей. Діаграма об'єктів показує статику, деякий знімок системи, зв'язки між об'єктами в даний момент часу, діаграма ж взаємодії, як і діаграма послідовностей, показує взаємодія об'єктів в часі, тобто в динаміці.

Слід зазначити, що використання діаграми послідовностей або діаграми взаємодії - особистий вибір кожного проектувальника і залежить від індивідуального стилю проектування. Хоча у загальному випадку, наприклад, частіше використовується діаграма послідовностей.

Необхідність номера повідомлення пояснюється дуже просто - на відміну від діаграми послідовностей, час на діаграмі взаємодії не показується у вигляді окремого виміру. Тому послідовність передачі повідомлень можна вказати лише за допомогою їх нумерації. Діаграма взаємодії для системи «реплікація інформації з урахуванням якісно-кількісних характеристик» зображена на рисунку 4.8.

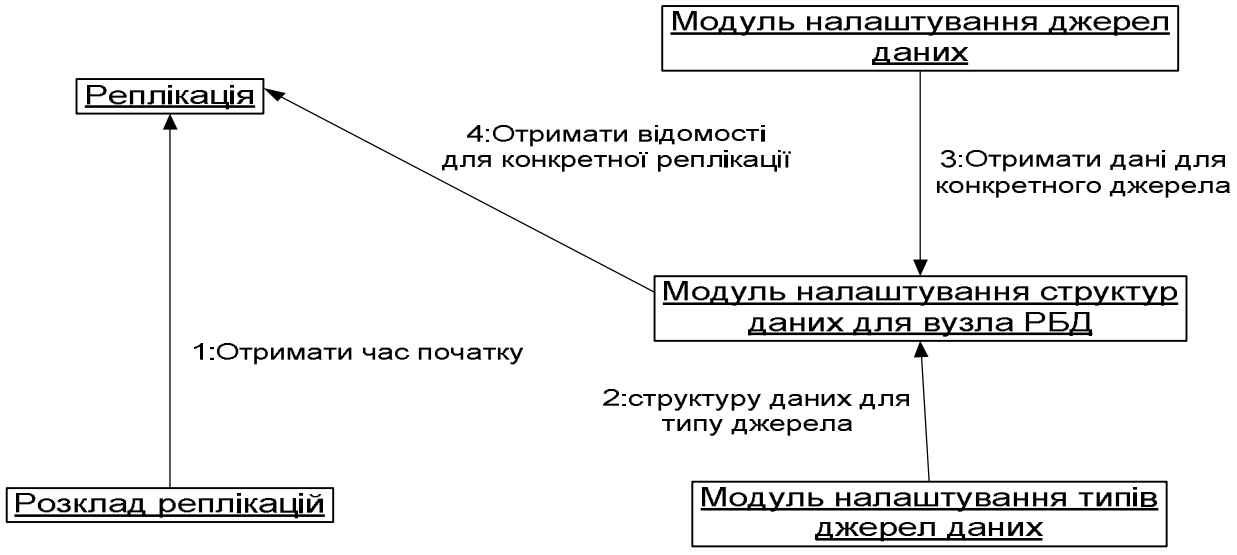


Рисунок 4.8. Діаграма взаємодії

4. Діаграма станів (statechart diagram). Об'єкти характеризуються поведінкою і станом, в якому знаходяться. Наприклад, людина може бути новонародженою, немовлям, дитям, підлітком або дорослим. Іншими словами, об'єкти щось роблять і щось "знають". Діаграми станів застосовуються для того, щоб пояснити, яким чином працюють складні об'єкти. Не дивлячись на те що сенс поняття "стану" інтуїтивно зрозумілий, все ж приведемо його визначення у такому вигляді, в якому його дають класики і Zicom Mentor:

Стан (state) - ситуація в життєвому циклі об'єкту, під час якої він задовольняє деякій умові, виконує певну діяльність або чекає якоїсь події. Стан об'єкту визначається значеннями деяких його атрибутів і присутністю або відсутністю зв'язків з іншими об'єктами.

Діаграма станів показує, як об'єкт переходить з одного стану в інше. Вочевидь, що діаграми станів служать для моделювання динамічних аспектів системи (як і діаграми послідовностей, кооперації, прецедентів і, як ми побачимо далі, діаграми діяльності). Діаграма станів корисна при моделюванні життєвого циклу об'єкту (як і її приватний різновид - діаграма діяльності).

Від інших діаграм діаграма станів відрізняється тим, що описує процес зміни станів лише одного екземпляра певного класу - одного об'єкту, причому об'єкту реактивного, тобто об'єкту, поведінка якого характеризується його реакцією на зовнішні події. Поняття життєвого циклу застосовне якраз до реактивних об'єктів, справжній стан (і поведінка) яких обумовлений їх минулим станом. Але діаграми станів важливі не лише для опису динаміки окремого об'єкту. Вони можуть використовуватися для конструювання виконуваних систем шляхом прямого і зворотного проектування.

Діаграма станів для системи «реплікація інформації з урахуванням якісно-кількісних характеристик» зображена на рисунку 4.9.

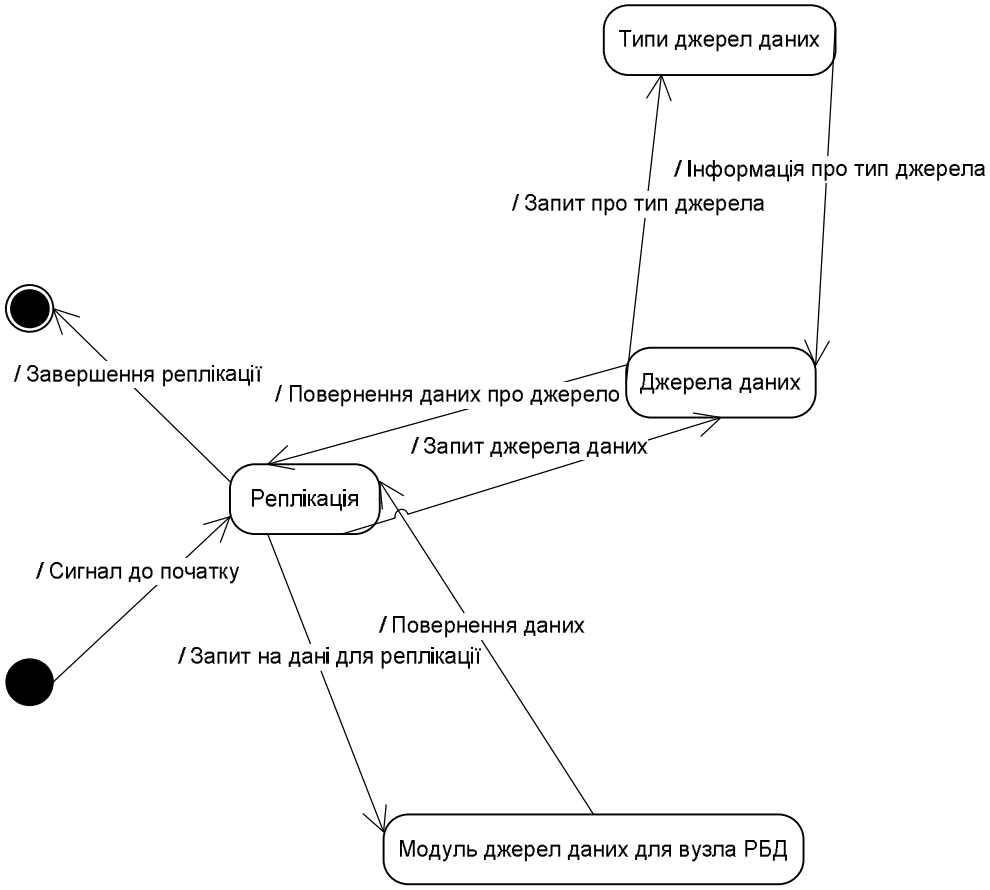


Рисунок 4.9. Діаграма станів

5. Діаграма активності (діяльності, activity diagramю). Моделюючи поведінку проектованої системи, часто недостатньо змалювати процес зміни її станів, а потрібно також розкрити деталі алгоритмічної реалізації операцій, що виконуються системою. Для цієї мети традиційно використовувалися блок-схеми або структурні схеми алгоритмів. У UML для цього існують діаграми діяльності, що є частковим випадком діаграм станів. Діаграми діяльності зручно застосовувати для візуалізації алгоритмів, по яких працюють операції класів. Алгоритм - послідовність певних дій або елементарних операцій, виконання яких наводить до здобуття бажаного результату.

Позначення на діаграмі активності також нагадують позначення на блок-схемі, хоча є, як ми побачимо далі, і деякі істотні відмінності. З іншого боку, нотація діаграм активності дуже схожа на ту, яка використовується в

діаграмах станів. Діаграма активності для системи «реплікація інформації з урахуванням якісно-кількісних характеристик» зображена на рисунку 4.10.

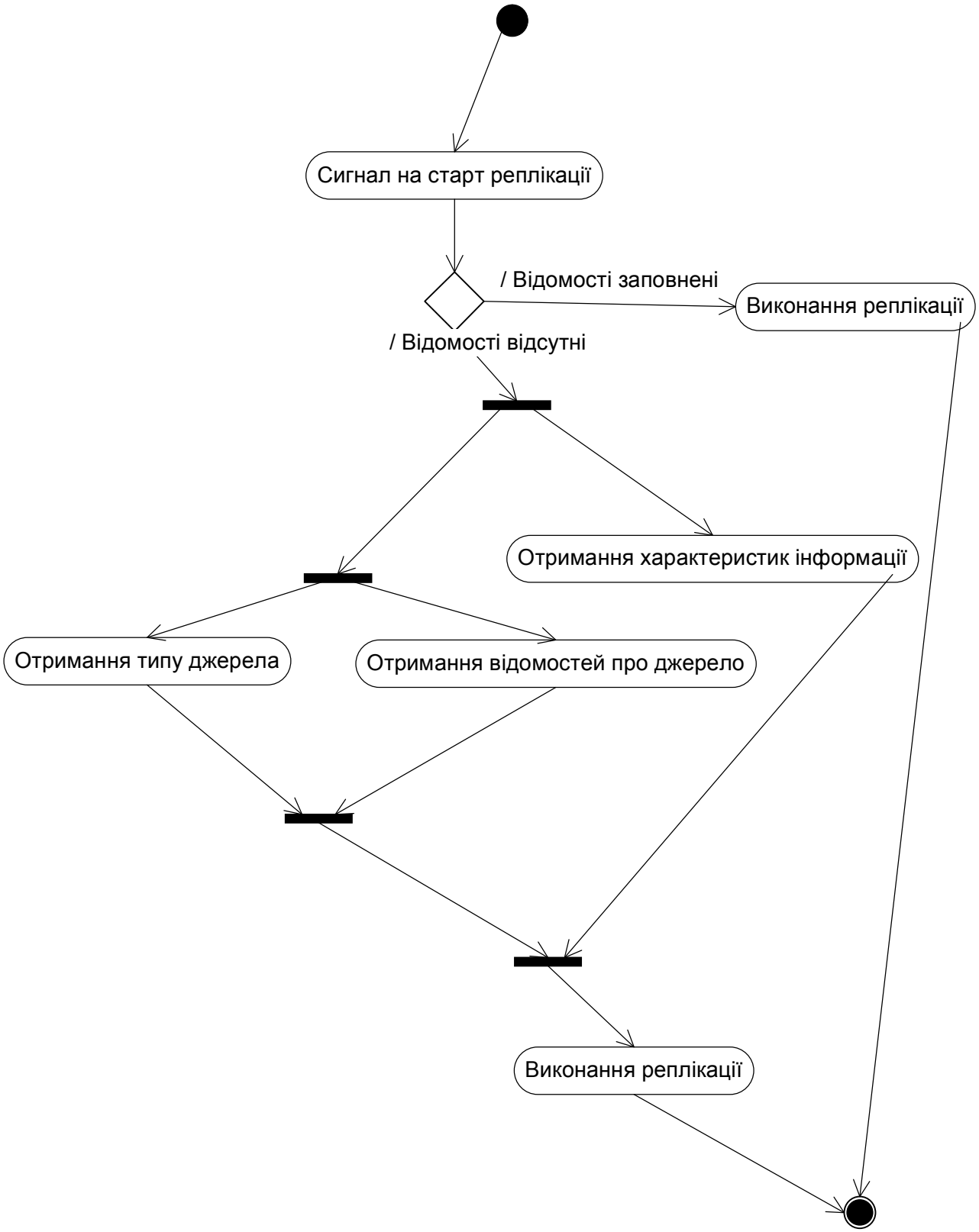


Рисунок 4.10. Діаграма активності

4.5 Тестування системи синхронізації баз даних користувачів із врахуванням якісно-кількісних характеристик.

В ході створення даної інформаційної системи було створено наступні форми у IDE Microsoft Visual Studio 2008: “Реплікація з урахуванням якісно-кількісних характеристик» - головна сторінка, “Список реплікацій”, “Реплікація”, “Типи якісно-кількісних характеристик”, “Джерела даних”, “Типи джерел даних”.

При запуску системи з’являється екранна форма “Реплікація з урахуванням якісно-кількісних характеристик”, реалізована окремим класом, яка являється головною формою та дозволяє отримати доступ до усіх інший модулів системи, оцінити кількість активних та запланованих реплікацій. Екранна форма “Реплікація з урахуванням якісно-кількісних характеристик” зображена на рисунку 4.11.

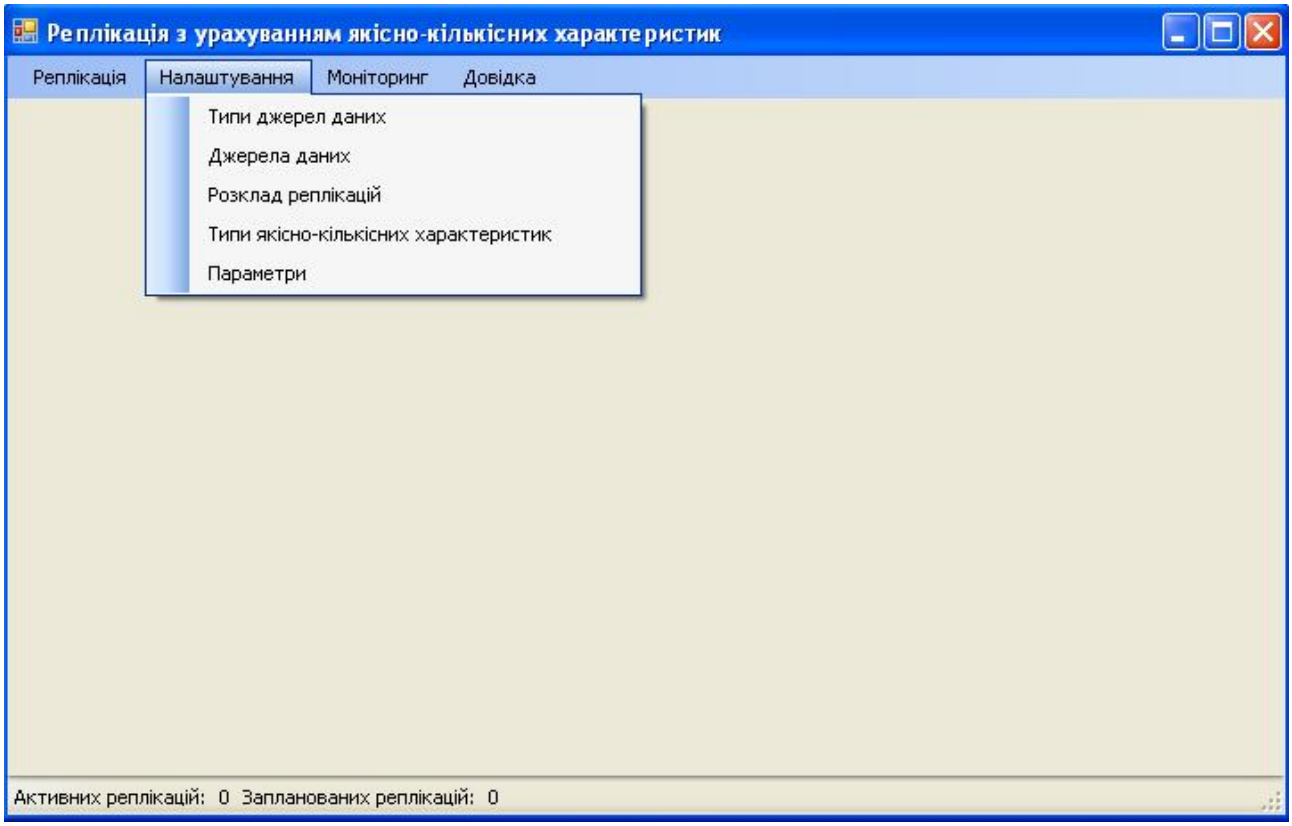


Рисунок 4.11. Реплікація з урахуванням якісно-кількісних характеристик.

Однією з найбільш важливих характеристик, які необхідно налаштувати на початку роботи системи, являються типи джерел даних, розміщені на однойменній формі. Дані зберігаються, і відповідна екранна форма “Типи джерел даних” зображена на рисунку 4.12. Тут реалізовано такі основні фунції як вибір і додавання джерел даних для здійснення синхронізації, їх редагування та видалення. Перегляд та редагування відповідних атрибутів: адреса, порт, користувач, пароль та інші.

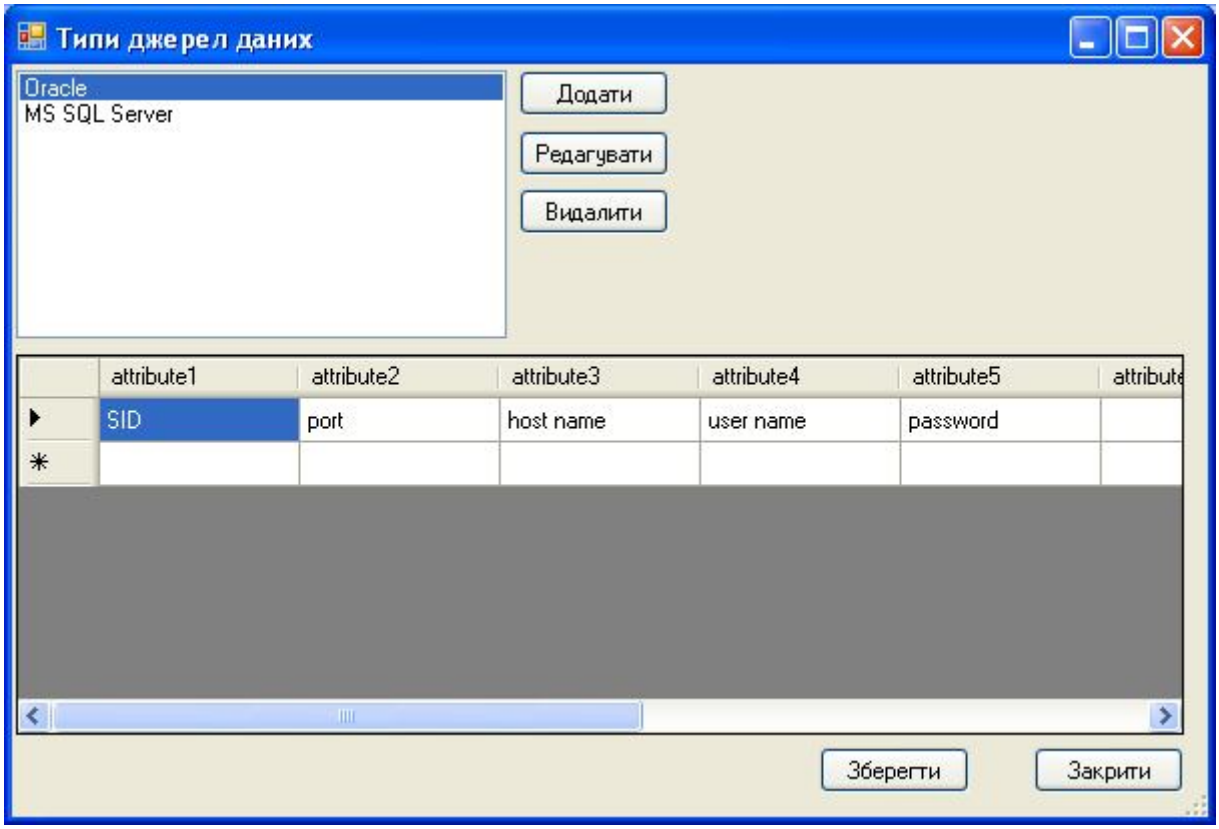


Рисунок 4.21. Типи джерел даних

Після заповнення типів джерел даних, що виступають своєрідним довідником, доступна можливість заповнення інформації про конкретні джерела даних. Екранна форма джерел даних представлена на рисунку 4.12. На даному рисунку наведено приклади відповідних атрибутів для СУБД Oracle та MS SQL Sercer.

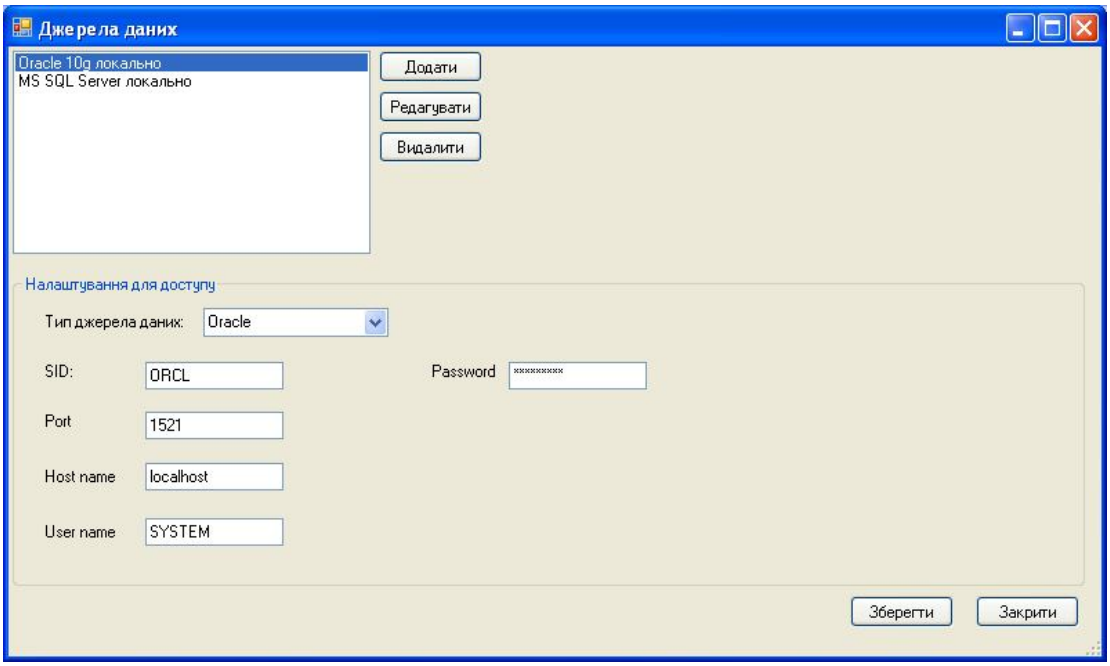


Рисунок 4.13. Джерела даних

Особливо важливими є налаштування параметрів для механізму визначення якісно-кількісних характеристик інформації, що реплікується. Екранна форма для налаштування даних параметрів зображена на рисунку 4.14

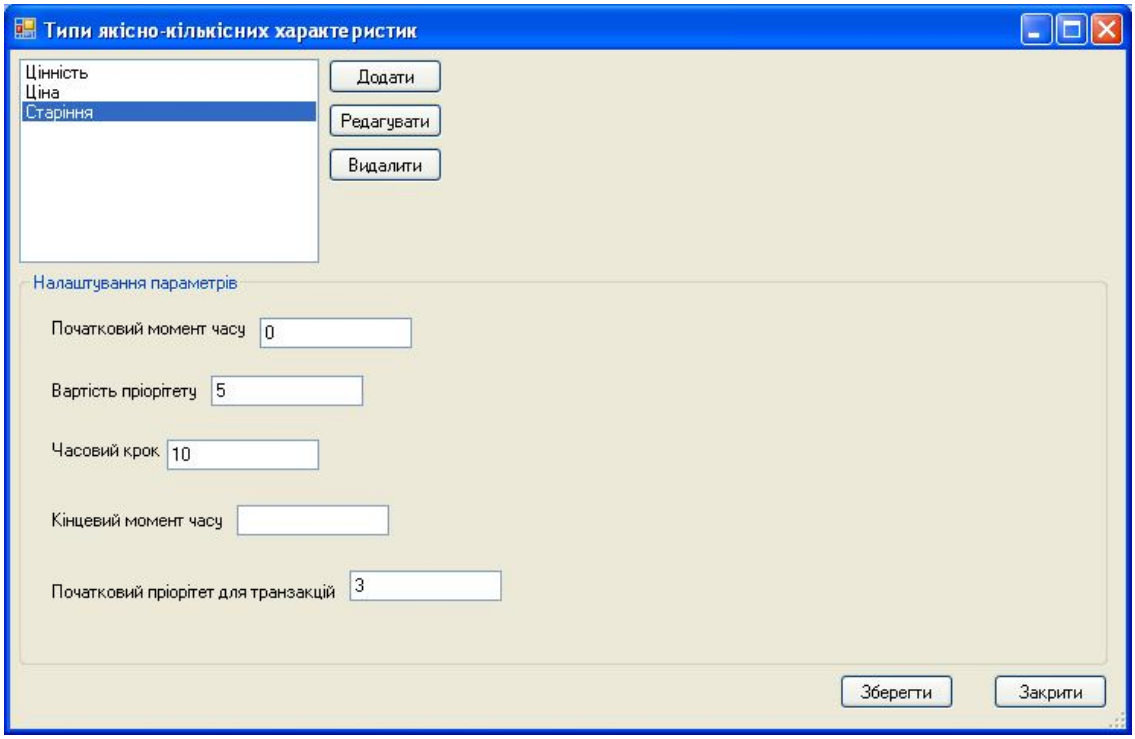


Рисунок 4.14. Типи якісно-кількісних характеристик інформації, що реплікується

На рисунку 4.15 наведено процес синхронізаці двох баз даних із врахуванням якісно-кількісних характеристик.

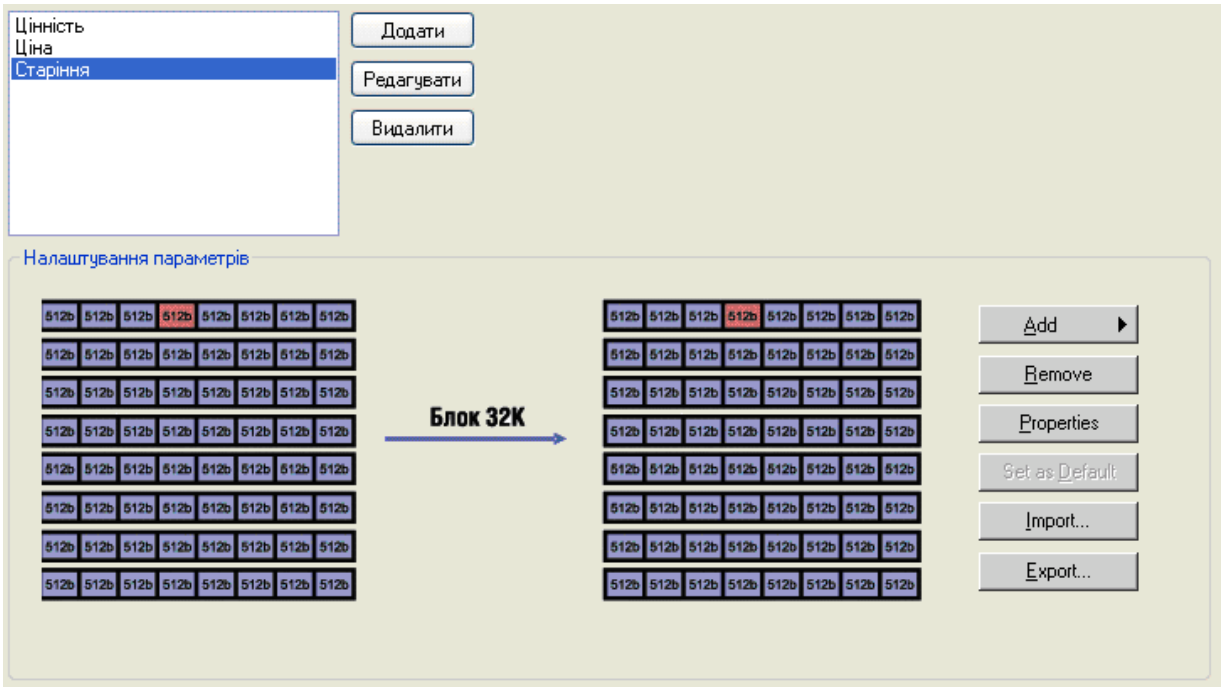


Рисунок 4.15. Процедура синхронізації двох баз даних із врахуванням якісно-кількісних характеристик

Проведено дослідження системи для різних систем управління базами даних та різни операційних систем, тобто досліджено оособливості кросплатформеної організації. На рисунку 4.16 наведено результати експериментальних досліджень.

Рисунок 4.16. Результати досліджень кросплатформеності системи

Висновки до розділу 4

В основній частині дипломної роботи запропонована загальна структура системи, обґрунтовано ефективність методів визначення цінності та старіння інформації, проведено тестування методів розв’язання задач інформаційної системи, розроблена програмна реалізації системи синхронізації баз даних користувачів мультиплатформеного веб-сайту та здійснено тестування системи синхронізації баз даних користувачів із врахуванням якісно-кількісних характеристик.

РОЗДІЛ 5

ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Метою даного розділу є обґрунтування економічної ефективності реалізації проєкту синхронізації баз даних із врахуванням цінності інформації.

Щоб виконати оцінку економічної ефективності необхідно розрахувати трудомісткість реалізації проєкту, витрати на оплату праці найманим працівникам, витрати апаратного і програмного забезпечення, амортизаційні відрахування, витрати енергоресурсів та інші витрати які є основними пунктами виконання обчислень, а також показники економічної ефективності розробки проекту.

5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Розробку проєкту синхронізації баз даних із врахуванням цінності інформації поділено на декілька етапів, що дозволяє полегшити і структурувати виконання побудови системи.

Основні етапи при виконанні побудови інформаційної системи, наступні:

1. Підготовка опису задачі.
2. Збір необхідної інформації для реалізації .
3. Вибір програмних засобів для побудови інформаційної системи психометричних тестів.
4. Побудова структури інформаційної системи.
5. Побудова і наповнення інформаційної системи.
6. Тестування інформаційної системи.

Для оцінки тривалості виконання окремих робіт використовують нормативи часу.

Виконавцем усіх операцій по побудові інформаційної являється інженер.

Витрати часу по окремих операціях технологічного процесу відображені в таблиці 5.1.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1.	Підготовка опису задачі.	Інженер	8
2.	Збір необхідної інформації для побудови інформаційної системи.	Інженер	16
3.	Вибір програмних засобів для побудови інформаційної системи для консолідації соціо-комунікаційних ресурсів "розумного міста".	Інженер	32
4.	Побудова структури інформаційної системи.	Інженер	64
5.	Побудова і наповнення інформаційної системи для консолідації соціо-комунікаційних ресурсів "розумного міста".	Інженер	72
6.	Тестування інформаційної системи	Інженер	20
Разом			212

Загальні затрати часу на реалізацію даної розробки становлять 212 годин, найбільше часу витрачено на побудову і наповнення інформаційної системи – 72 години.

5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

Відповідно до Закону України «Про оплату праці» заробітна плата – це «винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу».

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника. Заробітна плата складається з основної та додаткової оплати праці.

Основна заробітна плата нараховується за виконану роботу за тарифними ставками.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов'язані з виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Тарифна ставка інженера – 130 грн./год.

Основна заробітна плата розраховується за формулою:

$$З_{осн.} = Т_с \cdot К_г \,, \tag{5.1}$$

де $Т_с$ – тарифна ставка, грн.; $К_г$ – кількість відпрацьованих годин.

Отже, основна заробітна плата буде розраховуватись за однією формулою:

$$З_{осн.} = 130 \cdot 212 = 27560 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати.

$$З_{дод.} = З_{осн.} \cdot К_{допл.}, \tag{5.2}$$

де $К_{допл}$ – коефіцієнт додаткових виплат працівникам, 0,1–0,15 (візьмемо його рівним 0,15).

$$З_{дод} = 27560 \cdot 0{,}15 = 4134 \text{ грн.}$$

Звідси загальні витрати на оплату праці ($В_{о.п.}$) визначаються за формулою:

$$В_{о.п.} = З_{осн.} + З_{дод.} \quad (5.3)$$

$$В_{о.п.} = 27650 + 4134 = 31694 \text{ грн.}$$

Крім того, слід визначити відрахування на соціальні заходи:

- єдиний соціальний внесок ЄСВ (прибутковий податок) – 22%;
- військовий збір – 1,5%.

У сумі зазначені відрахування становлять 23,5 %.

Отже, сума відрахувань на соціальні заходи буде становити:

$$В_{с.з.} = Ф_{оп} \cdot 0{,}235 \quad (5.4)$$

де $Ф_{оп}$ – фонд оплати праці, грн.

$$В_{с.з.} = 31694 \cdot 0{,}235 = 7446{,}09 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблицю 5.2.

Таблиця 5.2 – Розрахунки витрат на оплату праці

№з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Нарахув. на $Ф_{ОП}$, грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн	Кількість відпрацьованих	Фактично нарах. з/пл., грн.			
А	Б	1	2	3	4	5	6
1.	інженер	130	212	27560	4134	7448,09	39142,09

З таблиці розрахунки витрат на оплату праці видно, що всього витрати на плату праці становить 39142,09 грн.

5.3 Розрахунок матеріальних витрат

Матеріальні витрати визначаються як добуток кількості витрачених матеріалів та їх ціни:

$$M_{вi} = q_i \cdot p_i \,, \qquad (5.5)$$

де: q_i – кількість витраченого матеріалу і–го виду; p_i – ціна матеріалу і–го виду.

Звідси, загальні матеріальні витрати можна визначити:

$$З_{м.в.} = \sum M_{вi} \,. \qquad (5.6)$$

Розрахунки занесемо у таблицю 5.3.

Таблиця 5.3 – Розрахунки матеріальних витрат

Найменування матеріальних ресурсів	Один. виміру	Норма витрат	Ціна за один., грн.	Затрати матер., грн.	Транс-портно-заготівель-ні витрати, грн.	Загальна сума витрат на матер., грн.
1. Основні матеріали						
Використання мережі Internet	години	–	–	150	–	150
2. Допоміжні витрати						
Папір формату А4	шт.	400	0,18	72	–	72
Разом:						222

Загальні матеріальні витрати на Internet і Папір формату А4 становить 222 грн.

5.4 Розрахунок витрат на електроенергію

Затрати на електроенергію 1–ці обладнання визначаються за формулою:

$$З_в = W \cdot T \cdot S,$$

(5.7)

де W – необхідна потужність, кВт; T – кількість годин на реалізацію розробки; S – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 2,42 грн.

Потужність комп'ютера для створення дипломної роботи – 212 Вт, кількість годин роботи обладнання згідно таблиці 5.1 – 212 годин.

Тоді,

$$З_в = 0{,}4 \cdot 212 \cdot 2{,}42 = 205{,}22 \text{ грн.}$$

Згідно формули затрати на електроенергію, де необхідна потужність множиться на кількість годин на реалізацію розробки і множиться на вартість кіловат-години електроенергії, що в висновку дорівнює 205,22 грн.

5.5 Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їхнього повного відновлення.

Для визначення амортизаційних використовується формула:

$$A = \frac{Б_В \cdot Н_А}{100\,\%}, \qquad (5.8)$$

де A – амортизаційні відрахування за звітний період, грн.; $Б_В$ – балансова вартість групи основних фондів на початок звітного періоду, грн.; $Н_А$ – норма амортизації.

Комп'ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для даної дипломної роботи засобом розробки є комп'ютер. Його сума становить 10299 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 10299 \cdot 5\% / 100\% = 514{,}95 \text{ грн.}$$

Згідно формули для визначення амортизаційних, де $Б_В$ множиться $Н_А$ і ділиться на 100% амортизація розробки становить 514,95 грн.

5.6 Обчислення накладних витрат

Накладні витрати пов'язані з обслуговуванням виробництва, утриманням апарату управління спілкою та створення необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб'єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$Н_в = В_{о.п.} \cdot 0{,}2 \ldots 0{,}6 \,, \qquad (5.9)$$

де $Н_в$ – накладні витрати.

Отже, накладні витрати:

$$Н_в = 31694{,}09 \cdot 0{,}2 = 6338{,}8 \text{ грн.}$$

Накладні витрати згідно розрахунку формули, становить 6338,8 грн.

5.7 Складання кошторису витрат та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків зведемо у таблицю 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці	31694	68,27
Відрахування на соціальні заходи	7448,09	16,04
Матеріальні витрати	222	0,48
Витрати на електроенергію	205,22	0,44
Амортизаційні відрахування	514,95	1,11
Накладні витрати	6338,8	13,65
Собівартість	46423,06	100,00

Собівартість ($C_в$) програмного продукту розрахуємо за формулою:

$$C_в = B_{о.п.} + B_{с.з.} + З_{м.в.} + З_в + A + H_в \,. \quad (5.10)$$

Отже, собівартість програмного продукту дорівнює:

Св = 31694+7448,09+222+205,22+514,95+6338,8= 46423,06 грн.

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 46423,06 грн.

5.8 Розрахунок ціни програмного продукту

Ціну науково-дослідної роботи можна визначити можна визначити за формулою:

$$Ц = \frac{C_В \cdot (1 + P_{рен}) + К \cdot B_{н.і.}}{К} \cdot (1 + ПДВ), \quad (5.11)$$

де $P_{рен.}$ – рівень рентабельності, 30 %; $К$ – кількість замовлень, од. (встановлюється лише при розробці програмного продукту та мікропроцесорних систем); $B_{н.і.}$ – вартість носія інформації, грн. (встановлюється лише при розробці програмного продукту); *ПДВ* – ставка податку на додану вартість, (20 %).

Оскільки розробка є прикладною, і використовуватиметься тільки для однієї установи, то для розрахунку ціни не потрібно вказувати коефіцієнти К та Ві.н , оскільки їх в даному випадку не потрібно.

Тоді, формула для обчислення ціни розробки буде мати вигляд:

$$Ц = C_В \cdot (1 + P_{рен}) \cdot (1 + ПДВ) \quad (5.12)$$

Звідси ціна на роботу складе:

$$Ц = 46423{,}06 \cdot (1 + 0{,}3) \cdot (1 + 0{,}2) = 72419{,}97 \text{ грн.}$$

Загальний розрахунок ціни реалізації проєкту становить 72419,97грн.

5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність (E_p) полягає у відношенні результату виробництва до затрачених ресурсів:

$$E_p = \frac{П}{C_B}, \qquad (5.13)$$

де $П$ – прибуток; C_B – собівартість.

Плановий прибуток ($П_{пл}$) знаходимо за формулою:

$$П_{пл} = Ц - C_в \, . \qquad (5.14)$$

Розраховуємо плановий прибуток:

$$П_{пл} = 72419{,}97 - 46423{,}06 = 25996{,}91 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_p = \frac{П_{пл}}{C_в}. \qquad (5.15)$$

Тоді,

$$E_p = 25996{,}91 \, / \, 46423{,}06 = 0{,}56.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (T_p):

$$T_p = \frac{1}{E_p} \, , \qquad (5.16)$$

Термін окупності дорівнює:

$$T_p = 1 \, / \, 0{,}56 = 1{,}8 \text{ р.}$$

Згідно формул плановий прибуток від реалізації проєкту синхронізації баз даних із врахуванням цінності інформації становить 25996,91 грн., економічна ефективність дорівнює 0,56 , а термін окупності становить 1,8 роки що вважається доцільним та економічно вигідним.

Висновок до розділу 5

В розділі обґрунтування економічної ефективності дипломної роботи освітнього рівня «магістр» було розраховано основні техніко-економічні показники реалізації проєкту синхронізації баз даних із врахуванням цінності інформації (див. таблицю 5.5).

Значення показника економічної ефективності становить 0,56, що є достатньо високим значенням.

Період окупності повинен розраховуватись від 1 до 3 років, тоді проект вважається доцільним та економічно вигідним. Термін окупності даної роботи становить 1,8 років.

Таблиця 5.5 – Техніко-економічні показники науково-дослідної роботи

№ п/п	Показник	Значення
1.	Собівартість, грн.	46423,06
2.	Плановий прибуток, грн.	25996,91
3.	Ціна, грн.	72419,97
4.	Економічна ефективність	0,56
5.	Термін окупності, рік	1,8

Отже, реалізації проєкту синхронізації баз даних із врахуванням цінності інформації може бути реалізована та розвинена, оскільки вона є економічно вигідною.

РОЗДІЛ 6
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1. Аналіз нещасних випадків та порядок їх розслідування

Розслідування проводиться у разі виникнення нещасного випадку, а саме обмеженої в часі події або раптового впливу на працівника небезпечного виробничого фактора чи середовища, що сталися у процесі виконання ним трудових обов'язків, внаслідок яких зафіксовано шкоду здоров'ю, зокрема від одержання поранення, травми, у тому числі внаслідок тілесних ушкоджень, гострого професійного захворювання і гострого професійного та інших отруєнь, одержання сонячного або теплового удару, опіку, обмороження, а також у разі утоплення, ураження електричним струмом, блискавкою та іонізуючим випромінюванням, одержання інших ушкоджень внаслідок аварії, пожежі, стихійного лиха (землетрусу, зсуву, повені, урагану тощо), контакту з представниками тваринного і рослинного світу, які призвели до втрати працівником працездатності на один робочий день чи більше або до необхідності переведення його на іншу (легшу) роботу не менш як на один робочий день, зникнення, а також настання смерті працівника під час виконання ним трудових (посадових) обов'язків.

До гострого професійного отруєння належить захворювання, що виникло після однократного впливу на працівника шкідливої речовини (речовин).

До гострого професійного захворювання належить захворювання, що виникло після однократного (протягом не більш як однієї робочої зміни) впливу шкідливих факторів фізичного, біологічного та хімічного характеру.

Про кожний нещасний випадок потерпілий або працівник, який його виявив, чи інша особа - свідок нещасного випадку повинні негайно

повідомити безпосереднього керівника робіт чи іншу уповноважену особу підприємства і вжити заходів до надання необхідної допомоги потерпілому.

У разі настання нещасного випадку безпосередній керівник робіт зобов'язаний терміново організувати надання першої невідкладної допомоги потерпілому, забезпечити у разі потреби його доставку до лікувально-профілактичного закладу; негайно повідомити роботодавця про те, що сталося; зберегти до прибуття комісії з розслідування (спеціального розслідування) нещасного випадку обстановку на робочому місці та машини, механізми, обладнання, устаткування (далі - устаткування) у такому стані, в якому вони були на момент настання нещасного випадку (якщо це не загрожує життю чи здоров'ю інших працівників і не призведе до більш тяжких наслідків та порушення виробничих процесів), а також вжити заходів до недопущення подібних нещасних випадків.

Лікувально-профілактичний заклад обов'язково проводить необхідні дослідження і складає протокол про наявність в організмі потерпілого алкоголю (наркотичних засобів чи отруйних речовин) та визначає ступінь його сп'яніння. Відповідний висновок чи витяг з протоколу, а також висновок про ступінь тяжкості травми згідно з Класифікатором травм за ступенем тяжкості, затвердженим наказом Міністерства охорони здоров'я України № 370 від04.07.2007 р. надаються на запити роботодавця, ФССНВ або голови комісіїз розслідування нещасного випадку.

Роботодавець, одержавши повідомлення про нещасний випадок, зобов'язаний:

1) протягом однієї години передати з використанням засобів зв'язку та протягом доби на паперовому носії повідомлення про нещасний випадок згідно з додатком 2 до Порядку:

- ФССНВ за місцезнаходженням підприємства, на якому стався нещасний випадок;

- керівникові первинної організації профспілки незалежно від членства потерпілого в профспілці (у разі наявності на підприємстві кількох

профспілок - керівникові профспілки, членом якої є потерпілий, а у разі відсутності профспілки - уповноваженій найманими працівниками особі з питань охорони праці);

- керівникові підприємства, де працює потерпілий, якщо потерпілий є працівником іншого підприємства;

- органові державного пожежного нагляду за місцезнаходженням підприємства у разі настання нещасного випадку внаслідок пожежі;

- закладові державної санітарно-епідеміологічної служби, який здійснює санітарно-епідеміологічний нагляд за підприємством (у разі виявлення гострого професійного захворювання (отруєння);

2) протягом доби утворити комісію у складі не менш як три особи та організувати проведення розслідування.

До складу комісії входять керівник (спеціаліст) служби охорони праці або посадова особа, на яку роботодавцем покладено виконання функцій з охорони праці (голова комісії), представник Фонду за місцезнаходженням підприємства, представник первинної профспілки (у разі наявності на підприємстві кількох профспілок - представник профспілки, членом якої є потерпілий, а у разі відсутності профспілки - уповноважена найманими працівниками особа з питань охорони праці), а також представник підприємства, інші особи.

Якщо потерпілий є працівником іншого підприємства, до складу комісії входять також представники такого підприємства та його первинної організації профспілки, а у разі відсутності на підприємстві профспілки - уповноважена найманими працівниками особа з питань охорони праці.

До складу комісії не може входити безпосередній керівник робіт.

Потерпілий або уповноважена ним особа, яка представляє його інтереси, не входить до складу комісії, але має право брати участь у її засіданнях, вносити пропозиції, подавати документи щодо нещасного випадку, давати відповідні пояснення, в тому числі викладати в усній і письмовій формі особисту думку щодо обставин і причин настання

нещасного випадку та одержувати від голови комісії інформацію про хід проведення розслідування.

Голова комісії зобов'язаний письмово поінформувати потерпілого або уповноважену ним особу, яка представляє його інтереси, про його або її права і з початку роботи комісії запросити до співпраці.

Комісія зобов'язана протягом трьох робочих днів з моменту її утворення:

- обстежити місце настання нещасного випадку, одержати письмові пояснення потерпілого, якщо це можливо, опитати осіб - свідків нещасного випадку та причетних до нього осіб;

- визначити відповідність умов праці та її безпеки вимогам законодавства про охорону праці;

- з'ясувати обставини і причини настання нещасного випадку;

- вивчити первинну медичну документацію (журнал реєстрації травматологічного пункту лікувально-профілактичного закладу, звернення потерпілого до медичного пункту або медико-санітарної частини підприємства, амбулаторну картку та історію хвороби потерпілого, документацію відділу кадрів, відділу (служби) охорони праці тощо);

- визначити, пов'язаний чи не пов'язаний нещасний випадок з виробництвом;

- установити осіб, які допустили порушення вимог законодавства про охорону праці, а також розробити план заходів щодо запобігання подібним нещасним випадкам;

- скласти у п'яти примірниках акт проведення розслідування нещасного випадку за формою Н-5 (далі - акт за формою Н-5) згідно з додатком 3 та акт про нещасний випадок, пов'язаний з виробництвом, за формою Н-1 (далі - акт за формою Н-1) згідно з додатком 4 (у разі, коли нещасний випадок визнано таким, що пов'язаний з виробництвом) і передати їх роботодавцеві для затвердження;

- скласти у разі виявлення гострого професійного захворювання (отруєння), пов’язаного з виробництвом, крім актів за формою Н-5 і Н-1, у шістьох примірниках картку обліку професійного захворювання (отруєння) за формою П-5 (далі - картка за формою П-5) згідно з додатком 5.

Акти за формою Н-5 і Н-1 підписуються головою та всіма членами комісії. У разі незгоди із змістом акта член комісії підписує його з відміткою про наявність окремої думки, яку викладає письмово і додає до акта за формою Н-5 як його невід’ємну частину.

У разі виникнення потреби у проведенні лабораторних досліджень, експертизи, випробувань для встановлення обставин і причин настання нещасного випадку строк розслідування може бути продовжений за письмовим погодженням з територіальним органом Держгірпромнагляду за місцезнаходженням підприємства.

У разі отримання письмового погодження роботодавець приймає рішення про продовження строку проведення розслідування.

У разі коли нещасний випадок визнаний комісією таким, що не пов’язаний з виробництвом, складається акт за формою Н-5, а акт Н-1 не складається.

6.2. Психофізіологічне розвантаження працівників галузі ІТ

Поява та впровадження нових інформаційно-комунікаційних технологій зумовлює необхідність подальшого вдосконалення охорони праці фахівців ІТіндустрії.

З метою належного правового забезпечення необхідно розширити ти доповнити перелік основних професій комп’ютерної галузі у національному класифікаторі ДК-003-2010, а також підготувати відповідний випуск у кваліфікаційному довіднику посад фахівців ІТ-індустрії, що сприятиме вирішенню питань їх соціального захисту, пенсійного забезпечення, атестації робочих місць основних професій за умовами праці на предмет подальших

певних видів пільг та компенсацій за важкі шкідливі і небезпечні умови праці.

Важливим напрямом стосовно визначення професійної придатності фахівців з інформаційних технологій є проведення психофізіологічної експертизи відповідно до 5 статті Закону України «Про охорону праці».

Робота з комп'ютерами нового покоління характеризується певним психофізіологічними перенавантаженнями, втомою зорового аналізатора, гіпокінезією, відсутність диференційованих норм праці при роботі з новою комп'ютерною технікою в залежності від віку, статі, категорії зорової роботи, режимів праці і відпочинку (протягом робочого дня, тижня, щорічного режиму відпусток).

Все це потребує розробки нових нормативно-правових актів з регламентації праці та відпочинку фахівців IT-індустрії і стандартів підприємств, центрів комп'ютерної техніки, центрів інформаційних технологій, сучасних комп'ютерних класів. Особлива роль з точки зору збереження та відновлення здоров'я працюючих в комп'ютерні' галузі належить попереднім та періодичним наглядам з подальшої психофізіологічної експертизи і встановленням професійної придатності при роботі з комп'ютерами нового покоління, якийсупроводжується виникненням певних факторів професійного ризику електротравматизму при їх ремонті та обслуговуванні.

В цьому зв'язку необхідне запровадження експертизи на предмет безпечної експлуатації ПЕОМ, тобто офіційне підтвердження фактичних параметрів електробезпеки, їх відповідності вимогам нормативної документації фахівців, які проводять таку експертизу повинні пройти навчання і перевірку знань відповідно до вимог ДНАОП 0.00-8.20-99. За результатами експертизи повинні прийматися рішення про відповідність ПЕОМ нормам безпеки, терміни чергової експертизи, оформлюються протоколи вимірювань і випробувань, проведені у разі потреби розрахунки та експертний висновок.

Для підвищення розумової працездатності то зорової роботи повинна здійснюватися ергономічна оптимізація в рамках системи «оператор-термінал», яка сприятиме результативній фізичній та інтелектуальний працездатності і відновленню психосоматичного здоров'я фахівців ІТ-індустрії.

Заслуговує на увагу зарубіжний досвід створення у приміщеннях та в зоні їх розміщення на територіях підприємств спеціальних візуальних комфортних умов та забезпечення вимог виробничої естетики, дотримання норм рівнів виробничого шуму та акустичної тиші за межами офісу.

Також дуже важливим є використання в офісних приміщеннях та кабінетах психофізіологічного розвантаження функціональної музики, яка сприяє попередженню перевтоми і підтриманню необхідного рівня розумової працездатності фахівців комп'ютерної галузі [2] [19]. В цьому напрямі заслуговує на увагу створення при великих центрах інформаційних технологій кімнат (кабінетів) психофізіологічного розвантаження працівників галузі (на 5 місць).

Зарубіжний досвід охорони праці при використанні новітніх інформаційних технологій та сучасного комп'ютерного обладнання передбачає з метою попередження наслідків монотонної праці, підвищення рівня рухової активності і покращення розумової працездатності фахівців ІТ-індустрії під частехнологічних перерв участь у спеціальних облаштованих приміщеннях необхідним спортивним інвентарем та різними тренажерами відповідних фізичних вправ, індивідуальних тренінгових завдань відповідно до віку, статі та категорії зорової роботи.

Такий підхід дозволяє зняти надлишкове психофізіологічне перевантаження, підвищити працездатність центральної нервової системи, попередити перевтому зорового аналізатора. Показана ефективність проведення різноманітних за своєю спрямованістю вправ робітників цієї галузі (приблизно на 5-30%) [20].

Всі наведені заходи щодо вдосконалення охорони праці фахівців ІТ-індустрії повинні контролюватися службою охорони праці та комісією з охорони праці підприємства. Особливе значення у соціальному захисті цієї категорії працівників належить прийняття комплексного договору, який може забезпечити фахівців додатковими пільгами та компенсаціями.

6.3. Інженерний захист персоналу об'єкту та населення. Правила застосування.

Функціонування на території нашої країни численних об'єктів підвищеної небезпеки, переважно в зонах з підвищеною концентрацією населення, різко посилює небезпеку великих техногенних катастроф, провокує та збільшує негативну дію особливо небезпечних стихійних явищ. Щороку втрати від таких надзвичайних ситуацій вимірюються тисячами людських життів, мільярдними збитками та невиправною шкодою для природного середовища.

Масштабність і багатогранність завдань щодо протидії сучасним природним і техногенним загрозам вимагають висококваліфікованої, технічно оснащеної, мобільної державної системи цивільного захисту.

Така система визнана складовою національної безпеки, а виконання її завдань - важливим обов'язком органів виконавчої влади всіх рівнів. Відповідно, з метою наближення до світових стандартів, від назви основного інструмента державної політики у сфері протидії наслідкам катастроф - цивільна оборона ми переходимо до назви - цивільний захист. І це не випадково. Сукупність завдань, що стоять перед службами цивільної оборони багатьох країн, більше пов'язані сьогодні з проблемами мирного часу, що дозволяє говорити про цивільний захист населення і територій, а не про цивільну оборону у воєнний час.

До захисних споруд цивільного захисту належать:

1) сховище – герметична споруда для захисту людей, в якій протягом певного часу створюються умови, що виключають вплив на них небезпечних

факторів, які виникають внаслідок надзвичайної ситуації, воєнних (бойових) дій та терористичних актів;

2) протирадіаційне укриття – негерметична споруда для захисту людей, в якій створюються умови, що виключають вплив на них іонізуючого опромінення у разі радіоактивного забруднення місцевості;

3) швидкоспоруджувана захисна споруда цивільного захисту – захисна споруда, що зводиться із спеціальних конструкцій за короткий час для захисту людей від дії засобів ураження в особливий період.

Для захисту людей від деяких факторів небезпеки, що виникають внаслідок надзвичайних ситуацій у мирний час, та дії засобів ураження в особливий період також використовуються споруди подвійного призначення та найпростіші укриття.

Споруда подвійного призначення – це наземна або підземна споруда, що може бути використана за основним функціональним призначенням і для захисту населення.

Найпростіше укриття – це фортифікаційна споруда, цокольне або підвальне приміщення, що знижує комбіноване ураження людей від небезпечних наслідків надзвичайних ситуацій, а також від дії засобів ураження в особливий період.

За місцем розташування сховища можуть бути вбудовані і окремо розташовані. До вбудованих відносяться сховища, які розташовані в підвальних приміщеннях будинків, а до окремо розташованих – сховища, які розташовані за межами будинків і споруд.

Для вирішення питань щодо укриття населення в захисних спорудах цивільного захисту центральні органи виконавчої влади, місцеві державні адміністрації, органи місцевого самоврядування та суб’єкти господарювання завчасно створюють фонд таких споруд.

Порядок створення, утримання фонду захисних споруд цивільного захисту та ведення його обліку визначається Кабінетом Міністрів України. Проектування, будівництво, пристосування і розміщення захисних споруд та

об’єктів подвійного призначення здійснюються згідно з нормами, які розробляються відповідно до Закону України "Про будівельні норми".

Вимоги щодо утримання та експлуатації захисних споруд визначаються центральним органом виконавчої влади, який забезпечує формування та реалізує державну політику у сфері цивільного захисту.

Утримання захисних споруд цивільного захисту у готовності до використання за призначенням здійснюється суб’єктами господарювання, на балансі яких вони перебувають (у тому числі споруд, що не увійшли до їх статутних капіталів у процесі приватизації (корпоратизації), за рахунок власних коштів.

У разі використання однієї захисної споруди кількома суб’єктами господарювання вони беруть участь в утриманні споруди відповідно до укладених між ними договорів.

Захисні споруди цивільного захисту можуть використовуватися у мирний час для господарських, культурних і побутових потреб у порядку, що визначається Кабінетом Міністрів України. З моменту виключення захисної споруди із фонду споруд цивільного захисту вона втрачає статус захисної споруди цивільного захисту.

Володіння, користування та розпорядження спорудами, які втратили статус захисних споруд цивільного захисту, здійснюється відповідно до закону. Захисні споруди цивільного захисту державної та комунальної власності не підлягають приватизації (відчуженню).

Захисні споруди у мирний час можуть передаватися в оренду для задоволення господарських, культурних та побутових потреб із збереженням цільового призначення таких споруд, крім тих, що перебувають у постійній готовності до використання за призначенням, а саме:

1) в яких розташовані пункти управління;

2) призначених для укриття працівників суб’єктів господарювання, що мають об’єкти підвищеної небезпеки;

3) розташованих у зонах спостереження атомних електростанцій та призначених для укриття населення під час радіаційних аварій.

Особливості оренди захисних споруд визначаються типовим договором оренди, який затверджується Кабінетом Міністрів України. Контроль за готовністю захисних споруд цивільного захисту до використання за призначенням забезпечує центральний орган виконавчої влади, який здійснює державний нагляд у сферах техногенної та пожежної безпеки, спільно з відповідними органами та підрозділами цивільного захисту, місцевими державними адміністраціями.

6.4. Запобігання наслідкам аварії на виробництвах із застосуванням аміаку. Вплив аміаку організм людини. Перша допомога. Профілактика уражень.

Для отримання низьких температур технологічними схемами компресорного цеху багатьох промислових підприємств харчової та переробної промисловості передбачено застосування токсичної речовини – аміаку.

Потенційна небезпека таких технологічних схем полягає у порушенні герметичності обладнання і трубопроводів, що містять аміак. Найбільшу небезпеку з цієї точки зору являють собою руйнування автоцистерн з рідким аміаком; руйнування напірних трубопроводів компресорів; порушення герметичності відокремлювачів рідини, лінійних та циркуляційних ресиверів, запірної арматури, батарей холодильних камер.

Наслідком таких аварій є виникнення загазованості виробничого приміщення, відкритого майданчика цеху і підприємства в цілому, а також прилеглих житлових районів; утворення вибухонебезпечної суміші аміаку з повітрям в приміщеннях, внаслідок чого можливі вибухи і пожежі.

Джерелами локальних викидів аміаку можуть служити процеси стиснення газоподібного і нагнітання рідкого аміаку, а також зливно-наливні операції.

Аварії (катастрофи) на підприємствах, транспорті та продуктопроводах можуть супроводжуватися викидом (виливом) в атмосферу і на прилеглу територію небезпечних хімічних речовин (НХР), таких як хлор, аміак, синильна кислота, фосген, сірчаний ангідрид та інші. Це являє серйозну небезпеку для населення, заражене повітря уражає органи дихання, а також очі, шкіру та інші органи.

Фактори небезпеки викиду (розливу) хімічно небезпечних речовин: забруднення навколишнього середовища, небезпека для всього живого, що опинилося на забрудненій місцевості (загибель людей, тварин, знищення посівів та ін.), крім того, внаслідок можливого хімічного вибуху виникнення сильних руйнувань на значній території.

Аміак – безбарвний газ з характерним різким запахом і їдким смаком. Він майже у два рази легший від повітря.

За звичайних умов аміак легко зріджується під тиском, а при випаровуванні поглинає тепло – сильно охолоджується. Ця властивість використовується у промислових та побутових холодильниках на м'ясокомбінатах, молокозаводах, овочевих базах, тобто там, де є необхідність в охолодженій продукції. Крім того, він є сировиною багатьох хімічних виробництв. Аміак зберігається і транспортується у зрідженому стані.

Він один з найважливіших продуктів сучасної хімічної промисловості. Головною галуззю його застосування є виробництво нітратної кислоти і азотних добрив. Крім того, аміак використовують для виробництва багатьох інших хімічних продуктів. Останнім часом зріджений аміак і водний розчин аміаку стали широко застосовувати безпосередньо як азотне добриво.

Як рідина, аміак легший за воду, має меншу густину і при виході на повітря утворює слабкий дим. Вогненебезпечний, створює вибухові суміші з повітрям, отруйний. Особливо небезпечний для очей.

У випадку розливу рідкого аміаку і його концентрованих розчинів не можна доторкатися до розлитої рідини.

Ознаки отруєння аміаком:

- нежить, кашель, важке дихання, задуха;

- підвищене серцебиття, порушена частота пульсу;

- при контакті з рідким аміаком виникає обмороження, можливий опік з пухирями, виразки.

Перша допомога при отруєнні аміаком:

- одягніть протигаз і виведіть ураженого на свіже повітря;

- дайте подихати зволоженим повітрям (теплими водяними парами 10%-ного розчину ментолу в хлороформі);

- дайте йому теплого молока з «Боржомі» або харчовою содою;

- при задусі необхідний кисень;

- при спазмі голосових щілин забезпечте тепло на ділянку шиї, теплі ванночки, інгаляцію;

- при зупинці дихання проведіть серцево-легеневу реанімацію;

- при потраплянні в очі − промийте водою або 0,5-1%-ним розчином квасців, вазеліновою або оливковою олією;

- при уражені шкіри − обмийте чистою водою, зробіть примочки з 5%-ного розчину оцтової, лимонної або соляної кислоти.

При отруєнні аміаком винести потерпілого із зони зараження, шкіру, рот, ніс промити водою. В очі закапати по дві-три краплі 30% альбуциду, в ніс - оливкове масло. При необхідності відправити потерпілого до медичного закладу.

РОЗДІЛ 7
ЕКОЛОГІЯ

7.1. Класифікація показників екологічності виробництва

Комплексна оцінка екологізації економіки і екологічності виробництва є кількісною характеристикою декількох зведених взаємопов’язаних показників з урахуванням чинника техногенної безпеки у взаємозв'язку з економічними результатами виробничої діяльності.

Показники економічної ефективності відтворюють фінансові та виробничі результати реалізації екологічних програм, що передбачають певні економічні вигоди як для підприємств, так і для державних структур, що реалізують дані програми. Економічний ефект полягає в забезпеченні необхідного рівня прибутковості при здійсненні економічної діяльності. Для природоохоронних заходів таким ефектом може стати відсутність необхідності спрямовувати на вирішення екологічних проблем всезростаючу кількість матеріальних ресурсів в майбутньому, оскільки ці проблеми вже будуть вирішені. Економічний ефект від реалізації екологічних регіональних програм оцінюється приростом збереженої сумарної вартості природних ресурсів на території їх дії.

Оцінка й порівняння витрат і результатів необхідні для підготовки обґрунтованих розв’язків відносно доцільності реалізації різних програм.

При цьому слід визначати: а) компоненти витрат; б) економічні показники, що дозволяють оцінювати різноманітні елементи витрат та результатів у єдиній системі цінностей; в) чисту віддачу (різницю між результатами й витратами, у т.ч. між суспільними вигодами й суспільними витратами).

На економічному рівні ефективність інвестицій оцінюється розміром економічних ресурсів збережених внаслідок економії, або запобігання втрат природних ресурсів, живої та упредметненої праці у виробничій та

невиробничої сферах народного господарства, а також у сфері особистого споживання.

Кількісними показниками ефективності реалізованих екологічних проектів є:

- порівняльна оцінка витрат на проведення природоохоронних заходів і досягнутих завдяки цим заходам економічним результатам;

- порівняльна оцінка показників величини попереджених збитків внаслідок забруднення навколишнього середовища та річного приросту прибутків, отриманого за рахунок економії витрат і реалізації утилізованої супутньої продукції;

- порівняльна оцінка річних економічних збитків, понесених внаслідок забруднення навколишнього середовища до та після реалізації конкретного екологічного проекту.

Стан навколишнього середовища є одним з найбільш істотних факторів, що впливають на здоров'я населення. За оцінками фахівців ООН стан здоров’я населення на 20-40% залежить від стану навколишнього середовища, на 15-20% від генетичних факторів, на 25-50% – від способу життя й тільки на 10% – від діяльності системи охорони здоров’я, тому показники соціальної ефективності враховують соціально-демографічні наслідки реалізації заходів екологічної програми для суспільства в цілому, ступінь їх корисності, що характеризується поліпшенням стану здоров'я населення, зменшенням кількості захворювань і смертей внаслідок зниження викидів шкідливих речовин у навколишнє середовище.

Останнім часом усе більшу тривогу викликає погіршення демографічних показників та характеристик стану здоров’я населення в Україні. Зростання кількості забруднюючих атмосферне повітря речовин визначає антропогенний вплив людини на всі природні екосфери, яке за масштабами стало співвідносним, а в багатьох випадках і в багато разів переважаючим дію природних катастроф на середовище проживання людини. Це призводить до порушення причинно-наслідкових зв'язків між окремими

природними системами, і цей деструктивний процес рано чи пізно позначається на здоров'ї населення [9, с. 8-12]. Тому є всі підстави стверджувати, що якість життя населення є основним показником стану навколишнього середовища, яке в такій проекції можна оцінювати за наступними критеріями:

- приростом чисельності населення поточного року в порівнянні з попереднім;

- темпами народжуваності та темпами смертності.

Показники екологічної ефективності відтворюють наслідки реалізації екологічного проекту для стану навколишнього природного середовища, що може виражатися в екологічній ємності території, збільшенні біорозмаїття, підвищенні асиміляційного потенціалу території. Також до числа показників екологічної ефективності можуть бути віднесені показники, що характеризують динаміку зниження кількості забруднюючих викидів і скидань. У результаті природне середовище стає менш забрудненим, а, отже, більш стійким до негативного антропогенного впливу внаслідок економічної діяльності людини.

Ефективність природоохоронних заходів на екологічному рівні полягає в зниженні негативного впливу на навколишнє середовище та поліпшенні його стану. Зокрема, витрати на попередження негативного впливу забрудненого середовища на реципієнтів при забрудненні водних ресурсів, визначаються величиною фінансових ресурсів, необхідних для попередження використання забрудненої води на технологічні та комунально-побутові потреби. До їх числа відносяться витрати на: а) розведення (деактивацію) стічних вод; б) застосування досконаліших (ефективніших) способів очищення води; в) перенесення водозабору або забезпечення доступу водокористувачів до більш чистих водних ресурсів.

При атмосферному забрудненні аналогічні витрати необхідні для: а) застосування систем очищення повітря, що надходять у житлові та виробничі

приміщення; б) при подачі повітря для технологічних потреб; в) створення санітарно-захисних зон.

До числа витрат, спрямованих на попередження впливу забрудненого середовища відносяться витрати на збір, вилучення та поховання відходів виробництва та споживання, включаючи витрати на відчуження земель для організації місць зберігання відходів. Оціночними показниками в цьому випадку є:

- обсяги викидів (забруднення) у навколишнє середовище та їх територіальна поширеність;

- кількість та якість придатних до використання земельних, атмосферних і водних ресурсів.

В цілому, соціальна та екологічна ефективність природоохоронних заходів може виражатися в обмеженні або усуненні негативного впливу господарської діяльності на суспільство й навколишнє середовище, а також підвищенні рівня здоров’я населення й відновленні природних ресурсів і елементів, необхідних для забезпечення життєдіяльності людину. Пріоритет повинен віддаватися тим проектам, які спрямовані на запобігання забруднення навколишнього середовища.

7.2 Робота з банками екологічної інформації

Екоінформатика – це область, що формується, і реально в науці і практиці є лише її окремі структурні елементи, мало пов'язані один з одним. Необхідність же розвитку цього напряму ясно визначається вимогами сьогоднішнього дня.

Типізація інформації в загальному випадку можлива на основі визначення класу об'єкта, управління, мети управління, належності до відповідного функціонального блоку інформаційної системи, форми збирання, передачі, зберігання і зображення. Цілком зрозуміло, що типізація враховує кількісний, семантичний і прагматичний аспекти інформації.

Найбільш загальні типологічні одиниці, прямо не пов'язані з екологічною інформацією, визначаються належністю інформації до певного функціонального блоку інформаційної системи.

Можна виділити інформацію на стадії її збирання, на стадії передачі, зберігання, обробки і зображення.

Інформацію, зібрану безпосередньо за допомогою яких-небудь вимірювальних засобів, можна визначити як первинну. її характеристики повністю залежать від технічних особливостей вимірювальних засобів, від принципів організації вимірювань відповідно до цілей збирання інформації.

Своєю чергою, первинна інформація може бути за цілями поділена на такі типи: науково-пошукова, науково-режимна і практичного призначення.

Особливість науково-пошукової інформації, що збирається зазвичай в рамках фундаментальних досліджень під нові, іноді дуже нечітко визначені гіпотези і цілі: вона майже завжди надмірна й інтерактивна. Надмірність пов'язана з великою невизначеністю при постановці завдань, висока інтерактивність визначається необхідною швидкою зміною методів і методик спостережень, що забезпечує розширений науковий пошук.

Другий тип наукової інформації — науково-режимна інформація — пов'язується з дослідженнями, в яких здійснюється верифікація досить реалістичних гіпотез і теорій. Характерною її межею є достатньо чітке обґрунтування змінних, методів, об'ємів інформації, що збирається. Такого типу дослідження, як правило, проводяться за добре обґрунтованими програмами.

Специфічною рисою первинної інформації в практичній сфері діяльності є її жорстке підпорядкування певним народногосподарським завданням із максимальним здешевленням всієї процедури збирання. Вимірюються тільки ті змінні і в такому об'ємі, в якому це необхідно і достатньо для надійних практичних дій в даній предметній області.

Звичайно, майже як усюди, реальні множини, що належать до того або іншого типу інформації, є нечіткими, і завжди можна виділити деякі перехідні ситуації.

У загальному випадку завдання обробки екологічної інформації — виявлення в тій або іншій формі залежностей між змінними. Методи такого аналізу можуть застосовуватися найрізноманітніші, але мета в усіх випадках одна: здобуття емпіричних залежностей.

У всіх випадках обробки первинної інформації одержують вторинну інформацію, яка сама по собі бере участь в різних перетвореннях і має самостійну науково-практичну цінність.

В інформаційних системах з одним блоком обробки інформації результати обробки можуть розглядатись як основа для ухвалення рішень. У складніших системах вторинна інформація надходить в наступні блоки обробки й аналізу.

Досить типовим варіантом є інтеграція вторинної інформації першого роду про деякі змінні з аналогічною або первинною інформацією за іншими змінними. Наприклад, спряжений аналіз тематичних карт за різними компонентами природи.

В іншому достатньо типовому варіанті здійснюється узагальнення емпіричної вторинної інформації в теорію, дедуктивну за своєю природою. Ця операція здійснюється на основі верифікації наявних моделей або моделей, спеціально розроблених для даного об'єкта. В результаті таких операцій відбувається ще більше стискування інформації і надання їй форми елементарних дійсних висловів.

Отже, доцільно виділити вторинну інформацію другого роду. Якісно вона відрізняється від вторинної інформації першого роду — продукту прямої обробки наявних масивів даних. В одних випадках ця інформація пов'язується з вищим рівнем інтеграції змінних, а в іншому — з глибшим теоретичним узагальненням відносин.

Нарешті інформація переходить в блок підготовки її для кінцевого споживача. Аналіз первинної інформації дозволив її максимально стиснути, надавши форми, зручної для використання. Аби скористатися результатами, необхідно знов перетворити її так, щоб з обмеженого числа елементарних висловів можна було отримати множину можливих істинних складних висловів — наслідків. Така інформація є вже результат роботи моделі або теорії з її мовою. Це інформація, в якій не залишилося жодної невизначеності, а там, де невизначеність існує, вона оцінена з позицій статистичних гіпотез або ризиків, є кінцевим продуктом локальної інформаційної системи і може бути визначена як третинна. Вона може використовуватись і для практичних рішень, і для синтезів вищих рівнів інтеграції тощо.

Остаточно всі типи інформації можуть бути визначені як знання. Первинна інформація являє собою множину більш-менш точно зібраних фактів і відповідає суто кількісному аспекту інформаційного процесу. Вторинна інформація наділяється певною семантикою, третинна інформація з необхідною і можливою повнотою розкриває структуру мови і правила перетворень елементарних висловів (правила виведення наслідків).

Кожен із розглянутих типів інформації має власну потенційну прагматичну цінність і підлягає незалежному зберіганню. При цьому мається на увазі, що на будь-якому рівні обробки інформацію, як первинну, так і вторинну, можна перетворювати скільки завгодно різними способами, і немає гарантії, що прийнятий спосіб є найкращим. З цих позицій первинна інформація має найвищу прагматичну цінність, оскільки тільки при її збереженні можливе постійне послідовне вдосконалення методів отримання вторинної інформації і взагалі відображення об'єкта управління в системі знань. Якщо збережена первинна інформація, то завжди можна дістати будь-яку іншу. Якщо вона загублена, то реконструкція і розвиток досліджень здебільшого неможливі.

ВИСНОВКИ

В роботі проведено аналіз відомих методів реплікацій та синхронізації баз даних. За результатами проведеного аналізу обрано методи та алгоритми проведення реплікації, а саме наступні параметри:

1. Принципи встановлення з'єднання при реплікації.

2. Способи виявлення змін у таблицях, що синхронізуються.

Розглянуто та реалізовано врахування у програмній системі основні кількісно-якісні характеристики інформації, а саме: цінність, ціна та старіння, що дозволяють оптимальним чином проводити реплікацію в межах розподіленої бази даних торговельної мережі.

У межах реалізації програмної системи проведення реплікацій з урахуванням кількісно-якісних параметрів інформації, що передається, передбачено наступні функціональні можливості системи:

– Визначення інтерфейсів передачі даних для різних СУБД.

– Визначення інтерфейсів для якісно-кількісних характеристик передаваємої інформації.

– Надання можливості гнучкого налаштування системних параметрів для побудови оптимальної послідовності транзакцій.

– Ведення історії здійснених реплікацій, зберігання усієї необхідної інформації для подальшого аналізу.

– Можливість довільного вибору баз даних, між якими буде здійснюватися реплікація.

– Можливість реплікації даних між різнотипними вузлами розподіленої бази даних.

– Моніторинг виконання реплікацій в певний момент часу.

Таким чином, розроблено методи організації реплікацій в розподілених базах даних з урахуванням якісно-кількісних характеристик інформації, що передається.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Андон, Ф. И. Логические модели интеллектуальных информационных систем / Ф. И. Андон, А. Е. Яшунин, В. А. Резниченко. - К: Наукова думка.-1999.-397 с.

2. Антонченко, М. О. Експертні системи як засіб формування якісних знань учнів 7-8 класів з предметів природничого циклу: Автореф. дис. ... канд. пед. наук: 13.00.09 / Антонченко Марія Олексіївна ; Харк. держ. пед. ун-т ім. Г. С. Сковороди. -Х., 2001. - 16 с.

3. Атанов, Г. А. Семантическая предметная модель студента-экономиста по линейной алгебре / Г. А. Атанов, Е. Г. Евсеева // Теорія та методика навчання математики, фізики, інформатики: Збірник наукових праць: В 3-х томах. - Кривий Ріг: Видавничий відділ НацМетАУ, 2002. - Т. 1: Теорія та методика навчання математики. - С. 4-20.

4. Гайна Г. А. Основи проектування баз даних: навч. посібник / Г. А. Гайна – К.: КНУБА, 2005. – 204 с.

5. Гайна Г.А. Основи проектування баз даних: Навчальний посібник. – К.: Кондор, 2008. – 200 с.

6. Гоулд Дэвид А.Д. Полное руководство по программированию Maya. Подробное описание языка MEL и интерфейса C++ API/Пер. с англ. – М.: КУДИЦ-ОБРАЗ, 2004, – 528 с.

7. Грачьов А.Ю. Введення в СУБД Informix // Діалог-МІФІ. – 2005. – С.87 – 99.

8. Діренко, І. С. Система управління вмістом веб-ресурсів на основі онтологічно-документного моделювання / І. С Діренко, О. В. Олецький // Теоретичні та прикладні аспекти побудови програмних систем. Матеріали міжнародної конференції TAAPSD'2006. Київ, грудень 2006 р. - С171-176

9. Дейт К. Введение в системы баз данных. 7-е изд. – М.; СПб.: Вильямс, 2001; – 8-е изд. – М.; СПб.: Вильямс, 2005.

10.Джордж Бакларц, Білл Вонг DB2 для UNIX, Linux, Windows й OS/2 // DB2 Universal Database v7.1 for UNIX, Linux, Windows and OS/2 // Лорі. – 2004. – С. 113–125.

11.Диго С. М. Базы данных: проектирование и использование: учебник / С. М. Диго. – М.: Финансы и статистика, 2005. – 592 с.

12.Иванчева Н.А., Иванчева Т. А. Постреляционная СУБД Cache. – Новосибирск: Новосибирский

13.Ким Вон. Технология объектно-ориентированных баз даннях //Открытые системы. 1994. №3. С. 30-

14.Конноли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Конноли, К. Бегг. – М.: Вильямс, 2003. – 1440 с.

15.Корнєєв В.В., Гарєєв А.Ф., Васютін С.В., Райх В.В. Бази даних. Інтелектуальна обробка інформації // Нолідж. – 2006. – С. 97–127.

16.Крейг С. Маллінс Адміністрування баз даних: Повний довідковий посібник з методів і процедур // Database Administration. The Complete Guide to Practices and Procedures // КУДІЦ-Образ. – 2003. – С. 79–115.

17.Кузнецов С.Д., Базы данных: языки и модели. Учебник – М.: ООО «Бином-Пресс», 2008. – 720 с.:

18.Кэри Мілсап Джефф Хольт Oracle. Оптимізація продуктивності // Символ-Плюс. – 2006 – С. 129–143.

19. Лаврищева, Е. М. Методы программирования: теория, инженерия, практика / Е. М. Лаврищева. - К.: Наукова думка. - 2006. - 451 с.

20.Овсяк О. Класи інформаційної системи генерування коду /О.Овсяк //Вісник Тернопільського державного технічного університету, №1, 2010. – С.171 – 176.

21. Олецький О. В. Застосування формальних моделей онтологій для формалізації інформаційних потоків у системах управління контентом / О. В. Олецький //Теоретичні та прикладні аспекти побудови програмних систем. Матеріали міжнародної конференції TAAPSD'2005. Київ, 7-9 грудня 2005 р. -

С. 26-29.

22. Петровский, А. Б. Извлечение знаний для оценки кредитоспособности: подход теории мультимножеств / А. Б. Петровский // Труды Девятой национальной конференции по искусственному интеллекту с международным участием (КИИ-2004). - М.: Физматлит, 2004, том 2. -С. 853-860.

23. Поль Дюбуа MySQL // Вільямс. – 2006. – С. 45–70.

24. Разработка приложений баз данных данных в архитектуре клиент-сервер для СУБД Cache. Сост.,

25. Рамський Ю.С., Цибко Г.Ю. Проектування і опрацювання баз даних. Посібник для вчителів. – К: 2003. – 136 с.

26. Самойлов, В. Д. Модельное конструирование компьютерных приложений / В. Д. Самойлов. - К.: Наукова думка. - 2007. - 198 с.

27. Семикин, В. А. Семантическая модель контента образовательных электронных зданий: Автореф. дис. … канд. тех. наук: 05.13.18 / Семикин Виктор Алексеевич ; Тюменск. гос. ун-т. - Тюмень, 2004. - 21 с.

28. Титенко, С. В. FreshKnowledge - система управління навчальним Веб-контентом на семантичному рівні / С. В. Титенко, О. О. Гагарін // VII международная конференция «Интеллектуальный анализ информации ИАИ-2007», Киев, 15-18 мая 2007г. : Сб. тр. / Ред. кол. : С. В. Сирота (гл.ред.) и др. - К.: Просвіта, 2007. - С. 342-352.

29. Шиловский О. А., Давыдов Д. А. – Волгоград: ВолгГТУ, 2005.– 38 с.

30. Чмир, І.О. Моделювання систем у середовищі UML (Unified Modeling Language) : навч. посібник / І. О. Чмир, М. Ф. Ус ; Черкаськ. акад. менеджменту. - Черкаси : ЧАМ, 2004. - 100 с.

31. Ярмуш О.В. Інформатика і комп’ютерна техніка [Текст] : навч. посіб. / О.В. Ярмуш, М.М. Редько. – К. : Вища освіта, 2006. – 359 с.

32. Brusilovsky, P. Methods and techniques of adaptive hypermedia / P. Brusilovsky // User Modeling and User-Adapted Interaction. - 1996. - № 6 (2-3).- P. 87-129.

33. Murray, T. Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art / T. Murray. // International Journal of Artificial Intelligence in Education. -1999.-№10-P. 98-129

34. Brusilovsky, P. Adaptive and intelligent Web-based educational systems / P. Brusilovsky, C Peylo // International Journal of Artificial Intelligence in Education. Special Issue on Adaptive and Intelligent Web-based Educational Systems -2003. -№13 (2-4). -P. 159-172.

35. Marshall, B. Convergence of Knowledge Management and E-Learning: the GetSmart Experience [Електронний ресурс] / Marshall, B., et al. // JCDL. - Houston, 2003. - Режим доступу: http://ai.bpa.arizona.edu/go/intranet/Publication/JCDL-2003-Marshall.pdf.

36. FreshKnowledge CMS - онтологічно-орієнтована система керування контентом, розроблена здобувачем [Електронний ресурс]. - Режим доступу : http://www.freshknowledge.net.

37. TENCompetence - European research project for lifelong competence development [Електронний ресурс]. - Режим доступу : http://www.tencompetence.org/.

38. Network Working Group Request for Comments: 3261 [Електронний ресурс]. - Режим доступу: http://www.ietf.org/rfc/rfc3261 .txt.

39. Kuhn Richard D. Security Considerations for Voice Over IP Systems. Recommendations of the national Institute of Standards and Technology / Richard Kuhn D., Walsh Thomas J., Fries Steffen. - NIST Special Publication 800-58, 2005.-100 p.

40. SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS. Infrastructure of audiovisual services - Systems aspects Security and encryption for H-series (H.323 and other H.245-based) multimedia terminals. ITU-T Recommendation H.235, 2001 - 85 p. [Електронний ресурс]. - Режим доступу: http://www.itu.int/rec/T-REC-h.

41. Thermos P., Takanen A. Securing VoIP networks: threats, vulnerabilities, countermeasures. - Addison-Wesley Professional, 2007 - 384 p.

42.Database [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Database.

43.Database [Електронний ресурс] // Oracle FAQ. – 2008. – Режим доступу до ресурсу: http://www.orafaq.com/wiki/Database

44.PostgreSQL Wiki [Електронний ресурс] // postgresql.org – Режим доступу до ресурсу: https://wiki.postgresql.org/wiki/Main_Page.

45. Introduction to the Oracle Database [Електронний ресурс] // Oracle. – 2017. – Режим доступу до ресурсу: https://docs.oracle.com/cd/B19306_01/server.102/b14220/intro.htm.

46. Microsoft SQL Server [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Microsoft_SQL_Server.

47. MySQL [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/MySQL.

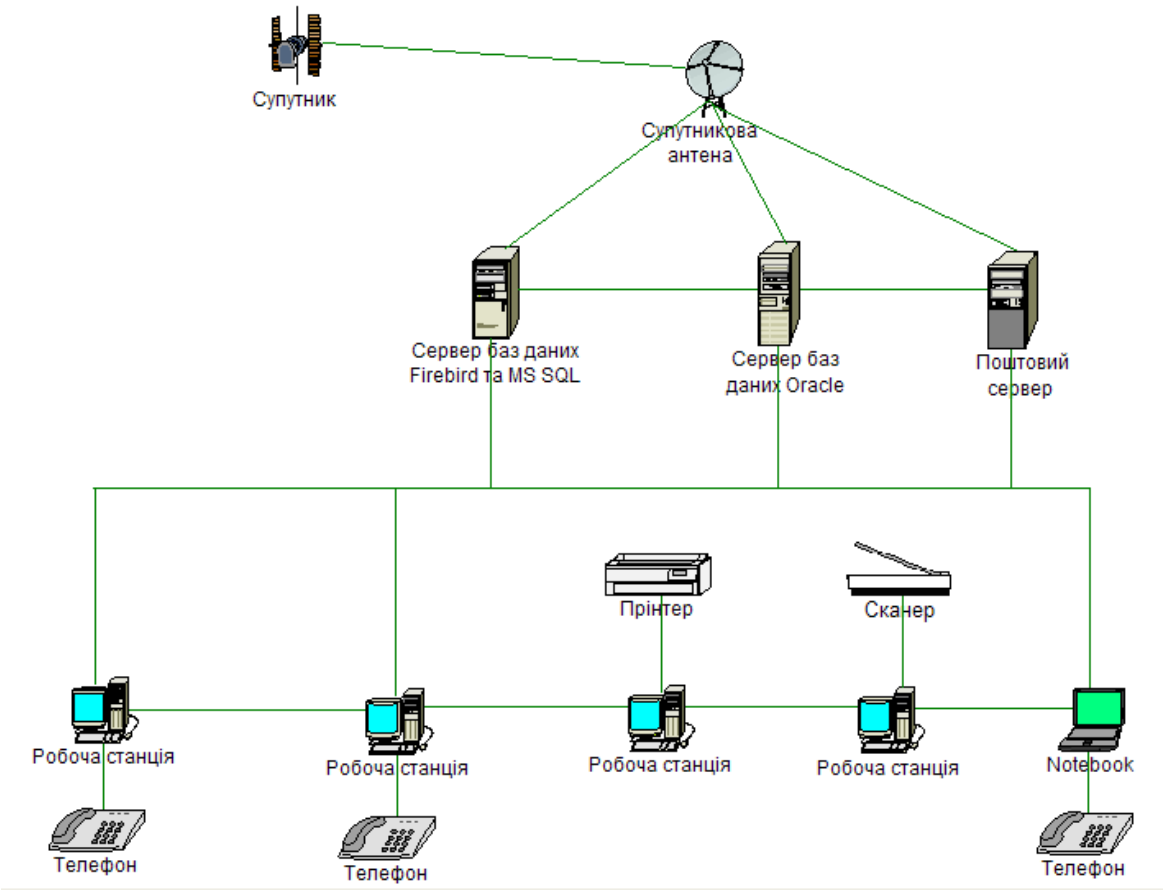
48. DB-Engines Ranking [Електронний ресурс] // DB-engines. – 2017. – Режим доступу до ресурсу: https://db-engines.com/en/ranking.

49.. Branson T. 8 Major Advantages of Using MySQL [Електронний ресурс] / Tony Branson // Datamation – Режим доступу до ресурсу: http://www.datamation.com/storage/8-major-advantages-of-using-mysql.html.\

50. Официальный сайт компании Intersystems. – Режим доступа: http://www.intersystems.com/.

ДОДАТОК А

Загальна схема стану технічного забезпечення, що застосовується при реплікації



ДОДАТОК Б

Порівняльна характеристика структури баз даних торговельної мережі, що знаходиться під керуванням двох різних систем керування базами даних

Oracle 10.g		Microsoft SQL Server	
Назва поля	Тип/Розмір	Назва поля	Тип/Розмір
1	2	3	4
		ccd_id	int
CCD_01_01	VARCHAR2(3)	ccd_01_01	varchar(3)
CCD_01_02	NUMBER(2,0),	ccd_01_02	numeric(2,0)
CCD_01_04	VARCHAR2(3),	ccd_01_03	varchar(3)
CCD_37_01	NUMBER(2,0),		
CCD_37_02	NUMBER(2,0),		
CCD_37_03	NUMBER(3,0),		
CCD_03_01	NUMBER(5,0)	ccd_03_01	smallint
CCD_04_01	NUMBER(4,0),	ccd_04_01	int
CCD_05_01	NUMBER(6,0)	ccd_05_01	int
CCD_06_01	NUMBER(6,0)	ccd_06_01	numeric(6,0)
CCD_07_01	NUMBER(9,0)	ccd_07_01	varchar(9)
CCD_07_02	NUMBER(4,0)	ccd_07_02	numeric(4,0)
CCD_07_03	NUMBER(6,0)	ccd_07_03	numeric(6,0)
CCD_11_01	NUMBER(3,0),	ccd_11_01	int
CCD_12_01	NUMBER(19,2),	ccd_12_01	numeric(16,2)
CCD_13_01	NUMBER(2,0)	ccd_13_01	varchar(1)
CCD_14_05	NUMBER(1,0),		
CCD_14_06	VARCHAR2(200),		
CCD_14_07	DATE,		
CCD_15_01	NUMBER(3,0),	ccd_15_01	int
CCD_16_01	VARCHAR2(15),		
CCD_17_01	NUMBER(3,0)	ccd_17_01	int
		ccd_18_01	smallint
CCD_19_01	NUMBER(6,0)	ccd_19_01	varchar(1)
		ccd_21_01	smallint
CCD_20_01	NUMBER(2,0),	ccd_20_01	int
CCD_20_02	VARCHAR2(30),	ccd_20_cnt	varchar(2)
CCD_20_03	NUMBER(2,0),	ccd_20_03	numeric(2,0)
CCD_22_01	NUMBER(3,0),	ccd_22_cur	int
CCD_22_02	NUMBER(19,2),	ccd_22_03	numeric(16,2)
CCD_23_01	NUMBER(16,8),	ccd_23_01	numeric(16,8)
CCD_24_01	NUMBER(3,0),	ccd_24_01	int
CCD_24_02	NUMBER(3,0),	ccd_24_02	varchar(3)
CCD_25_01	NUMBER(2,0),	ccd_25_01	int
CCD_26_01	NUMBER(2,0),	ccd_26_01	int
CCD_27_01	NUMBER(5,0),	ccd_27_01	varchar(9)
1	2	3	4
CCD_27_01_F	VARCHAR2(9),		

CCD_29_01	NUMBER(5,0),	ccd_29_01	varchar(9)
		ccd_30_01	varchar(9)
CCD_48_01	NUMBER(3,0),		
CCD_48_02	DATE,	ccd_48_01	datetime
CCD_48_03	DATE,		
CCD_48_04	NUMBER(19,2),		
CCD_49_01	NUMBER(1,0),		
CCD_49_02	VARCHAR2(20),		
CCD_51_01	DATE,	ccd_51_01	datetime
CCD_51_02	NUMBER(19,2),	ccd_51_02	numeric(16,2)
1	2	5	6
CCD_51_03	VARCHAR2(30),	ccd_51_03	varchar(20)
CCD_52_01	NUMBER(1,0),	ccd_52_01	datetime
CCD_52_02	NUMBER(5,0),		
CCD_52_03	NUMBER(3,0),		
CCD_53_01	NUMBER(5,0),	ccd_53_01	varchar(9)
CCD_54_01	NUMBER(5,0),	ccd_54_04	varchar(5)
CCD_54_02	VARCHAR2(200),	ccd_54_02	varchar(27)
CCD_54_03	VARCHAR2(200),	ccd_54_05	varchar(27)
CCD_54_04	VARCHAR2(200),	ccd_54_03	datetime
CCD_54_05	DATE,	ccd_54_01	varchar(16)
		ccd_54_06	varchar(27)

ДОДАТОК В

Лістинг методу синхронізації

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Net.Sockets;
using System.Net;
using System.Threading;
using System.Text.RegularExpressions;
using System.IO;

namespace DBServer
{
   / / Клас-обробник клієнта
   class Client
   {
      / / Відправка сторінки з помилкою
      private void SendError (TcpClient Client, int Code)
      {
         / / Отримуємо рядок виду "200 OK"
         / / HttpStatusCode зберігає в собі всі статус-коди HTTP/1.1
         string CodeStr = Code.ToString () + "" + ((HttpStatusCode) Code). ToString ();
         / / Код простий HTML-сторінки
         string Html = "<html> <body> <h1>" + CodeStr + "</ h1> </ body> </ html>";
         / / Необхідні заголовки: відповідь сервера, тип і довжина вмісту. Після двох
порожніх рядків - саме вміст
         string Str = "HTTP/1.1" + CodeStr + "\ nContent-type: text / html \ nContent-Length:" +
Html.Length.ToString () + "\ n \ n" + Html;
         / / Наведемо рядок до виду масиву байт
         byte [] Buffer = Encoding.ASCII.GetBytes (Str);
         / / Відправимо його клієнту
         Client.GetStream (). Write (Buffer, 0, Buffer.Length);
         / / Закриємо з'єднання
         Client.Close ();
      }

      / / Конструктор класу. Йому потрібно передавати прийнятого клієнта від TcpListener
      public Client (TcpClient Client)
      {
         / / Оголосимо рядок, в якому буде зберігається запит клієнта
         string Request = "";
         / / Буфер для зберігання прийнятих від клієнта даних
         byte [] Buffer = new byte [1024];
         / / Змінна для зберігання кількості байт, прийнятих від клієнта
         int Count;
         / / Читаємо з потоку клієнта до тих пір, поки від нього надходять дані
         while ((Count = Client.GetStream (). Read (Buffer, 0, Buffer.Length))> 0)
         {
            / / Перетворимо ці дані в рядок і додамо її до змінної Request
            Request + = Encoding.ASCII.GetString (Buffer, 0, Count);
```

```
            / / Запит повинен обриватися послідовністю \ r \ n \ r \ n
            / / Або обриваємо прийом даних самі, якщо довжина рядка Request перевищує 4
кілобайти
            / / Нам не потрібно отримувати дані з POST-запиту (і т. п.), а звичайний запит
            / / По ідеї не повинен бути більше 4 кілобайт
            if (Request.IndexOf ("\ r \ n \ r \ n")> = 0 | | Request.Length> 4096)
            {
               break;
            }
         }

         / / Парс рядок запиту з використанням регулярних виразів
         / / При цьому відсікаємо всі змінні GET-запиту
         Match ReqMatch = Regex.Match (Request, @ "^ \ w + \ s + ([^ \ s \?] +) [^ \ S] * \ s +
HTTP /. * |");

         / / Якщо запит не вдався
         if (ReqMatch == Match.Empty)
         {
            / / Передаємо клієнту помилку 400 - невірний запит
            SendError (Client, 400);
            return;
         }

         / / Отримуємо рядок запиту
         string RequestUri = ReqMatch.Groups [1]. Value;

         / / Наводимо її до початкового вигляду, перетворюючи екрановані символи
         / / Наприклад, "% 20" -> ""
         RequestUri = Uri.UnescapeDataString (RequestUri);

         / / Якщо в рядку міститься двокрапка, передамо помилку 400
         / / Це потрібно для захисту від URL типу http://example.com/../../file.txt
         if (RequestUri.IndexOf ("..")> = 0)
         {
            SendError (Client, 400);
            return;
         }

         / / Якщо рядок запиту закінчується на "/", то додамо до неї index.html
         if (RequestUri.EndsWith ("/"))
         {
            RequestUri + = "index.html";
         }

         string FilePath = "www /" + RequestUri;

         / / Якщо в папці www не існує даного файлу, посилаємо помилку 404
         if (! File.Exists (FilePath))
         {
            SendError (Client, 404);
            return;
```

```
}

/ / Отримуємо розширення файлу з рядка запиту
string Extension = RequestUri.Substring (RequestUri.LastIndexOf ('.'));

/ / Тип вмісту
string ContentType = "";

/ / Намагаємося визначити тип вмісту з розширення файлу
switch (Extension)
{
   case ". htm":
   case ". html":
      ContentType = "text / html";
      break;
   case ". css":
      ContentType = "text / stylesheet";
      break;
   case ". js":
      ContentType = "text / javascript";
      break;
   case ". jpg":
      ContentType = "image / jpeg";
      break;
   case ". jpeg":
   case ". png":
   case ". gif":
      ContentType = "image /" + Extension.Substring (1);
      break;
   default:
      if (Extension.Length> 1)
      {
         ContentType = "application /" + Extension.Substring (1);
      }
      else
      {
         ContentType = "application / unknown";
      }
      break;
}

/ / Відкриваємо файл, страхуючись на випадок помилки
FileStream FS;
try
{
   FS = new FileStream (FilePath, FileMode.Open, FileAccess.Read, FileShare.Read);
}
catch (Exception)
{
   / / Якщо трапилася помилка, посилаємо клієнту помилку 500
   SendError (Client, 500);
   return;
```

```
        }

        / / Посилаємо заголовки
        string Headers = "HTTP/1.1 200 OK \ nContent-Type:" + ContentType + "\ nContent-
Length:" + FS.Length + "\ n \ n";
        byte [] HeadersBuffer = Encoding.ASCII.GetBytes (Headers);
        Client.GetStream (). Write (HeadersBuffer, 0, HeadersBuffer.Length);

        / / Ще не досягнуто кінець файлу
        while (FS.Position <FS.Length)
        {
           / / Читаємо дані з файлу
           Count = FS.Read (Buffer, 0, Buffer.Length);
           / / І передаємо їх клієнту
           Client.GetStream (). Write (Buffer, 0, Count);
        }

        / / Закриємо файл і з'єднання
        FS.Close ();
        Client.Close ();
     }
  }

  class Server
  {
     TcpListener Listener; / / Об'єкт, що приймає TCP-клієнтів

     / / Запуск сервера
     public Server (int Port)
     {
        Listener = new TcpListener (IPAddress.Any, Port); / / Створюємо "слухача" для
зазначеного порту
        Listener.Start (); / / Запускаємо його

        / / В нескінченному циклі
        while (true)
        {
           / / Приймається нових клієнтів. Після того, як клієнт був прийнятий, він
передається в новий потік (ClientThread)
           / / З використанням пулу потоків.
           ThreadPool.QueueUserWorkItem (new WaitCallback (ClientThread),
Listener.AcceptTcpClient ());

           / *
           / / Приймається нового клієнта
           TcpClient Client = Listener.AcceptTcpClient ();
           / / Створюємо потік
           Thread Thread = new Thread (new ParameterizedThreadStart (ClientThread));
           / / І запускаємо цей потік, передаючи йому прийнятого клієнта
           Thread.Start (Client);
           * /
        }
```

```
        }

        static void ClientThread (Object StateInfo)
        {
            / / Просто створюємо новий екземпляр класу Client і передаємо йому наведений до
класу TcpClient об'єкт StateInfo
            new Client ((TcpClient) StateInfo);
        }

        / / Зупинка сервера
        ~ Server ()
        {
            / / Якщо "слухач" був створений
            if (Listener! = null)
            {
                / / Зупинимо його
                Listener.Stop ();
            }
        }

        static void Main (string [] args)
        {
            / / Визначимо потрібне максимальну кількість потоків
            / / Нехай буде по 4 на кожен процесор
            int MaxThreadsCount = Environment.ProcessorCount * 4;
            / / Встановимо максимальну кількість робочих потоків
            ThreadPool.SetMaxThreads (MaxThreadsCount, MaxThreadsCount);
            / / Встановимо мінімальна кількість робочих потоків
            ThreadPool.SetMinThreads (2, 2);
            / / Створимо новий сервер на порту 80
            new Server (80);
        }
    }
}
```

ДОДАТОК Г

Копія публікації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

VII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ »

11–12 грудня 2019 року

ТЕРНОПІЛЬ
2019

А. Постолюк
Тернопільський національний технічний університет імені Івана Пулюя

ТЕОРЕТИЧНІ ЗАСАДИ СИНХРОНІЗАЦІЇ ТА РЕПЛІКАЦІЇ БАЗ ДАНИХ

UDC 004.415.5
A. Postolyuk
(Ternopil Ivan Puluj National Technical University, Ukraine)

THEORETICAL BACKGROUND OF DATABASE SYNCHRONIZATION AND REPLICATION

Синхронізація баз даних – ліквідація відмінностей між двома копіями даних. Під реплікацією баз даних розуміється механізм синхронізації вмісту декількох копій об'єкта (наприклад, вмісту бази даних). При реплікації зміни, зроблені в одній копії об'єкта, можуть бути поширені в інші копії. Синхронізація може бути синхронною або асинхронною. У випадку синхронної реплікації, якщо дана репліка оновлюється, всі інші репліки того ж фрагмента даних також повинні бути оновлені в одній і тій же транзакції. Це означає, що існує лише одна версія даних.

У більшості продуктів синхронна реплікація реалізується за допомогою тригерних процедур (можливо, прихованих і керованих системою). Але синхронна реплікація має той недолік, що вона створює додаткове навантаження при виконанні всіх транзакцій, в яких оновлюються репліки (крім того, можуть виникати проблеми, пов'язані з доступністю даних).

У випадку асинхронної реплікації оновлення однієї репліки поширюється на інші через деякий час, а не в тій же транзакції. Таким чином, при асинхронній реплікації вводиться затримка, або час очікування, протягом якого окремі репліки можуть бути фактично неідентичних (тобто визначення репліки виявляється не зовсім відповідним, оскільки ми не маємо справу з точними і своєчасно створеними копіями).

У більшості продуктів асинхронна реплікація реалізується за допомогою читання журналу транзакцій або постійної черги тих оновлень, які підлягають поширенню. Перевага асинхронної реплікації полягає в тому, що додаткові витрати реплікації не пов'язані з транзакціями оновлень, які можуть мати важливе значення для функціонування всього підприємства і пред'являти високі вимоги до продуктивності.

До недоліків цієї схеми відноситься те, що дані можуть виявитися несумісними (тобто несумісними з точки зору користувача). Іншими словами, надмірність може проявлятися на логічному рівні, а це, строго кажучи, означає, що термін контрольована надмірність в такому випадку не застосуємо.

Розглянемо коротко проблему узгодженості (або, швидше, неузгодженості). Справа в тому, що репліки можуть ставати несумісними в результаті ситуацій, які важко (або навіть неможливо) уникнути і наслідки яких важко виправити. Зокрема, конфлікти можуть виникати з приводу того, в якому порядку повинні застосовуватися оновлення. Наприклад, припустимо, що в результаті виконання транзакції A відбувається вставка рядка в репліку X, після чого транзакція B видаляє цей рядок, а також припустимо, що Y - репліка X. Якщо поновлення поширюються на Y, але вводяться в репліку Y в зворотному порядку (наприклад, через різні затримки при передачі), то транзакція B не знаходить в Y рядок, що підлягає видаленню, і не виконує свою дію, після чого транзакція A вставляє цей рядок. Сумарний ефект полягає в тому, що репліка Y містить зазначений рядок, а репліка X - ні. В цілому завдання усунення конфліктних ситуацій і забезпечення узгодженості реплік є досить складними.