МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кваліфікаційна наукова
праця на правах рукопису

ЛАВРІВ ОРЕСТ АНДРІЙОВИЧ

УДК 621.391

ДИСЕРТАЦІЯ

Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах

05.12.02 – телекомунікаційні системи та мережі
(шифр і назва спеціальності)

05 «Технічні науки»
(галузь знань)

Подається на здобуття наукового ступеня
доктора технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____
(підпис, ініціали та прізвище здобувача)

Науковий консультант –
Климаш Михайло Миколайович,
д.т.н., професор

Ідентичність всіх примірників дисертації
ЗАСВІДЧУЮ:
Вчений секретар спеціалізованої
вченої ради

/І.В. Демидов/

Львів – 2018

АНОТАЦІЯ

Лаврів О.А. Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора технічних наук за спеціальністю 05.12.02 «Телекомунікаційні системи та мережі» (172 – Телекомунікації та радіотехніка). – Національний університет «Львівська Політехніка» МОН України, Львів, 2018.

Дисертація присвячена аспектам структурно-функціонального синтезу та надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах з метою покращення значень метрик якості надання послуг шляхом оптимізації структури та показників ефективності функціонування телекомунікаційної підсистеми, а також підвищення захищеності, відмовостійкості та зниження тривалості відновлення працездатності інформаційної підсистеми.

Бурхливий розвиток галузі інформаційних технологій сьогодні нерозривно пов'язаний із телекомунікаційними системами, які забезпечують передавання інформації між компонентами розподілених інформаційних сервісів. Оскільки питома вага сервісів, функціонування яких не обмежується фізично єдиною обчислювальною системою, значно переважає частку централізованих сервісів, то роль телекомунікацій набуває особливого значення. Поряд із клієнт-серверними та пірінговими архітектурами розподілених інформаційно-телекомунікаційних систем, з'являються Cloud-системи, Fog-системи, а широке розповсюдження технологій Інтернету речей приводить до того, що на телекомунікаційні системи покладається функція підтримки універсального інформаційного простору З аналізу праць іноземних та вітчизняних учених за обраною тематикою випливає, що сьогодні існує протиріччя, яке полягає у відсутності методів керування ресурсами та потоками даних, які б дали змогу врахувати процеси комунікації між елементами розподілених сервісів на

прикладному рівні розподілених гетерогенних інформаційно-телекомунікаційних систем зі збереженням обсягів їх апаратного забезпечення.

Отже, розроблення методів та моделей надання послуг у гетерогенних розподілених інформаційно-телекомунікаційних системах для підвищення якості обслуговування та покращення структурно-функціональних параметрів і характеристик цих систем з метою узгодження взаємодії сервісного шару та шару передавання даних є актуальною невирішеною на сьогодні науковою проблемою.

У першому розділі роботи проведено аналіз впливу системної архітектури розподілених гетерогенних інформаційно-телекомунікаційних систем на їх метрики якості. Проаналізовано процеси комунікації у розподілених гетерогенних інформаційно-комунікаційних системах з урахуванням рівня застосунків. Проведений аналіз є основою для подальшого виконання завдань розроблення моделей компонентів досліджуваних інформаційно-телекомунікаційних систем та методів надання послуг в таких системах.

У другому розділі роботи запропоновано математичну модель пересилання запиту в гетерогенних сервісно-орієнтованих системах. Сформульовано завдання розподілу ресурсів гетерогенних інформаційно-комунікаційних систем між сервісними потоками. На основі територіально-залежної моделі обслуговування отримано множину координат абонентів у процесі моделювання, класифіковано користувачів за критерієм швидкості руху, що забезпечує можливість централізації керування обслуговуванням в моменти перевантажень фрагментів мережі. Отримані параметри процесів комунікації дають змогу охарактеризувати процес надання послуг у системі шляхом розрахунку імовірності втрат і системної доступності. Запропонований метод балансування навантаження з багатоетапним перемиканням точок присутності послуг збільшує доступність інформаційних ресурсів у періоди пікових навантажень. Втрати комунікаційних сеансів скоротились на 14% у порівнянні із звичайним режимом роботи системи. Ефективна конфігурація SDP (як структури взаємоз'єднання фізичних інтерфейсів, так і управління потоком) здійснена

шляхом розрахунку необхідних параметрів для максимізації продуктивності службової інфраструктури та забезпечення процесів передавання сервісних потоків з мінімальними затримками, що значно підвищує показники якості надання послуг масштабованої розподіленої сервісної системи та унеможливлює відмову в обслуговуванні, коли кількість активних користувачів та їх вимоги різко зростають. Завдання забезпечення потоків даних задовільним сервісом полягає у оптимальному комбінуванні фізичної і логічної структур телекомунікаційної мережі.

У третьому розділі роботи проведено дослідження конфігурацій мережевих топологій на основі теорії графів із їх інтегральним оцінюванням за методом аналізу ієрархій. Проведено дослідження архітектури телекомунікаційного каналу з підтримкою множини класів обслуговування, де кожен клас обслуговування характеризується високими вимогами до метрик QoS. Запропоновано розв’язок завдання розподілу ресурсів між мережними сервісними потоками за умови рівноправної конкуренції за доступні фізичні ресурси для оцінки максимальної кількості клієнтів, які можуть використовувати послугу з різними рівнями якості обслуговування. Завдання сформульовано в термінах теорії графів і розв’язано методом лінійного програмування для заданого графа мережі. Досліджено вплив логічної структури на наявність мережевих ресурсів. Запропонований метод дає змогу виявити надлишковість у фізичній структурі та зменшити вартість мережі або підвищити продуктивність мережі шляхом введення k-шляхової маршрутизації через невикористані канали.

У четвертому розділі роботи розроблено нову модель корпоративного клієнта інформаційно-комунікаційної системи надання послуг. Ця модель відрізняється від відомих тим, що реалізована у вигляді програмного узагальненого коду, який внаслідок ініціалізації його параметрами, дає змогу утворити як завгодно багато об’єктів, що представляють корпоративну клієнтську інфраструктуру. Розроблено нову модель спільної cloud інфраструктури. Ця модель реалізована у вигляді програмного коду, і як і попередня, базується на технології імітаційного моделювання. Вона дає змогу

представити спільну мультиклієнтську інфраструктуру у вигляді ініціалізованого параметрами програмного об’єкта, який поєднано із клієнтським програмним об’єктом. Разом із логікою надання послуг ці дві моделі формують розподілену гетерогенну інформаційно-телекомунікаційну систему надання послуг, поведінка якої під впливом користувацького навантаження досліджена у шостому розділі роботи. Розроблено та досліджено комплексну систему інформаційної безпеки у телекомунікаційному сегменті розподіленої гетерогенної інформаційно-телекомунікаційної системи. Результати показують, що досягнуто зниження впливу атак на інформаційно-телекомунікаційну систему більш ніж на 90%. Досліджено процес відновлення інфраструктури після катастрофи на базі ручної та автоматичної активності. Проведено оцінку тривалості відновлення в разі ручного та автоматичного відновлення на основі концепції IaaC. Отримані результати підтвердили переваги підходу IaaC для відновлення після аварій. Значення виграшу залежить від складності хмарної системи і може сягати декількох порядків.

У п’ятому розділі проведено аналіз процесу безперервної інтреграції та запропоновано архітектуру його реалізації. Запропонована архітектура може бути впроваджена в сучасних інформаційно-комунікаційних системах, оскільки доставка послуг включає в себе сервісну інтеграцію, а також передавання послуг кінцевому користувачеві. Запропоновано удосконалений метод повної впорядкованої групової розсилки, який дає можливість зберігати працездатність процесу синхронізації за умови, що повідомлення процесу-відправника втрачається. Удосконалення полягає у обмеженні часу очікування на відповідь від усіх процесів-адресатів. Це забезпечує зниження імовірності відмови процесу. Адекватність моделі програмного компонента перевірено дослідженням процесу обслуговування мультимедійних інформаційних потоків апаратним маршрутизатором та програмним компонентом гетерогенної розподіленої інформаційно-телекомунікаційної системи. Максимальне відхилення затримки пакетів між програмним та апаратним компонентами знаходиться в межах 0,5 %. Доведено, що балансування навантаження за

допомогою інтегрованої системи керування розгортанням інформаційних сервісів дає змогу зменшити тривалість обслуговування запитів у середньому у 3 рази. Для дослідження затримок обслуговування запитів у розподіленій системі з балансуванням навантаження згідно згаданого методу розроблено модель пересилання потоків IPTV та виведено співвідношення для оцінки затримки для каналів зі змінною пропускною здатністю. Наведено порівняльні оцінки затримки, які дають змогу стверджувати, що зростання пропускної здатності каналу у 10 разів приводить до зниження затримки обслуговування запиту у середньому у 2,5 рази, що пояснюється обмеженнями методу балансування навантаження.

У шостому розділі роботи представлено методи розгортання cloud інфраструктури на основі PaaS. Подано характеристики надання послуг в інформаційно-телекомунікаційних системах, побудованих на основі cloud сервісів. Запропоновано удосконалену трьохрівневу архітектуру інформаційно-телекомунікаційних систем із резервуванням, побудованих із застосуванням cloud сервісів. Представлено концепцію локального кешування вмісту баз даних із застосуванням принципу стано-незалежних серверів. Проведено дослідження доступності cloud-сервісів у інформаційно-телекомунікаційних системах із багатокомпонентною cloud-інфраструктурою. Отримано оцінки середньої та ефективної доступності та їх залежності від опрацьованого навантаження та кількості фізичних серверів у системі. Отримані результати характеризують поведінку cloud системи за умови незмінності програмного коду cloud сервісів, які формують програмну частину системи. Здійснено навантажувальне тестування досліджуваної розподіленої гетерогенної інформаційно-телекомунікаційної системи, яка базується на cloud інфраструктурі. За результати навантажувального тестування виявлено вузьке місце у досліджуваній системі, а також отримано оцінки використання інфраструктурних компонентів системи і значень метрик якості обслуговування та якості сприйняття послуг користувачами досліджуваної інформаційно-телекомунікаційної системи. Показано, що оновлення програмного коду системи

робить неможливим розроблення та використання імітаційних універсальних моделей системи у цілому, оскільки навіть незначні зміни програмного коду, який логічно поєднує компоненти системи, призводить до зміни поведінки системи у цілому, що не може бути враховано як у аналітичній, так і у імітаційній моделях. У зв'язку з цим запропонована методика дослідження розподілених гетерогенних інформаційно-телекомунікаційних систем на базі cloud інфраструктури є важливим інструментом у процесі їх проектування та експлуатації та дозволяє у цілому вирішити проблему гармонізації рівнів запропонованої концептуальної структури інформаційно-телекомунікаційних систем із покращенням значень метрик якості обслуговування та оптимальним розподілом ресурсів таких систем.

Ключові слова: розподілена гетерогенна інформаційно-телекомунікаційна система, балансування навантаження, якість надання послуг, навантажувальне тестування, cloud інфраструктура, міжрівнева взаємодія.

Список публікацій здобувача:

Наукові праці, в яких опубліковані основні наукові результати дисертації:

[1] B. Buhyl, P. Huskov, O. Lavriv, R. Bak, A. Luntovskyy, "Maximization of Service Flows Rates as a Solution of Network Capacity Allocation Problem," *Internet Things Eng. Appl.*, vol. 3, no. 1, pp. 1–10, May 2018.

[2] P. Yatskiv, O. Lavriv, Z. Kharkhalis, V. Mosorov, and M. Niedźwiedziński, "Security. Provisions for Ensuring Telecommunication Networks' Protection from Malicious Influences," *Przedsiębiorczość i Zarządzanie*, vol. 17, no. 11, pp. 111–125, 2016.

[3] Р. П. Кришталь, О. А. Лаврів, З. М. Хархаліс, "Удосконалення алгоритму повного групового розсилання у системах оброблення даних з розподіленою архітектурою," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 849, С. 241–247, 2016.

[4] Р. І. Бак, П. О. Гуськов, О. А. Лаврів, “Імітаційна макромодель поведінки абонентів у мережі коміркового зв’язку,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 849, С. 274–284, 2016.

[5] M. Beshley, M. Seliuchenko, O. Lavriv, V. Chervenets, H. Kholiavka, M. Klymash, “Increasing the efficiency of real-time content delivery by improving the technology of priority assignment and processing of IP traffic,” *Smart Comput. Rev.*, vol. 5, no. 2, pp. 1–13, 2015.

[6] O. Shpur, M. Klymash, M. Seliuchenko, B. Strykhaliuk, O. Lavriv, “Improving the Quality of Composite Services Through Improvement of Cloud Infrastructure Management,” *Int. J. Comput. Sci. Inf. Secur.*, vol. 13, no. 9, pp. 36–44, 2015.

[7] М. І. Бешлей, М. О. Селюченко, О. А. Лаврів, А. Р. Масюк, Г.В. Холявка, “Оцінка адекватності функціонування програмного маршрутизатора у процесі обслуговування мультимедійного трафіку,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 818, С. 162–173, 2015.

[8] M. M. Klymash, H. A. Al-Zayadi, O. A. Lavriv, “Improving Thoughput Using Channel Quality Indicator in LTE Technology,” *Збірник наукових праць Донецького національного технічного університету, серія “Обчислювальна техніка та автоматизація,”* №. 1(26), С. 134–144, 2014.

[9] М. М. Климаш, М. І. Бешлей, Б. М. Стрихалюк, О. А. Лаврів, Г.В. Холявка,“Підвищення якості обслуговування в конвергентних мобільних системах на основі платформи UMA-A,” *Проблеми телекомунікацій*, №. 1(13), С. 3–19, 2014.

[10] М. І. Бешлей, В. П. Ткачук, Б. А. Бугиль, О. А. Лаврів, “Модель системи динамічного управління пропускною здатністю каналу інтегрованої WI-FI/GSM мережі,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 796, С. 83–96, 2014.

[11] H. A. Al-Zayadi, M. M. Klymash, O. A. Lavriv, M. Al-Shuraifi, “Mobility Affected on Channel Estimation Using Different Modulation in LTE,” *Проблеми телекомунікацій*, no. 2(14), pp. 30–41, 2014.

[12] O. Yaremko, B. Stryhalyuk, T. Maksymyuk, O. Lavriv, D. Kozhurov, “The optimal power control method in multiuser cellular networks,” *An Int. Q. J. Econ. Technol. New Technol. Model. Process.*, vol. 2, no. 1, pp. 63–67, 2013.

[13] Б. А. Бугиль, О. А. Лаврів, М. І. Бешлей, В. В. Червенець, “Методи оптимізації фізичної та логічної структур телекомунікаційних мереж,” *Вісник* Національного *університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 766, С. 76–81, 2013.

[14] І. М. Кузь, З. М. Хархаліс, П. О. Гуськов, О. А. Лаврів, “Дослідження методів частотного планування коміркових мереж на основі технологій LTE та GSM,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 766, С. 82–87, 2013.

[15] Y. Hayali, O. Lavriv, B. Buhyl, R. Bak, M. Klymash, “Models and mechanisms for traffic balancing of IPTV VoD service,” *Int. J. Serv. Econ. Manag.*, vol. 5, no. 4, pp. 291–300, 2013.

[16] М. М. Климаш, Б. А. Бугиль, О. А. Лаврів, Т. В. Корецький, “Інтегральна оцінка ефективності вибору маршрутів передавання потоків даних для різних конфігурацій мережевих топологій,” *Вісник Національного університету “*Львівська *політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 738, С. 95–99, 2012.

[17] О. А. Лаврів, М. І. Бешлей, М. М. Гнатчук, А. В. Поліщук, “Модель системи управління ресурсами мультисервісних мереж в умовах самоподібності трафіку,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та* телекомунікації*,”* №. 738, С. 165–172, 2012.

[18] M. M. Klymash, I. V. Demydov, O. A. Lavriv, Y. D. Dobush, “Analysis of service workflows distribution and service delivery platform parameters,” *Системи обробки інформації*, №. 6(104), С. 103–107, 2012.

[19] О. А. Лаврів, М. І. Бешлей, Б. А. Бугиль, “Дослідження якості надання послуг абонентам мобільного зв’язку в IMS-мережі,” *Збірник наукових праць Інститут проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України*, № 65, С. 132–140, 2012.

[20] Б. А. Бугиль, М. М. Климаш, О. А. Лаврів, І. В. Демидов, “Підвищення ефективності розподілу ресурсів телекомунікаційної мережі шляхом зміни маршрутів передавання даних,” *Проблеми телекомунікацій*, №. 4(9), С. 32–44, 2012.

[21] M. Klymash, Y. Hayali, O. Lavriv, “Models and mechanisms of IPTV VoD Traffic Balancing,” *Comput. Probl. Electr. Eng.*, vol. 2, no. 2, pp. 41–45, 2012.

[22] О.А. Лаврів, “Сервісно-орієнтоване планування ресурсів мультисервісної телекомунікаційної мережі,” *Збірник наукових праць Інститут проблем* моделювання *в енергетиці ім. Г.Є. Пухова НАН України*, №. 62, С. 118–126, 2012.

[23] M. Klymash, O. Lavriv, B. Buhil, “Service-oriented Resource Planning for Multiservice IP Networks,” *Comput. Probl. Electr. Eng.*, vol. 2, no. 1, pp. 59–63, 2012.

[24] М. Климаш, А. Ложковський, О. Лаврів, Р. Бак, “Графо-аналітичний підхід до порівняння ефективності систем безпровідного зв’язку,” *Комп’ютерні технології друкарства*, №. 27, С. 189–193, 2012.

[25] М. М. Климаш, О. А. Лаврів, І. О. Кагало, Б. В. Коваль, Т. А. Максимюк, “Покращення параметрів радіоінтерфейсу LTE/HSOPA,” *Комп’ютерні технології* друкарства, №. 26, С. 130–137, 2011.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

[26] Y. Hayali, M. Klymash, O. Lavriv, “Analysis of IPTV VoD Traffic Balancing Mechanisms,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 140–142.

[27] М. М. Климаш, О. А. Лаврів, М. Бешлей, “Моделювання та прогнозування параметрів і структур мультисервісної мережі з

диференційованим обслуговуванням послуг," в *VI Міжнародний науково-технічний симпозіум «Нові технології в телекомунікаціях»*, 2013, С. 28–31.

[28] М. М. Климаш, Б. А. Бугиль, О. А. Лаврів, "Дослідження ефективності топологій телекомунікаційних мереж на основі їх інтегральної оцінки за методом аналізу ієрархій," в *Сучасні інформаційно-комунікаційні технології COMINFO'2012*, 2012, С. 46–48.

[29] О. А. Лаврів, Б. А. Бугиль, І. Д. Орлевич, "Архітектура і програмна концепція телекомунікаційної платформи SDN – OPENFLOW," в *Науково-методична конференція "Сучасні проблеми телекомунікацій і підготовка фахівців в галузі телекомунікацій – 2012,"* 2012, С. 13–15.

[30] H. A. Al-Zayadi, O. Lavriv, M. Klymash, "QoE Estimation on the Basis of LTE Service Architecture," in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 590–592.

[31] М. І. Бешлей, О. А. Лаврів, Г. В. Холявка, "Дослідження методів побудови конвергентної мережі оператора мобільного зв'язку для надання послуг quad play," в *Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій*, 2014, С. 76–77.

[32] О. В. Красько та О. А. Лаврів, "Адаптація смуги пропускання в мережах NG-SDH/WDM," у Всеукраїнська науково-практична конференція "Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014," 2014, С. 201–204.

[33] M. Seliuchenko, A. Kovalchuk, O. Lavriv, and M. Klymash, "Enhancing Reliability of Transport Software-Defined Networks Using Flow Table Mutual Reservation Method," in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 573–575.

[34] І. В. Демидов, О. А. Лаврів, О. М. Шпур, and М. О. Селюченко, "Доступність композитних застосувань у сервісно-орієнтованих системах," в

Вимірювальна та обчислювальна техніка в технологічних процесах, 2014, С. 119–122.

[35] З. М. Хархаліс and О. А. Лаврів, “Порівняльний аналіз методів побудови мультиоператорних мереж мобільного зв’язку,” *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, С. 249–252.

[36] M. Klymash, M. Beshley, and O. Lavriv, “Model of Network Resources Management on the Basis of Services Priorities Association,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 164–166.

[37] O. Lavriv, I. Kahalo, and R. Kolodiy, “Application of NoSQL approach in data-centered network architectures: big data case,” in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, pp. 103–105.

[38] О. А. Лаврів, З. М. Хархаліс, and В. Я. Мосоров, “Моделювання DDoS атаки на веб-сервер,” в *10-а Міжнародна науково-технічна конференціїя "Проблеми телекомунікацій – 2016"*, 2016, С. 348–351.

[39] M. Al-Shuraifi, H. Al-Zayadi, O. Lavriv, and M. Klymash, “Improving QoS in MAX C/I Scheduling Using Resource Allocation Type 1 of LTE,” in *13 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2015, pp. 12–14.

[40] Д. В. Кожуров, О. А. Лаврів, and А. Р. Масюк, “Модель обміну керуючою інформацією в сервісно-орієнтованій Cloud-мережі,” в *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, С. 105–108.

[41] О. А. Лаврів, Б. А. Бугиль, П. О. Гуськов, Р. Р. Баса, “Розподіл ємності телекомунікаційної мережі із максимізацією інтенсивності інформаційних потоків,” в *“Фізико-технологічні проблеми передавання, оброблення та зберігання інформації в інфокомунікаційних системах,”* 2016, С. 134–135.

[42] О. А. Лаврів, Б. М. Стрихалюк, О. М. Шпур, and Т. А. Максимюк, “Дистанційний моніторинг параметрів місця вчинення злочину з використанням мобільних технологій,” в *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, С. 67–69.

[43] Б. М. Стрихалюк, О. А. Лаврів, З. М. Хархаліс, “Проблематика розгортання SDP для операторів мобільного зв’язку на основі віртуалізованих мереж дата-центрів,” в *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, С. 221–224.

[44] М. М. Климаш, М. О. Селюченко, О. А. Лаврів, “Забезпечення відмовостійкості багаторівневої ієрархії управління у програмно-керованих мережах,” в Всеукраїнська *науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, С. 225–228.

[45] O. Lavriv, M. Klymash, G. Grynkevych, O. Tkachenko, V. Vasylenko, “Method of cloud system disaster recovery based on ‘Infrastructure as a code’ concept,” in *14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 1139–1142.

[46] R. Bak, O. Lavriv, B. Koval, “Load balancing based on multi-hop handover for wireless cellular networks,” in *2017 IEEE 1st Ukraine Conference on Electrical and Computer Engineering, UKRCON 2017 - Proceedings*, 2017, pp. 1103–1106.

[47] O. Krasko, M. Klymash, O. Lavriv, “Data flows transmission models in converged optical access networks,” in 2015 2nd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2015 - Conference Proceedings, 2015, pp. 65–68.

[48] O. Lavriv, Z. Kharkhalis, B. Strykhaliuk, “Concept of intrusion detection system with implementation of topological sorting,” in *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings*, 2017, pp. 170–172.

[49] M. Seliuchenko, O. Lavriv, O. Panchenko, V. Pashkevych, “Enhanced multi-commodity flow model for QoS-aware routing in SDN,” in *2016 IEEE International Scientific Conference “Radio Electronics and Info Communications”, UkrMiCo 2016 - Conference Proceedings*, 2016, pp. 1–3.

[50] I. Demydov, O. Lavriv, Z. Kharkhalis, M. M. El-Hatri, “Concept of the migrating firewall to scalable cloud networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET 2016*, 2016, pp. 643–645.

[51] P. Huskov, O. Lavriv, M. Klymash, “The concept of services assurance in heterogeneous service-oriented systems,” in 2016 3rd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2016 - Proceedings, 2017, pp. 24–26.

[52] O. Lavriv, B. Buhyl, P. Huskov, R. Bak, “Heterogeneous network capacity distribution among service flows,” in 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings, 2017, pp. 173–175.

[53] A. Masiuk, M. Beshley, O. Lavriv, Y. Deschynskiy, “Common radio resource management model for heterogeneous cellular networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of* the *13th International Conference on TCSET 2016*, 2016, pp. 661–663.

[54] O. Lavriv, B. Buhyl, M. Klymash, G. Grynkevych, “Services continuous integration based on modern free infrastructure,” in *2017 2nd International Conference on Advanced* Information *and Communication Technologies (AICT)*, 2017, pp. 150–153.

[55] M. O. Seliuchenko, O. A. Lavriv, M. M. Klymash, “Analysis of load balancing methods for improving of utilization efficiency of hardware resources of distributed systems,” in *Microwave and Telecommunication Technology (CriMiCo), 2013 23rd International Crimean Conference*, 2013, pp. 515–516.

[56] H. Al-Zayadi, O. Lavriv, M. Klymash, A.-S. Mushtaq, “Increase throughput by expectation channel quality indicator,” in *2014 First International Scientific-*

Practical Conference Problems of Infocommunications Science and Technology, 2014, pp. 120–121.

[57] M. Klymash, O. Lavriv, T. Maksymyuk, M. Beshley, “State of the art and further development of information and communication systems,” in *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, 2016, pp. 1–6.

ABSTRACT

Lavriv O.A. Methods and models of service provision in heterogeneous distributed information and telecommunication systems. – Qualifying scientific paper on the rights of manuscripts.

A thesis submitted in fulfilment of the Doctor of Engineering Science degree in technical sciences on specialty 05.12.02 «Telecommunication systems and networks» (172 – Telecommunications and Radioengineering). – Lviv Polytechnic National University of Ministry for Education and Science of Ukraine, Lviv, 2018.

The dissertation is devoted to the aspects of structural and functional synthesis and services provision in heterogeneous distributed information and telecommunication systems in order to improve the metrics of the delivery service quality by optimizing the structure and indicators of the operational efficiency of the telecommunication subsystem, as well as increasing the security, fault tolerance and reducing the recovery duration of the information subsystem.

The rapid development of information technology today is inextricably connected with telecommunication systems that provide the transmission of information between components of distributed information services. Since the partition of services, which are not limited to the only computer system physically prevailing the paartition of centralized services, the role of telecommunications becomes of particular importance. Along with the client-server and peer-to-peer architecture of distributed information and telecommunication systems, Cloud-systems, Fog-systems appear, and the widespread of Internet technology leads to the fact that telecommunication systems have the function of supporting the universal information space. From the analysis of

the works of foreign and Ukrainian scientists on selected topics, it follows that today there is a contradiction between the development of information technology and telecommunication systems, which consists in the use of outdated methods for managing the resources and data flows. These methods do not consider the requirements of communication between the elements of distributed services at the application layer.

Consequently, the development of methods and models for providing services in heterogeneous distributed information and telecommunication systems to improve the quality of service and improve the structural and functional parameters and characteristics of such systems in order to reconcile the interaction of service layer and data transmission layer is an urgent scientific problem that remains unsolved today.

The first chapter analyzes the influence of the system architecture of distributed heterogeneous information and telecommunication systems on their quality metrics. The processes of communication in distributed heterogeneous information-communication systems are analyzed, considering the application layer influence. The analysis performed is the basis for further implementation of the tasks of developing models of components of the investigated information and telecommunication systems and methods of providing services in such systems.

The second chapter of the thesis proposes a mathematical model for query forwarding in heterogeneous service-oriented systems. The problem of resources allocation of heterogeneous information and communication systems between service flows is stated. Based on the area-based model of service delivery, a set of user's coordinates was calculated, users are classified by a speed criterion, and that provides the centralizing service management under the overloads of network segments. The calculated parameters of communication processes allow to characterize the process of providing services in the system by calculating the losses probability and system availability. The proposed method of load balancing with multi-stage switching of service points increases the availability of information resources during periods of peak load. Number of lost communication sessions were reduced by 14% compared to the normal system operation. Effective SDP configuration (both physical and flow

management interfaces) is carried out by calculating the required parameters to maximize the performance of the service infrastructure and ensure the processes of transmission of service flows with minimal delays, which significantly increases the quality of service delivery in scalable distributed service system and eliminates service discontinuity when the number of active users and their demands are sharply increasing. The task of providing data flows with satisfactory service is to optimally combine the physical and logical network structures.

The third chapter contains the research of configurations of network topologies based on the theory of graphs. The routing tasks for modern telecommunication networks can be stated and solved using methods of algebraic topology. The research of the architecture of a telecommunication channel supported by a set of service classes was performed. The solution of the resource allocation problem between network service flows is proposed in case of equal competition for available physical resources. The goal is to estimate the maximum number of clients who can use the service with different levels of service quality. The problem is formulated in graph theory terms and solved by the method of linear programming for a given network graph. The influence of the logical structure on the availability of network resources is investigated. The proposed method detects redundancy in the physical structure and reduces the cost of the network or changes the routing policy by introducing *k*-route routing through unused channels, and as a result – by increasing network performance.

The fourth chapter presents a new model of the corporate client of the information and communication system. A new model of shared cloud infrastructure has been developed. Both models are implemented as a software and allow to simulate the system performance without its actual building. In combination with the system logic of service provision, these two models form a distributed heterogeneous information and telecommunication system for providing services. The behavior of such system under high user load is investigated in the sixth chapter of the thesis. The information security system as IaaS service in the telecommunication segment of the distributed heterogeneous information and telecommunication system was developed and investigated. The results show that the impact of attacks on the information and

telecommunication system has been reduced by more than 90%. The process of infrastructure recovery after catastrophes based on manual and automatic activity is explored. We have estimated the recovery duration in the case of manual and automatic recovery based on the IaaC. The results confirmed the benefits of the IaaC approach to recovery after accidents. The benefit ration may be by an order.

The fifth chapter analyses the process of continuous integration and the architecture of its implementation. The proposed architecture can be implemented in modern information and communication systems, as the delivery of services includes service integration, as well as the transfer of services to the end user. We offered the advanced method of fully ordered group communication, which keeps the synchronization process operational, even if the message of the sender process is lost. The improvement is to limit time-outs for all recipient processes. This reduces the probability of the process failure. The adequacy of the software component model was verified by studying the routing of multiservice information flow with the hardware router and the software component of the heterogeneous distributed information and telecommunication system. Maximum deviation of packet delays between software and hardware components is within 0,5%. We proved that load balancing in the integrated information service deployment management system could reduce the average duration of queries up to 3 times. In order to investigate the delays in queuing service in a distributed load balancing system according to the above method, an IPTV streaming model was developed and a delay estimation for variable bandwidth channels was performed. The comparative estimations of the delay are given, which allow to assert that the increase of the channel capacity by 10 times leads to up to 2,5 time decrease in the service delivery delay, which is explained by the restrictions of the load balancing method.

The sixth chapter of the thesis presents the methods for deploying cloud infrastructure based on the PaaS principle. The characteristics of providing services in the information and telecommunication systems, built on the basis of cloud services are given. The advanced three-layered architecture of information and telecommunication systems with is offered. The concept of local caching of database

contents with the use of the principle of stateless servers is presented The study of the of cloud services availability in information and telecommunication systems with multicomponent cloud infrastructure has been conducted. Estimates of average and effective availability and their dependence on the service load and the number of physical servers in the system are obtained. The obtained results characterize the behavior of the cloud system under the condition that the software of the cloud services is unmutable. We performed the load testing of the distributed heterogeneous information and telecommunication system based on the cloud infrastructure. The results of load testing revealed a bottleneck in the system. We also simulated utilization of infrastructure components and metrics of service quality and quality of experience. It has been shown that updating of the system code makes it impossible to design and use universal models of the system, since even minor changes to the code which logically combine the components of the system leads to a change in the behavior of the system, which cannot be considered in both analytical and simulation models. In this regard, the proposed methodology for the investigation of distributed heterogeneous information and telecommunication systems based on cloud infrastructure is an important tool in their design and operation and allows solving the problem of harmonizing the levels of the proposed conceptual structure of information and telecommunication systems with improvement of service quality metrics and optimal distribution of resources of such systems. This indicates the achievement of the thesis objectives.

Keywords: distributed heterogeneous information and telecommunication system, load balancing, quality of service provision, load testing, cloud infrastructure, inter-level interaction.

The list of author's publications:

Journals where basic scientific results of the thesis has been published:

[1] B. Buhyl, P. Huskov, O. Lavriv, R. Bak, A. Luntovskyy, "Maximization of Service Flows Rates as a Solution of Network Capacity Allocation Problem," *Internet Things Eng. Appl.*, vol. 3, no. 1, pp. 1–10, May 2018.

[2] P. Yatskiv, O. Lavriv, Z. Kharkhalis, V. Mosorov, and M. Niedźwiedziński, “Security. Provisions for Ensuring Telecommunication Networks’ Protection from Malicious Influences,” *Przedsiębiorczość i Zarządzanie*, vol. 17, no. 11, pp. 111–125, 2016.

[3] R.P. Kryshtal, O.A. Lavriv, Z.M. Kharkhalis, “Improving of fully ordered group communication protocol in data processing system with distributed architecture,” *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,”* No. 849, pp. 241–247, 2016.

[4] R.I. Bak, P.O. Huskov, O.A. Lavriv, “Simulation macromodel of subscriber behavior in communication cellular network,” *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,”* No. 849, pp. 274–284, 2016.

[5] M. Beshley, M. Seliuchenko, O. Lavriv, V. Chervenets, H. Kholiavka, M. Klymash, “Increasing the efficiency of real-time content delivery by improving the technology of priority assignment and processing of IP traffic,” *Smart Comput. Rev.*, vol. 5, no. 2, pp. 1–13, 2015.

[6] O. Shpur, M. Klymash, M. Seliuchenko, B. Strykhaliuk, O. Lavriv, “Improving the Quality of Composite Services Through Improvement of Cloud Infrastructure Management,” *Int. J. Comput. Sci. Inf. Secur.*, vol. 13, no. 9, pp. 36–44, 2015.

[7] M.I. Beshley, M.O. Seliuchenko, O.A. Lavriv, A.R. Masiuk, H.V. Kholiavka, “Estimation of adequacy of software router operation while serving multiservice traffic,” *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 818, pp. 162–173, 2015.

[8] M. M. Klymash, H. A. Al-Zayadi, O. A. Lavriv, “Improving Thoughput Using Channel Quality Indicator in LTE Technology,” Digest of scientific papers of *Donetsk National Technical University, Series of Computational Engineering and Automation,* No. 1(26), pp. 134–144, 2014.

[9] M. M. Klymash, M. I. Beshley, B. M. Strykhalyuk, O. A. Lavriv, H. V. Kholiavka, "Improving the quality of service in convergent mobile systems based on the UMA-A platform," *Problems of telecommunications*, No. 1 (13), pp. 3-19, 2014.

[10] M.I. Beshley, V.P. Tkachuk, B. A.A. Buhyl, O.A. Lavriv, "Model of system of dynamic control of bandwidth of an integrated WI-FI / GSM network channel," *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 796, pp. 83–96, 2014.

[11] H. A. Al-Zayadi, M. M. Klymash, O. A. Lavriv, M. Al-Shuraifi, "Mobility Affected on Channel Estimation Using Different Modulation in LTE," *Problems of telecommunications*, no. 2(14), pp. 30–41, 2014

[12] O. Yaremko, B. Stryhalyuk, T. Maksymyuk, O. Lavriv, D. Kozhurov, "The optimal power control method in multiuser cellular networks," *An Int. Q. J. Econ. Technol. New Technol. Model. Process.*, vol. 2, no. 1, pp. 63–67, 2013.

[13] B. A. Buhyl, O. A. Lavriv, M. I. Beshley, V. V. Chervenets, "Methods of optimization of physical and logical structures of telecommunication networks," *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 766, pp. 76–81, 2013.

[14] I. M. Kuz, Z. M. Kharkhalis, P. O. Huskov, O. A. Lavriv, "Investigation of Frequency Planning of Cellular Networks Based on LTE and GSM Technologies," *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 766, pp. 82–87, 2013.

[15] Y. Hayali, O. Lavriv, B. Buhyl, R. Bak, M. Klymash, "Models and mechanisms for traffic balancing of IPTV VoD service," *Int. J. Serv. Econ. Manag.*, vol. 5, no. 4, pp. 291–300, 2013.

[16] M.M. Klymash, B. A. Buhyl, O. A. Lavriv, T. V. Koretskiy, "Integral estimation of efficiency of choice of routes of data flow transmission for various configurations of network topologies," *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 738, pp. 95–99, 2012.

[17] O. A. Lavriv, M. I. Beshley, M. M. Hnatchuk, A. V. Polishchuk, "Model of management system of resources of multiservice networks in conditions of self-

similarity of traffic," *Herald of Lviv Polytechnic National University, Series of Radioelectronics and Telecommunications,* No. 738, pp. 165–172, 2012.

[18] M. M. Klymash, I. V. Demydov, O. A. Lavriv, Y. D. Dobush, "Analysis of service workflows distribution and service delivery platform parameters," *Information processing systems*, No. 6(104), pp. 103–107, 2012.

[19] O. A. Lavriv, M. I. Beshley, B. A. Buhyl, "Investigation of the provision services quality for subscribers of mobile communication in the IMS-network," *Digest of scientific works. Institute of Modeling Problems in Power Engineering n.a. G. E. Puhov, NAS of Ukraine*, No. 65, pp. 132-140, 2012.

[20] B. A. Buhyl, M. M. Klymash, A. A. Lavriv, I. V. Demydov, "Increasing the efficiency of distribution of resources of the telecommunication network by changing the routes of data transmission," *Problems of telecommunications*, No. 4 (9), pp. 32-44, 2012.

[21] M. Klymash, Y. Hayali, O. Lavriv, "Models and mechanisms of IPTV VoD Traffic Balancing," *Comput. Probl. Electr. Eng.*, vol. 2, no. 2, pp. 41–45, 2012.

[22] O.A. Lavriv, "Service-oriented planning of resources of the multiservice telecommunication network," *Digest of scientific works Institute for Modeling Problems in Energy n.a. G. E. Puhov, NAS of Ukraine*, no. 62, pp. 118-126, 2012.

[23] M. Klymash, O. Lavriv, B. Buhil, "Service-oriented Resource Planning for Multiservice IP Networks," *Comput. Probl. Electr. Eng.*, vol. 2, no. 1, pp. 59–63, 2012.

[24] M. Klymash, A. Lozhkovsky, O. Lavriv, R. Bak, "Graph-Analytical Approach to Comparison of the Efficiency of Wireless Communication Systems," *Computer Technologies of Printing*, No. 27, pp. 189-193, 2012.

[25] M. M. Klymash, O. A. Lavriv, I. O. Kahalo, B. V. Koval, T. A. Maksymyuk, "Improvement of Radio Interface Settings of LTE / HSOPA," *Computer Technologies of Printing*, No. 26, pp. 130-137, 2011.

Proceedings that certify an approvement of thesis materials:

[26] Y. Hayali, M. Klymash, O. Lavriv, “Analysis of IPTV VoD Traffic Balancing Mechanisms,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 140–142.

[27] M.M. Klymash, O. A. Lavrsv, M. Beshley, "Modeling and forecasting of parameters and structures of multiservice network with differentiated service maintenance", in *VI International Scientific and Technical Symposium "New Technologies in Telecommunications*", 2013, pp. 28-31.

[28] M. M. Klymash, B. A. Buhyl, O. A. Lavriv, "Research of the efficiency of telecommunications network topologies on the basis of their integral estimation by the method of hierarchy analysis", in *Modern information and communication technologies COMINFO'2012*, 2012, pp. 46-48.

[29] O. A. Lavriv, B. A. Buhyl, I. D. Orlevyh, "Architecture and Program Concept of the SDN-OPENFLOW Telecommunication Platform", in the *Scientific-Methodical Conference "Modern Problems of Telecommunications and Training of Specialists in the Field of Telecommunications - 2012*," 2012, pp. 13-15.

[30] H. A. Al-Zayadi, O. Lavriv, M. Klymash, “QoE Estimation on the Basis of LTE Service Architecture,” in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 590–592.

[31] M.I. Beshley, O.A. Lavrsv, H.V. Kholiavka, "Research of methods of constructing a convergent network of mobile communication operator for quad play services," in *Modern problems and achievements in the field of radio engineering, telecommunications and Information Technology*, 2014, pp. 76-77.

[32] O. V. Krasko and O. A. Lavriv, "Adaptation of the bandwidth in NG-SDH / WDM networks" at the *All-Ukrainian Scientific and Practical Conference "Modern Telecommunication Problems and Training of Telecommunication Specialists - 2014*," 2014, pp. 201-204.

[33] M. Seliuchenko, A. Kovalchuk, O. Lavriv, and M. Klymash, “Enhancing Reliability of Transport Software-Defined Networks Using Flow Table Mutual

Reservation Method," in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 573–575.

[34] I.V. Demidov, O.A. Lavriv, O.M. Shpur, and M.O. Seliuchenko, "Availability of composite applications in service-oriented systems," in *Measuring and computer technology in technological processes*, 2014, pp. 119–122.

[35] Z. M. Kharkhalis and O. A. Lavriv, "Comparative analysis of methods of construction of multioperational networks of mobile communication," in *All-Ukrainian scientific and practical conference "Modern problems of telecommunications and training of specialists in the field of telecommunications - 2013*," 2013, pp. 249-252.

[36] M. Klymash, M. Beshley, and O. Lavriv, "Model of Network Resources Management on the Basis of Services Priorities Association," in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 164–166.

[37] O. Lavriv, I. Kahalo, and R. Kolodiy, "Application of NoSQL approach in data-centered network architectures: big data case," in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT'2015)*, 2015, pp. 103–105.

[38] O. A. Lavriv, Z. M. Kharhalis, and V. Ya. Mosorov, "Modeling DDoS Attacks on a Web Server," in the *10th International Scientific and Technical Conference "Telecommunications Problems - 2016*", 2016, pp. 348-351.

[39] M. Al-Shuraifi, H. Al-Zayadi, O. Lavriv, and M. Klymash, "Improving QoS in MAX C/I Scheduling Using Resource Allocation Type 1 of LTE," in *13 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2015, pp. 12–14.

[40] D. V. Kozhurov, O. A. Lavriv, and A. R. Masiuk, "A Model for the Exchange of Control Information in a Service-Oriented Cloud Network," in the *All-Ukrainian Scientific and Practical Conference "Modern Problems of Telecommunications and Training of Specialists in Telecommunications Industry – 2013"*, 2013, pp. 105-108.

[41] O. A. Lavriv, B. A. Buhyl, P. O. Huskov, R. R. Basa, "Allocation of the capacity of a telecommunication network with maximization of the information flows rate," in *"Physical-technological problems of transmission, processing and storage information in infocommunication systems*," 2016, pp. 134-135.

[42] O. A. Lavriv, B. M. Strykhaliuk, O. M. Shpur, and T. A. Maksymyuk, "Remote monitoring of the parameters of the place of the crime with the use of mobile technologies," in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT'2015)*, 2015, pp. 67–69.

[43] B. M. Stryhalyuk, O. A. Lavriv, Z. M. Kharkhalis, "The Problem of Deployment of SDP for Mobile Telecommunication Operators on the Basis of Virtualized Datacenter Networks," in the *All-Ukrainian Scientific and Practical Conference "Modern Telecommunication Problems and Training of Telecommunication Specialists - 2014*," 2014, pp. 221-224.

[44] M.M. Klymash, M.O. Seliuchenko, O.A. Lavriv, "Providing Failover Reliability of the Multi-Level Management Hierarchy in Software-Defined Networks," in the *All-Ukrainian Scientific and Practical Conference "Modern Telecommunication Problems and Training of Specialists in the Field of Telecommunications – 2014*," 2014, pp. 225-228.

[45] O. Lavriv, M. Klymash, G. Grynkevych, O. Tkachenko, V. Vasylenko, "Method of cloud system disaster recovery based on 'Infrastructure as a code' concept," in *14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 1139–1142.

[46] R. Bak, O. Lavriv, B. Koval, "Load balancing based on multi-hop handover for wireless cellular networks," in *2017 IEEE 1st Ukraine Conference on Electrical and Computer Engineering, UKRCON 2017 - Proceedings*, 2017, pp. 1103–1106.

[47] O. Krasko, M. Klymash, O. Lavriv, "Data flows transmission models in converged optical access networks," in 2015 2nd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2015 - Conference Proceedings, 2015, pp. 65–68.

[48] O. Lavriv, Z. Kharkhalis, B. Strykhaliuk, “Concept of intrusion detection system with implementation of topological sorting,” in *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings*, 2017, pp. 170–172.

[49] M. Seliuchenko, O. Lavriv, O. Panchenko, V. Pashkevych, “Enhanced multi-commodity flow model for QoS-aware routing in SDN,” in *2016 IEEE International Scientific Conference “Radio Electronics and Info Communications”, UkrMiCo 2016 - Conference Proceedings*, 2016, pp. 1–3.

[50] I. Demydov, O. Lavriv, Z. Kharkhalis, M. M. El-Hatri, “Concept of the migrating firewall to scalable cloud networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET 2016*, 2016, pp. 643–645.

[51] P. Huskov, O. Lavriv, M. Klymash, “The concept of services assurance in heterogeneous service-oriented systems,” in 2016 3rd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2016 - Proceedings, 2017, pp. 24–26.

[52] O. Lavriv, B. Buhyl, P. Huskov, R. Bak, “Heterogeneous network capacity distribution among service flows,” in 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings, 2017, pp. 173–175.

[53] A. Masiuk, M. Beshley, O. Lavriv, Y. Deschynskiy, “Common radio resource management model for heterogeneous cellular networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of* the *13th International Conference on TCSET 2016*, 2016, pp. 661–663.

[54] O. Lavriv, B. Buhyl, M. Klymash, G. Grynkevych, “Services continuous integration based on modern free infrastructure,” in *2017 2nd International Conference on Advanced* Information *and Communication Technologies (AICT)*, 2017, pp. 150–153. (Scopus)

[55] M. O. Seliuchenko, O. A. Lavriv, M. M. Klymash, “Analysis of load balancing methods for improving of utilization efficiency of hardware resources of

distributed systems," in *Microwave and Telecommunication Technology (CriMiCo), 2013 23rd International Crimean Conference*, 2013, pp. 515–516. (Scopus)

[56] H. Al-Zayadi, O. Lavriv, M. Klymash, A.-S. Mushtaq, "Increase throughput by expectation channel quality indicator," in *2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology*, 2014, pp. 120–121.

[57] M. Klymash, O. Lavriv, T. Maksymyuk, M. Beshley, "State of the art and further development of information and communication systems," in *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, 2016, pp. 1–6.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API – Application Programmable Interface – Прикладний програмний інтерфейс;

ASP – Application Service Provider – Провайдер надання послуг;

CI – Continuous Integration – безперервна інтеграція інформаційно-телекомунікаційних систем;

DaaS – Data as a Service – Дані (та їх оброблення), як сервіс;

DC – Data Center – ЦОД;

DDoS – Distributed DoS – Розподілена версія мережної атаки виду DoS;

DHT – Distributed Hash Table – Розподілена таблиця з хеш-ключами;

DMZ – Демілітаризована зона;

DoS – Denial of Service - Відмова у обслуговуванні внаслідок перевантаження запитами сервісної мережної платформи;

ESC – Elementary Service Components – Елементарні сервісні компоненти;

GoP – Group of Pictures – Група кадрів у потоці IPTV

HTTP – Hyper Text Transfer Protocol - Протокол передачі гіпертекстових документів;

HTTPS – HTTP протокол із додатковим шаром шифрування/аутентифікації;

HV – Hipervisor – Програмний гіпервізор;

IaaC – Infrastructure as a Code – Інфраструктура як код;

IaaS – Infrastructure as a Service – Інфраструктура як сервіс;

IDS – Intrusion Detection System – Система виявлення функціональних втручань;

IoT – Internet of Things – Концепція «Інтернету речей»;

IP – Internet Protocol – Інтернет протокол;

IPS – Intrusion Prevention System – Система запобігання функціональним втручанням;

JSON – JavaScript Object Notation – текстовий формат обміну даними між комп'ютерами;

LTE – Long Term Evolution - довготерміновий розвиток систем мобільного зв’язку (група концепцій та стандартів);

NAI – Network Access Interface – Інтерфейс доступу абонента до мережі;

NFV – Network Functions Virtualization – Віртуалізація мережних функцій;

PaaS – Platform as a Service – Платформа, як сервіс;

PF – Packet Filters – Фільтрування пакетів;

PLR – Process Level Redundancy – Надлишковість рівня процесів;

QoE – Quality of Expirience – Якість сприйняття сервісу;

QoS – Quality of Service – Якість сервісу;

RAN – Radio Access Network – Радіомережа доступу;

RDBMS – Relational Database Management System – Система керування реляційними базами даних;

REST – Representational State Transfer – Передавання репрезентативного стану;

RPM – Red Hat Package Manager – Менеджер компонентів ред-хет подібних лінукс систем;

SaaS - Software as a Service – Програмне забезпечення як сервіс;

SAP – Service Access Point – Точка доступу до сервісу;

SDN – Software-defined Networking – Програмно-конфігурована мережа;

SDP – Service Delivery Platform – Сервісна платформа;

SI – Service Interface – Сервісний інтерфейс

SLA – Service Level Agreement – Угода про рівень якості обслуговування;

SOA – Service-Oriented Architecture – Сервісно орієнтована архітектура;

SOAP – Simple Object Access Protocol - Протокол обміну структурованими повідомленнями в розподілених обчислювальних системах;

URI - Uniform Resource Identifier - Уніфікований ідентифікатор ресурсів;

VCS – Version Control System – Система контролю версій;

VM – Virtual Machines – Віртуальні машини;

VoD – Video on Demand – Відео за запитом;

VoIP – Voice over IP –сервіс IP-телефонії;

WSDL – Web Services Description Language — Мова опису зовнішніх інтерфейсів веб-сервісу;

XML – Extensible Markup Language – Розшиирювана мова розмітки;

ІТС – Інформаційно-телекомунікаційна система;

ОС – операційна система;

СУБД – Система управління базами даних.

ВСТУП

Загальна характеристика роботи. Дисертація присвячена аспектам структурно-функціонального синтезу в процесі надання послуг у гетерогенних розподілених інформаційно-телекомунікаційних системах з метою покращення значень метрик якості надання послуг шляхом оптимізації структури та показників ефективності функціонування телекомунікаційної підсистеми, а також підвищення захищеності, відмовостійкості та зниження тривалості відновлення працездатності інформаційної підсистеми.

Актуальність роботи. Бурхливий розвиток галузі інформаційних технологій сьогодні нерозривно пов'язаний із телекомунікаційними системами, які забезпечують передавання інформації між компонентами розподілених інформаційних сервісів. Оскільки питома вага сервісів, функціонування яких не обмежується фізично єдиною обчислювальною системою, значно переважає частку централізованих сервісів, то роль телекомунікацій набуває особливого значення. Поряд із клієнт-серверними та піринговими архітектурами розподілених інформаційно-телекомунікаційних систем, з'являються Cloud-системи, Fog-системи, а широке розповсюдження технологій Інтернету речей приводить до того, що на телекомунікаційні системи покладається функція підтримки універсального інформаційного простору. Зважаючи на те, що поява і запровадження згаданих технологій стимулює розвиток фактично необмеженої множини сервісів, це потребує нового підходу до управління ресурсами та потоками даних в телекомунікаційній площині. Необхідно враховувати взаємодію сервісів на прикладному рівні, яка формує вимоги до відповідної взаємодії на рівні телекомунікаційної мережі. Для цього слід розробити нові методи та моделі надання послуг в інформаційно-телекомунікаційних системах.

Проблематикою надання послуг в розподілених гетерогенних інформаційно-телекомунікаційних системах займається багато українських та закордонних вчених: питання керування мережами – Беркман Л.Н., Толубко В.Б., Глоба Л.С., Теленик С.Ф., Andriy Luntovskiy, Alexander Schill, Толюпа С.В. та інші; сервісно-адаптовані інформаційно-телекомунікаційні

системи вивчають у своїх роботах Ложковський А.Г., Стрихалюк Б.М., Лемешко О.В., Natalia Kryvinska, Christine Strauss, Yu C.Z. тощо; керування ресурсами та структурний синтез мереж досліджують Агеєв Д.В., Воробієнко П.П., Гаркуша С.В., Jo M., Schmidt H., Walter F. Witt, Zhao X., Bloomers J. та інші.

З аналізу праць іноземних та вітчизняних учених за обраною тематикою випливає, що сьогодні існує ***протиріччя***, яке полягає у відсутності методів керування ресурсами та потоками даних, які б дали змогу врахувати процеси комунікації між елементами розподілених сервісів на прикладному рівні розподілених гетерогенних інформаційно-телекомунікаційних систем зі збереженням обсягів їх апаратного забезпечення.

Отже, розроблення методів та моделей надання послуг у гетерогенних розподілених інформаційно-телекомунікаційних системах для підвищення якості обслуговування та покращення структурно-функціональних параметрів і характеристик цих систем з метою узгодження взаємодії сервісного шару та шару передавання даних є актуальною невирішеною на сьогодні ***науковою проблемою.***

Зв'язок роботи з науковими програмами, планами, темами. Тематика дисертаційної роботи безпосередньо пов'язана з положеннями Постанови Верховної Ради України про «Концепцію національної інформаційної політики», а також «Концепції конвергенції телефонних мереж і мереж з пакетною комутацією в Україні», «Стратегії розвитку інформаційного суспільства в Україні», Закону України «Про Основні засади розвитку інформаційного суспільства в Україні на 2007-2015 роки». Дисертаційні дослідження виконувались у відповідності до наукового напряму кафедри телекомунікацій Національного університету «Львівська політехніка» - «Інфокомунікаційні системи та мережі», в рамках низки держбюджетних науково-дослідних робіт: «Дослідження та розроблення телекомунікаційних мережних систем для застосувань телематики і телеметрії» (ДБ/КОМ) (2011-2012 рр.), № держреєстрації 0111U001223, «Моделі та структури конвергентних телекомунікаційних мереж на основі CLOUD – технологій» («ДБ/CLOUD»)

(2013-2014 рр.), № держреєстрації 0113U003184, «Методи побудови та моделі інформаційно-телекомунікаційної інфраструктури на основі SDN-технологій для систем електронного урядування» (ДБ/SDN) (2015-2016), № держреєстрації 0115U000444, «Методи побудови гетерогенних інформаційно-комунікаційних систем для розгортання програмно-конфігурованих мереж 5G подвійного використання» (ДБ/5G) (2017-2018**),** а також госпдоговірної тематики ГД № 0489, 2014 р.

Мета і завдання дослідження. Метою дисертації є покращення значень метрик якості надання послуг шляхом оптимізації структури та показників ефективності функціонування телекомунікаційної підсистеми, а також підвищення захищеності, відмовостійкості та зниження тривалості відновлення працездатності інформаційної підсистеми в розподілених інформаційно-телекомунікаційних системах.

Для досягнення поставленої мети в межах дисертаційних досліджень були сформульовані та розв’язані такі завдання:

1. Аналіз методів побудови та процесів комунікації в сучасних інформаційно-телекомунікаційних системах.
2. Дослідження, систематизація та розроблення методів побудови та моделей гетерогенних розподілених інформаційно-телекомунікаційних систем.
3. Розроблення методів структурно-функціонального синтезу телекомунікаційної підсистеми інформаційно-телекомунікаційних систем.
4. Розроблення методів і дослідження процесів керування конфігурацією та забезпечення катастрофостійкості інформаційної підсистеми інформаційно-телекомунікаційних систем.
5. Розроблення моделей та дослідження процесів надання послуг в гетерогенних інформаційно-комунікаційних системах.
6. Експериментальне дослідження поведінки складної інформаційно-комунікаційної системи за умов зростання попиту на інформаційно-комунікаційні послуги.

Об'єкт дослідження – процес надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах.

Предмет дослідження – методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах.

Методи дослідження. У процесі наукових досліджень використано методи теорії масового обслуговування, теорії інформації, математичної статистики, методи оптимізації, теорії ймовірностей, математичного та імітаційного моделювання, теорію графів. Для підтвердження теоретичних результатів застосовано експериментальні методи дослідження.

Наукова новизна отриманих результатів.

Вперше запропоновано:

1. Метод передавання інформації у процесі надання інформаційно-комунікаційних послуг з урахуванням функцій корисності та параметрів процесу пересилання запиту на інформаційно-комунікаційну послугу, чим досягнуто міжрівневу максимізацію ефективності використання ресурсів інформаційно-телекомунікаційної системи;

2. Метод структурно-функціонального синтезу інформаційно-комунікаційних систем, який враховує процеси керування інформаційними потоками на рівні телекомунікаційної мережі, а також процес різнокласового передавання інформації з класифікацією інформаційно-телекомунікаційних послуг на основі рівнів підписки користувачів та забезпечує можливість оцінювання якості надання послуг у процесі їх доставки кінцевим користувачам;

3. Метод відновлення працездатності інформаційно-телекомунікаційної системи після відмов антропогенного походження на основі представлення інфраструктури у вигляді програмного коду, що дає змогу зменшити тривалість відновлення працездатності у процесі розгортання інфраструктури на базі cloud-сервісів;

4. Модель корпоративного клієнта та модель спільної cloud-інфраструктури системи надання інформаційно-комунікаційних послуг, що відрізняються від відомих програмним узагальненим поданням властивостей

об’єктів і які, у сукупності, дають змогу оцінити зміни значень метрик якості надання послуг у процесі навантажувального тестування за умов зростання попиту на інформаційно-комунікаційні послуги.

Набули подальшого розвитку:

5. Територіально-залежна модель надання інформаційно-телекомунікаційних послуг, яка, на відміну від відомих, враховує характер та динаміку переміщення користувачів, їх інформаційну активність та дає змогу збалансувати використання ресурсів мережі, мінімізувавши кількість вузьких місць, та підвищити доступність інформаційно-комунікаційних ресурсів;

6. Метод узгодженого проектування фізичної та логічної структур телекомунікаційної мережі, який відрізняється від відомих багатофакторним аналізом альтернативних структур, що дає змогу досягти оптимального планування фізичних ресурсів на основі виконання ітеративного пошуку.

Практичне значення одержаних результатів. Основним практичним результатом дисертації є розроблена методика дослідження розподілених гетерогенних інформаційно-телекомунікаційних систем на базі cloud-інфраструктури, що є важливим інструментом у процесі їх проектування та експлуатації та дає змогу у цілому розв’язати проблему гармонізації взаємодії рівнів запропонованої концептуальної структури інформаційно-телекомунікаційних систем із покращенням значень метрик якості обслуговування та оптимальним розподілом ресурсів таких систем. За неможливості використання технологій моделювання поведінки досліджуваних систем, розроблена комплексна методика об'єктно-орієнтованого навантажувального тестування дає змогу отримати основні показники використання інфраструктури та якості сприйняття послуг, які надаються користувачам, за умов впливу зростання абонентського навантаження.

На основі наукових результатів дисертації отримано такі практичні здобутки:

1. Із застосуванням розроблених моделі пересилання запиту та територіально-залежної моделі, а також методу балансування навантаження

вдалося знизити кількість втрачених комунікаційних сеансів на 14%, у порівнянні зі звичайним режимом роботи системи.

2. На основі нових моделей корпоративного клієнта інформаційно-комунікаційної системи надання послуг та моделі спільної cloud-інфраструктури у поєднанні з реалізованою логікою роботи досліджуваної системи розроблено методику їх навантажувального тестування, що дає змогу оцінити ефективність використання cloud-інфраструктури та показники якості сприйняття послуг користувачами системи під впливом процесів зростання навантаження та безперервної системної інтеграції.

3. Запропоновано алгоритм автоматичного відновлення cloud-інфраструктури після явищ катастрофічного характеру на основі концепції IaaC (Infrastructure as a Code). Отримані результати підтвердили переваги підходу IaaC для відновлення таких систем після аварій та дали змогу на порядок знизити тривалість післяаварійного відновлення.

4. Запропоновано методику перевірки адекватності моделей програмних компонентів гетерогенних розподілених інформаційно-телекомунікаційних систем, яка дає змогу шляхом аналізу процесу передавання даних встановити відхилення затримки передавання між апаратним вузлом та його програмним аналогом за умови однакового алгоритму їх функціонування. Максимальне відхилення затримки пакетів між досліджуваним програмним та апаратним компонентами знаходиться в межах 0,5 %.

5. Доведено, що балансування навантаження за допомогою інтегрованої системи керування розгортанням інформаційних сервісів дає змогу зменшити тривалість обслуговування запитів у середньому до 3 разів. Показано системний вплив механізмів балансування навантаження, який полягає у обмеженні мінімального значення затримки обслуговування запитів на послуги. Проведено порівняльне оцінювання затримки, результати якого дають змогу стверджувати, що зростання пропускної здатності каналу у 10 разів приводить до зниження затримки обслуговування запиту у середньому в 2,5 рази, що пояснюється обмеженнями методу балансування навантаження.

Основні результати дисертаційної роботи використано і впроваджено:

- у ТзОВ «Інформконсалт» для покращення якості обслуговування абонентів та зниження тривалості відновлення інформаційно-телекомунікаційної інфраструктури;
- у ТзОВ «Телекомунікаційна компанія» для адаптації якості обслуговування абонентів у інформаційно-телекомунікаційній системі з програмними компонентами;
- у ПП «Цифрові технології» для покращення часових показників функціонування і підвищення захищеності інформаційних сервісів, які підлягають розгортанню у cloud-інфраструктурі;
- у навчальному процесі кафедри телекомунікацій Національного університету «Львівська політехніка» для модернізації курсу лекцій з дисципліни «Розподілені сервісні системи та cloud-технології».

Особистий внесок здобувача. Основні наукові результати дисертаційної роботи отримано автором самостійно. У працях, опублікованих у співавторстві, авторові належать[1]: у роботах [1, 5, 6, 7, 13, 15, 16, 17, 40] –теоретичні основи міжрівневої взаємодії у гетерогенних розподілених інформаційно-комунікаційних системах; у роботах [2, 38, 48, 50] – теоретичні основи захисту мереж від впливу мережних атак типу DDoS; у роботах [4, 8, 9, 10, 11, 12, 14, 24, 25, 30, 31, 35, 39, 43, 46, 53, 56] – теоретичні основи та дослідження функціональних аспектів процесу надання послуг у безпровідних інформаційно-телекомунікаційних системах, які лягли в основі територіально-залежної моделі обслуговування; у роботах [3, 18, 19, 20, 21, 22, 23, 26, 27, 36, 41, 49] – теоретичні основи дослідження властивостей інформаційних потоків у розподілених гетерогенних інформаційно-телекомунікаційних системах та їх впливу на значення метрик якості обслуговування, формулювання оптимізаційних завдань; у роботах [28, 29, 51, 52, 55, 57] – теоретичні аспекти структурно-функціонального синтезу телекомунікаційної складової інформаційно-

[1] Посилання наведено на перелік публікацій автора у Додатку А.

телекомунікаційних систем; у роботах [34, 37, 54] – теоретичні особливості структурно-функціонального синтезу інформаційної складової інформаційно-телекомунікаційних систем; у роботах [32, 47] – теоретичне обґрунтування процесу пересилання інформаційних потоків у оптичних транспортних системах з адаптацією до типу сервісного навантаження; у роботах [33, 44, 45] – обґрунтування та дослідження надійнісних аспектів функціонування розподілених гетерогенних інформаційно-телекомунікаційних систем; у роботі [42] – приклад імплементації розподіленої гетерогенної інформаційно-телекомунікаційної системи.

Результати спільних наукових праць було використано у дисертаційних роботах таких співавторів, як Стрихалюк Б.М. [290] (для дослідження доступності сервісів у багатоланкових cloud системах), Шпур О.М. [283] (використано теоретичні розробки автора для дослідження параметрів якості надання композитних сервісів у віртуалізованих ЦОД), Демидов І.В. [274] (для дослідження систем блокування інформаційних загроз у системах надання послуг національного масштабу), Бешлей М.І. [285], який використав розроблені автором моделі трафіку інформаційно-телекомунікаційних систем для розроблення програмних компонентів таких систем, Бугиль Б.А. [271], який використав розроблені автором засади міжрівневої взаємодії для дослідження методів та моделей узгодженого проектування фізичної та логічної структур телекомунікаційних мереж.

Апробація результатів дисертації. Основні наукові результати і положення дисертації представлені, доповідались та обговорені на 22-х міжнародних і всеукраїнських науково-технічних конференціях та наукових семінарах: Міжнародних науково-технічних конференціях «Сучасні проблеми радіоелектроніки, телекомунікацій, комп’ютерної інженерії» (м. Львів-Славське, 2012, 2014, 2016, 2018 рр.); Міжнародних науково-технічних конференціях «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці». (м. Львів-Поляна, 2013, 2015, 2017 рр.); Міжнародній науково-технічній конференції «Проблеми телекомунікацій - 2016» ПТ-16

(м. Київ, 2016 рр.); International Scientific-Practical Conference Problems of Infocommunications Science and Technology (м. Харків, 2014, 2015, 2016); International Conference on Advanced Information and Communication Technologies (м. Львів, 2015, 2017); 2017 IEEE 1st Ukraine Conference on Electrical and Computer Engineering, UKRCON 2017 (м. Київ, 2017 р.); Науково-практичних конференціях «Сучасні проблеми телекомунікацій і підготовка фахівців в галузі телекомунікацій – 2012, 2013, 2014» (м. Львів, 2012, 2013, 2014 рр.); XIII Міжнародній науково-технічній конференції «Вимірювальна та обчислювальна техніка в технологічних процесах» (м. Одеса, 2014); Microwave and Telecommunication Technology (CriMiCo), 2013 23rd International Crimean Conference (м. Севастополь, Україна, 2013 р.); Міжнародній конференції з інформаційно-телекомунікаційних технологій та радіоелектроніки IEEE (УкрМіКо'2016/UkrMiCo'2016) (м. Київ, 2016 р.); VI Міжнародному науково-технічному симпозіумі «Нові технології в телекомунікаціях» (Вишків, 2013 р.); Сучасні інформаційно-комунікаційні технології COMINFO'2012 (Ялта-Лівадія, Україна, 2012 р.); "Фізико-технологічні проблеми передавання, оброблення та зберігання інформації в інфокомунікаційних системах" (Чернівці, 2016 р.). Крім цього, дисертаційна робота у повному обсязі представлена на наукових семінарах кафедри телекомунікацій Національного університету «Львівська політехніка».

Публікації. За результатами досліджень, які викладені у дисертаційній роботі, опубліковано 57 наукових праць, серед них статей у іноземних періодичних виданнях – 6, статей у наукових фахових виданнях України – 17 (з них 13 статей – у науковій періодиці, що входить до міжнародних наукометричних баз різного рівня), статей у наукових виданнях України – 2, у збірниках матеріалів і тез доповідей міжнародних та всеукраїнських конференцій – 32, з них індексованих у наукометричній базі Scopus – 16.

Структура та обсяг роботи. Робота складається з переліку умовних скорочень, вступу, шести розділів, висновків, списку використаних джерел і двох додатків. Загальний обсяг роботи складає 402 сторінки друкарського тексту, з

яких 308 сторінок основного тексту, 142 рисунки, 13 таблиць, список використаних джерел із 293 найменувань, 2 додатки на 13 сторінках. Додатки містять акти впровадження результатів дисертаційної роботи, а також список наукових праць автора.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ТЕХНОЛОГІЙ НАДАННЯ ПОСЛУГ В ГЕТЕРОГЕННИХ РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНО-ТЕЛЕКОКОМУНІКАЦІЙНИХ СИСТЕМАХ

1.1. Архітектури інформаційно-комунікаційних систем

Архітектура програмного забезпечення для програмної або обчислювальної системи – це структура системи, що складається з елементів програмного забезпечення, видимих зовнішніх властивостей цих елементів та взаємозв'язків між ними. Архітектура програмного забезпечення – це отримання структурованого рішення, яке відповідає всім технічним та операційним вимогам шляхом оптимізації таких спільних атрибутів як продуктивність, безпека та керованість. Вона включає низку рішень, що базуються на різноманітних факторах, і кожне з цих рішень може мати значний вплив на якість, продуктивність та загальну ефективність системи [182-184].

Інші визначення архітектури.

- Архітектура - це проект на найвищому рівні узагальнення.
- Архітектура – це загальна структура системи.
- Архітектура – це компоненти та взаємозв'язки.

1.1.1. Аналіз цілей розроблення архітектур

Архітектура програмного забезпечення охоплює сукупність важливих рішень щодо організації програмної системи, включаючи:

- вибір структурних елементів та інтерфейсів, за допомогою яких утворюється система [203-204];
- поведінку, яка є результатом взаємодії цих елементів [287];
- композицію структурних та протокольних елементів у великі підсистеми;
- архітектурний стиль, який керує визначає сам процес організації.

Архітектура ІТС також включає функціональність, зручність використання, стійкість, продуктивність, повторне використання, зрозумілість, економічні та технологічні обмеження, можливі компроміси тощо [197-200].

Будь-яке архітектурне рішення тягне за собою:

- Введення обмежень щодо імплементації;
- Визначення організаційної структури;
- Введення атрибутів якості [146];
- Можливість прогнозування якості функціонування;
- Більш точні оцінки витрат та планування робочого процесу.

Як і будь-яка інша складна структура, ІТС повинна бути побудована на міцному фундаменті. Для того, щоб мінімізувати ризики розроблення неуспішної ІТС, слід враховувати основні сценарії використання, загальні проблеми та оцінювати довгострокові наслідки ключових рішень. Сучасні інструменти та платформи допомагають спростити завдання створення ІТС, але вони не замінюють необхідність ретельно розробляти систему, виходячи з конкретних сценаріїв та вимог. Ризики, що виникають внаслідок поганої архітектури, включають нестабільну ІТС, яку складно підтримувати та доповнювати новими функіями, розгортати та керувати у реальному середовищі.

1.1.2. Аналіз впливу системної архітектури на значення метрик якості інформаційно-комунікаційної системи

Системи повинні бути розроблені з урахуванням користувачів, бізнес цілей та ІТ-інфраструктури. Для кожної з цих областей слід окреслити основні сценарії та визначати важливі атрибути якості (наприклад, надійність або масштабованість) та основні критерії досягнення якісного результату. У процесі розроблення ІТС необхідно враховувати показники, які відображають прогрес у кожній області [133].

Дуже часто необхідно йти на компроміси і знаходити баланс між суперечливими вимогами в цих трьох областях [278, 281]. Наприклад, загальний досвід користувачів є результатом прийнятих рішень в областях бізнесу та ІТ-інфраструктури, а зміни в одній або іншій можуть суттєво вплинути на кінцевий досвід користувача (Рис. 1.1).

Рис. 1.1. Фактори, що підлягають урахуванню в процесі розробки архітектур

Аналогічним чином, зміни у вимогах щодо роботи користувачів з програмами можуть мати значний вплив на вимоги щодо бізнесу та IT-інфраструктури. Продуктивність може бути основним пріоритетом для користувача та бізнесу, проте системний адміністратор може не мати змоги вкладати кошти в обладнання, необхідне для досягнення цієї мети. Компромісом може бути забезпечення максимальної продуктивності протягом 80 відсотків всього часу функціонування ІТС [137, 141].

Основним фокусом архітектора системи є визначення основних компонентів в ІТС та їх взаємодії. Вибір структур даних та алгоритмів або деталі реалізації окремих компонентів є проблемами програмної інженерії. В той же час вибір та визначення способу комунікації є завданнями телекомунікаційної інженерії. Розроблення архітектури чи інженерного рішення дуже часто накладаються [179].

Приклади атрибутів якості: зручність в користуванні, легкість щодо внесення змін, продуктивність. Архітектура є критичною для багатьох параметрів якості. Архітектура сама по собі не може забезпечити певні значення метрики якості [220, 248-250, 269].

Архітектура програмного забезпечення спрямована на узгодження бізнес-вимог та технічних вимог шляхом аналізу сценаріїв використання та пошуку шляхів реалізації цих сценаріїв. Основним завдання під час розробки архітектури є визначення потреб, які впливають на структуру системи. Хороша архітектура

зменшує бізнес ризики, пов'язані з побудовою технічного рішення [258-261]. Хороший проект є достатньо гнучким, щоб мати змогу впоратися із змінами, які неодмінно будуть виникати у апаратних і програмних технологіях, а також у сценаріях використання та вимогах користувачів [142, 188-190]. Архітектор повинен розглянути загальний ефект від архітектурних рішень, компроміси між атрибутами якості (наприклад, продуктивність та безпека), а також компроміси, необхідні для вирішення користувацьких, системних та бізнес-вимог.

Архітектура повинна [251-255, 257]:

- Розкривати структуру системи, але приховувати деталі реалізації.
- Реалізувати всі можливі випадки використання та сценарії.
- Намагатися відповідати вимогам усіх зацікавлених сторін.
- Підтримувати як функціональні, так і якісні вимоги.

1.2. Порівняльний аналіз архітектурних шаблонів інформаційно-комунікаційних систем

1.2.1. Архітектури інформаційної підсистеми та їх вплив на процес надання послуг

Програмне забезпечення ділиться на рівні, які можуть розроблятися одночасно. Всі рівні можуть використовувати загальнодоступні інтерфейси інших рівнів.

Шаблон брокера вирішує проблему наявності багатьох сервісів між декількома серверами. Він встановлює брокер між клієнтом і серверами. Брокер перенаправляє клієнта на правильний сервер та відповідь від серверів до клієнта. Для того, щоб відділити користувацький інтерфейс від функціональності, практикується розділяти логіку програмного забезпечення на три основні компоненти [131] (Рис. 1.2): модель даних, графічний формат та спосіб представлення даних, контролер, який диначімно керує форматом відображення даних та наповненням графічного інтерфейсу потрібними даними з моделі [157].

Таблиця 1.1. Функції елементів архітектурного шаблону «модель-представлення-контролер»

Рівень	Характеристика
Модель	Об'єкт для збереження даних. Він зазвичай являє собою якийсь об'єкт, наприклад користувач. Модель повинна мати методи для читання та запису атрибутів, а також мати можливість запускати події зміни.
Представлення	Представлення відображається користувачеві та відповідає за візуалізацію даних.
Контролер	Оперує даними та зміною їх представлення.

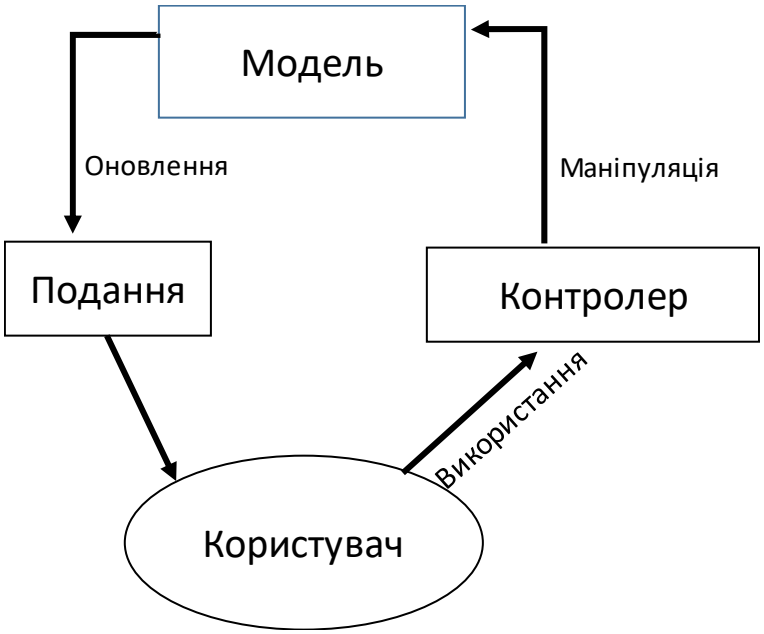


Рис. 1.2. Принцип функціонування архітектурного шаблону «модель-представлення-контролер»

Архітектурний шаблон компонентів і зв’язків (C & C) описує процес виконання, зберігання даних, клієнтів, серверів та їх взаємодії. Він документує потоки керування та потоки даних компонентів програми. У вигляді компонентів відображаються клієнти, сервери, фільтри, об'єкти та бази даних. Інтерфейси між компонентами називаються "портами". Всі порти мають бути явно задокументовані, особливо такі зв'язки як процедурні виклики, виклики до баз даних, обмін повідомленнями та асинхронні потоки даних. Значним недоліком багатьох варіантів архітектури C&C є відсутність іменування зв'язків. Всі елементи в архітектурі повинні бути чітко визначені.

Оскільки потоки даних та топології керування не завжди є синхронними, вони повинні бути представлені незалежно на діаграмі. Для складних систем для представлення двох потоків слід використовувати різні діаграми. Відображення між елементами C&C та елементами модуля спрощує розуміння відмінностей між компонентами та зв'язками.

Набір розподілених обчислювальних об'єктів, які володіють рівними правами і функціями називється Peer-to-peer системою. Взаємодія в таких системах зазвичай відбувається за принципом запит - відповідь.

Спільні ресурси та сервери можуть використовуватися великою кількістю клієнтів. Деякі компоненти можуть бути як клієнтами, так і серверами. Програмне забезпечення може перетворювати потоки даних кілька разів. В результаті утворюється потік запитів, які проходять через фільтри, що змінюють дані цих запитів у правильному порядку. Компоненти узгоджують взаємодію завдяки оголошенню повідомлень або подій. Компоненти можуть підписатися на сукупність подій. Будь-який компонент може як публікувати, так і підписуватися. Компонент, що публікує, повинен надати зручні доступні послуги для своїх користувачів. Споживачеві не потрібно нічого знати про реалізацію. Компоненти мають інтерфейси, які описують надані ними послуги. Часто ці інтерфейси описані на мові, що може транслюватися в будь яку іншу мову.

Хмарні обчислення – це тип обчислень на базі Інтернету, який забезпечує надання в оренду обчислювальних ресурсів на вимогу. Це модель для забезпечення широкомасштабного доступу до спільного пулу конфігурованих обчислювальних ресурсів (наприклад, комп'ютерних мереж, серверів, сховищ, програм та служб), які можуть бути швидко надані без суттєвих затрат зі сторони керування цими ресурсами. Рішення у сфері хмарних обчислень і зберігання надають користувачам та підприємствам різні можливості зберігати та обробляти свої дані в приватних або сторонніх центрах обробки даних, які можуть бути розташовані далеко від користувачів.

1.2.2. Пірингові системні архітектури та оверлейні мережі

Однорангова мережа (P2P) - це розподілена архітектура застосунків, яка розділяє завдання або навантаження між своїми вузлами. Вузли мають однакові привілеї та є рівноправними учасниками системи.

Вузли відкривають частину своїх ресурсів, таких як обчислювальну потужність, пам'ять або пропускну здатність мережі, безпосередньо для користування іншим учасникам мережі без необхідності центральної координації з боку серверів. Кожен вузол є водночас постачальником та споживачем ресурсів, на відміну від традиційної моделі клієнт-сервер. Новітні системи P2P перевершують попередників завдяки можливості інтеграції вузлів на різних платформах, що дає змогу виконувати нові, набагато складніші типи завдань [191].

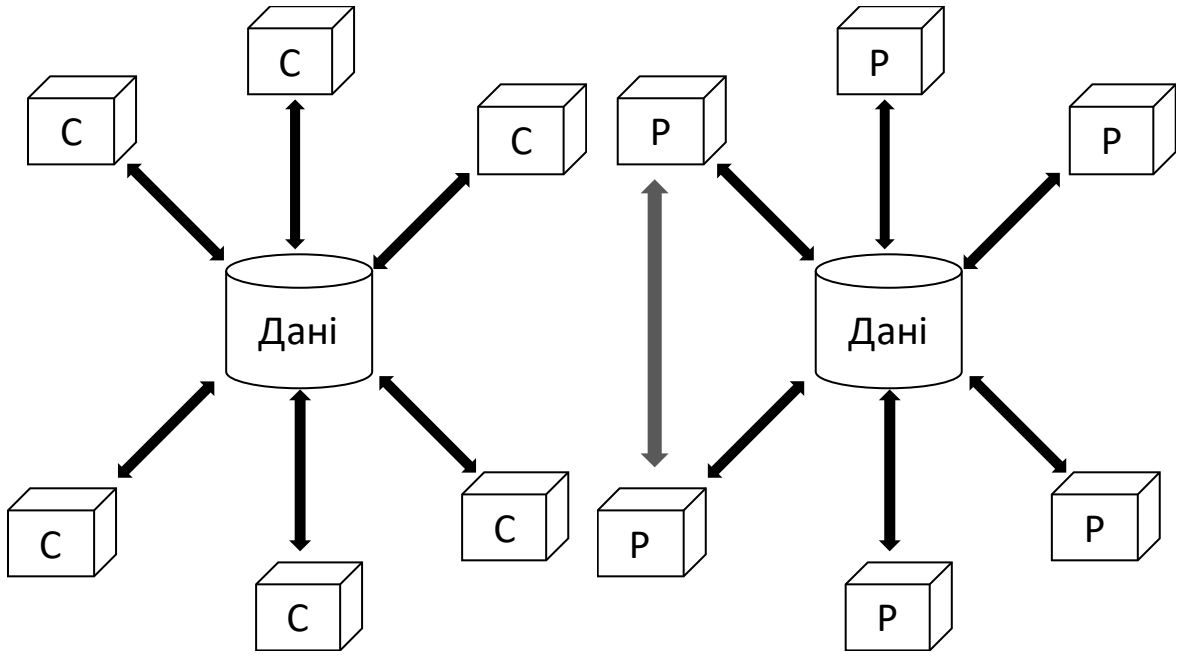
У неструктурованій P2P немає зв'язку між контентом вузла та його адресою. Неструктуровані однорангові мережі не накладають певної структури на мережу, а формуються вузлами, які випадково утворюють зв'язки один з одним. (Gnutella, Gossip і Kazaa є прикладами неструктурованих протоколів P2P).Оскільки немає глобально визначеної структури, неструктуровані мережі можуть легко проводити локальні оптимізації в різних частинах мережі. Також, оскільки роль всіх вузлів у мережі однакова, неструктуровані мережі дуже надійні у ситуаціях, коли велика кількість вузлів часто приєднуються до мережі та виходять з неї.

Для структурованої P2P характерне таке:

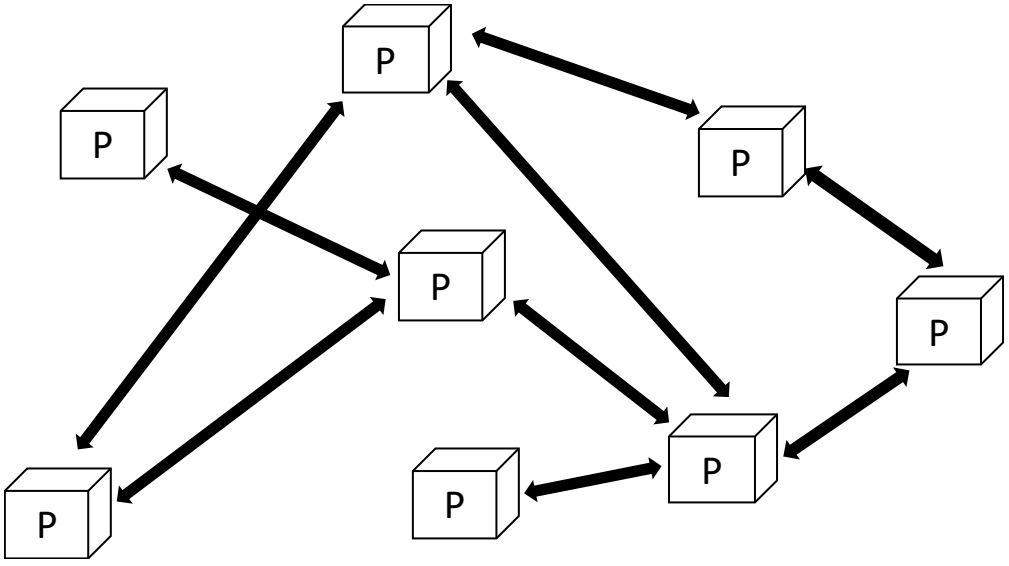
- Зв'язок між вузлом та контентом
- Зазвичай функціонує на основі розподіленої хеш-таблиці (DHT).

Структуровані однорангові мережі організовуються у певну топологію, а протокол забезпечує можливість будь-якому вузлу ефективно здійснювати пошук файлу чи ресурсу, навіть якщо цей ресурс трапляється нечасто.

Класифікація піринговых систем представлена на Рис. 1.3.



а б



в

Рис. 1.3. Класифікація піринговых систем

Виділяють такі пірингові системи:

– Неструктурована
– Централізована
– Чиста
– Гібридна

Такі однорангові мережі (наприклад, Napster і Publius) мають централізовану топологію та відображають декілька характеристик клієнта та

сервера. Цей тип однорангових мереж містить центральний сервер, який функціонує як сервер каталогів. Але цей сервер каталогів має менше завдань, ніж сервери в мережах клієнт - сервер. Коли вони приєднуються до системи, то оголошують про свою присутність і надають певну інформацію на сервер каталогу. Таким чином, існує один сервер, який містить індекс усіх доступних ресурсів у мережі. Недоліки системи в основному пов'язані з можливим вузьким місцем на сервері.

Найбільш яскрава особливість чистої децентралізованої неструктурованої системи, наприклад такої як Gnutella 0.4 – це відсутність централізованого компонента, який означає, що всі вузли безпосередньо пов'язані один з одним. Вузли функціонують як клієнти, сервери, маршрутизатори та кеш-пам'ять. Перевага цієї категорії однорангових архітектур полягає в тому, що немає єдиної точки відмови і що вона є відмовостійкою. Помилка одного або декількох вузлів мало впливає на продуктивність мережі. Основним завданням є розробка ефективного методу пошуку, здатного забезпечити задовільні результати пошуку при наявності перехідних вузлів. Наприклад, брак контенту у великій мережі обміну файлами може бути проблемним.

Гібридні моделі – це комбінація однорангових та клієнт-сервер моделей. Характеристикою гібридних моделей є наявність центрального сервера, який допомагає вузлам знаходити один одного. Spotify – це приклад гібридної моделі. Є цілий ряд гібридних моделей, кожна з яких реалізує компроміси між централізованою функціональністю, наданою структурованою мережею, і рівноправністю вузлів, наданою чистою одноранговою структурою безпровідних мереж. В даний час гібридні моделі мають більш високу продуктивність, ніж чисті неструктуровані мережі чи чисті структуровані мережі, оскільки певні функції, такі як пошук, вимагають централізованої функціональності, але мають перевагу від децентралізованої агрегації вузлів, наданих неструктурованими мережами.

Найпоширеніший тип структурованих мереж P2P реалізує розподілену хеш-таблицю (DHT), в якій варіант послідовного хешування використовується для

присвоєння права власності кожного файлу певному вузлі. Це дозволяє вузлам шукати ресурси в мережі, використовуючи хеш-таблицю: тобто пари (ключ, значення), які зберігаються в DHT, і будь-який вузол-учасник може ефективно отримати значення, пов'язане з заданим ключем.

Проте, для ефективної маршрутизації трафіку через мережу, вузли в структурованій мережі повинні підтримувати списки сусідів, які задовольняють певним критеріям. Це робить їх менш надійними в мережах із високою швидкістю оновлення стану (тобто з великою кількістю вузлів, які часто приєднуються та залишають мережу). Останні дослідженн таких мереж в умовах реальних робочих навантажень виявили проблемні місця рішень на базі DHT, зокрема висока вартість надання/пошуку ресурсів та статичний і динамічний дисбаланс завантаження.

До загально відомих прикладів, що використовують DHT, належить розподілений трекер BitTorrent, мережа Kad, Storm botnet, YaCy та мережа надання контенту Coral. Деякі наукові проекти включають в себе проект Chord, Kademlia, утиліту зберігання PAST, P-Grid та систему розподілу контенту CoopNet. DHT-мережі також широко використовуються для ефективного пошуку ресурсів для мережних обчислювальних систем, оскільки це допомагає керувати ресурсами та планувати використання застосунків.

Для централізованих піринговых систем (Рис. 1.4) характерні такі властивості:

- Основою є центральний сервер пошуку
- Топологія зірка
- Кожен вузол з'єднаний з пошуковим сервером
- Запити обробляються пошуковим сервером, повертаючи точку доступу до сервісу
- Сам контент передається прямими найкоротшими шляхами

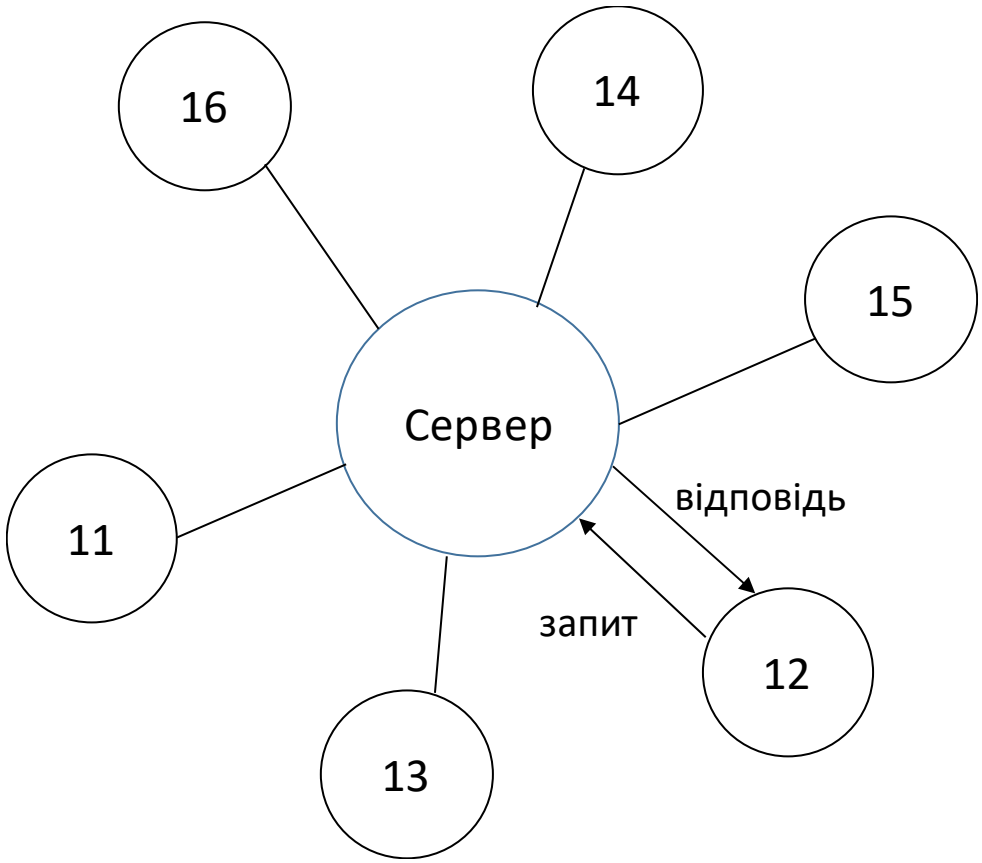


Рис. 1.4. Структура централізованої пірингової системи

На початковому етапі мережі P2P були визначені як повністю децентралізовані, однак з практичних причин були розроблені системи з різним ступенем централізації. Тому, крім суто децентралізованих архітектур, були запропоновані також гібридні системи.

У суто розподіленій P2P системі всі вузли в мережі виконують точно ті ж завдання, що і клієнти та сервери, проте відсутня централізована координація їх діяльності. Основні недоліки суто розподілених систем повязані з масштабованістю та низькою продуктивністю під час обробки запитів.

У гібридних централізованих системах індексації існує центральний сервер, що полегшує взаємодію між вузлами. Централізований індекс створюється на спеціалізованому вузлі. Наприклад, Napster використовує централізований індекс, який побудований у взаємодії з усіма іншимим вузлами. Централізований індекс зберігає інформацію про дані, що зберігаються в кожному вузлі, разом із одночасним ідентифікатором. Тому централізовані показники ефективні під час обробки запитів; для визначення того, який вузол зберігає відповідну

інформацію, потрібно одне повідомлення. Фактичний обмін інформацією між вузлами здійснюється без взаємодії з центральним сервером. Незважаючи на ефективність під час обробки запитів, централізовані індекси мають суттєвий недолік, що полягає в існуванні єдиної точки відмови. Крім того, централізований індекс може стати вузьким місцем для системи, особливо у випадку великої мережі.

Гібридні децентралізовані системи індексування використовують переваги як централізованої, так і суто розподіленої архітектури. Мережі Superpeer вирішують проблеми масштабування та "єдиної точки відмови" централізованих підходів, використовуючи переваги повністю розподіленого підходу, де кожен збирає і підтримує індекс над власними файлами. Ці системи подібні до повністю децентралізованих систем, але деякі з вузлів відіграють більш важливу роль і несуть відповідальність за збереження інформації, доступної своїм сусідам. Зазвичай такі системи розглядаються як суто децентралізовані.

Недоліки централізованих P2P:

- Єдина точка відмови
- Вузьке місце, масштабованість
- Центральний сервер керує усіма вузлами

Переваги централізованих P2P:

- Простота
- Швидкість і повнота пошуку
- Можливість нечіткого запиту

Для чистих P2P характерно таке:

- Будь-який вузол може бути видалений без втрати функціональності
- Немає центральних вузлів
- З'єднання встановлюються випадковим чином

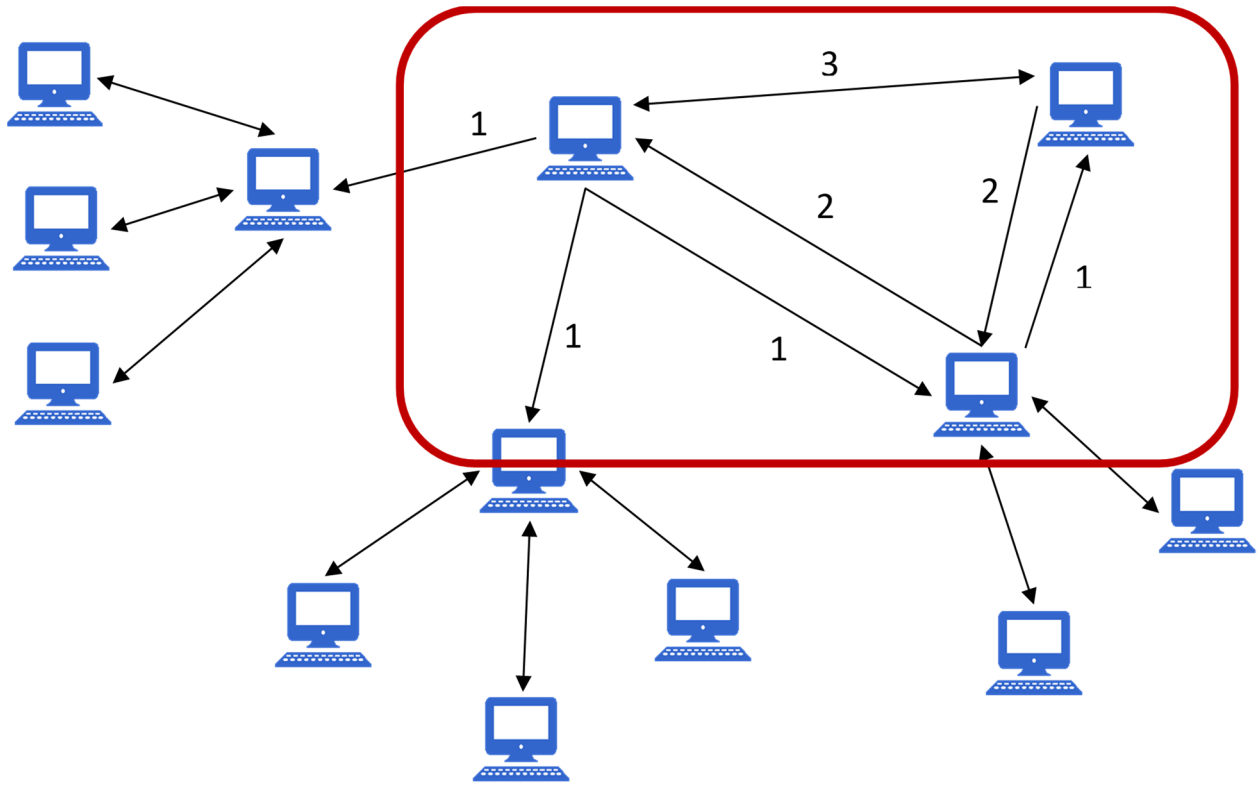


Рис. 1.5. Принцип функціонування системи Gnutella

Gnutella – суто децентралізована система P2P (Рис. 1.5). Ні один вузол не відповідає за організацію мережі, і між клієнтом та сервером не існує жодної дискримінації. Вузли в системі підключаються один до одного безпосередньо через певного програмного посередника. Мережа Gnutella розширюється, коли нові вузли приєднуються до мережі та згортається, коли вузли залишають мережу. У цьому сенсі – це мережева інфраструктура на основі програмного забезпечення. Маршрутизатори, комутатори та концентратори не потрібні для зв'язку на цьому рівні.

Основні операції Gnutella включають приєднання або вихід з мережі, пошук та завантаження файлів:

– Коли вузол приєднується до мережі Gnutella, він надсилає повідомлення "PING", щоб вказати свою присутність. Повідомлення "PING" передається на інші вузли за допомогою трансляції. Коли вузли отримують повідомлення "PING", вони відсилають повідомлення "PONG" у відповідь, а це означає, що вони тепер знають про існування нового вузла. З повідомлень "PONG" новачок може отримати інформацію про ці вузли

та встановити зв’язок з окремими сусідами. Коли вузол залишає мережу, вузол не повинен повідомляти своїх сусідів. Навпаки, кожен вузол повинен регулярно перевіряти своїх сусідів повідомленнями "PING", щоб перевірити, чи його сусіди все ще онлайн. Якщо відповідь не повертається, вузол вважатиме, що сусід залишив мережу і онобновить список своїх сусідів.

– Коли вузол хоче знайти певні файли, він запитує своїх сусідів, відправивши запит на пошук, а його сусіди, у свою чергу, передають повідомлення всім своїм сусідам таким же чином. Ті, у кого є потрібні файли, відповідають ініціатору запиту "повідомленнями", які зворотньо спрямовуються назад тим же ж шляхом, що і маршрут запиту. Процес трансляції триває, доки не буде покрито всю мережу, або значення TTL (час очікування) пошукового повідомлення досягне нуля. Тепер оригінальний вузол може мати достатню кількість відповідей, з яких може вибрати кілька вузлів для підключення, а потім завантажити потрібні файли. Крім того, кожне повідомлення додається до унікального ідентифікатора. Коли вузол отримує повідомлення з ідентифікатором, який він бачив, він видалить повідомлення таким чином, щоб можна було уникнути циклів повідомлення. Рис. 1.5 ілюструє процес маршрутизації у системі Gnutella.

Недоліки чистих P2P:

– Великі обсяги службового трафіку
– Сформована топологія часто не оптимальна
– Складні маршрути

Переваги чистих P2P:

– Відсутність єдиної точки відмови
– Анонімність

Завантаження відбувається таким чином:

– Необхідно знайти принаймні один активний вузол

– Кешування завантаження (спроба підключитися до вузлів з попередніх сеансів)
– Завантажувальний сервер (загальновідомий сервер який є центральним елементом архіьтектури)
– Трансляція (широкомовна розсилка IP (обмежена локальною мережею))

Маршрутизація:

– Повністю децентралізована
– Реактивна – маршрути до контент-провайдерів встановлюються лише за запитом
– Запити: широкомовна розсилка
– Відповіді: відправлені у зворотньому няпрямку
– Пошук контенту

1.2.3. Архітектури типу «клієнт – сервер»

Архітектурний шаблон клієнт - сервер описує розподілені системи, що включають окрему клієнтську та серверну системи, а також комунікаційну мережу. Найпростіша форма системи клієнт-сервер включає серверну програму, доступ до якої безпосередньо здійснюється кількома клієнтами.

У більш загальному плані цей архітектурний шаблон описує взаємозв'язок між клієнтом і одним або кількома серверами, де клієнт ініціює один або декілька запитів (можливо, використовуючи графічний інтерфейс), очікує на отримання відповідей і обробляє відповіді. Сервер зазвичай авторизує користувача, а потім виконує обробку, необхідну для генерації результату. Сервер може надсилати відповіді за допомогою ряду протоколів та форматів даних для передачі інформації клієнту.

Сьогодні деякі приклади архітектури клієнт – сервер включають:

– веб застосунки, що працюють у мережі Інтернет;
– операційні системи Microsoft Windows®, що працюють на мережевих сервісах передавання даних;

– програми, які отримують доступ до віддалених сховищ даних (такі як клієнти електронної пошти, клієнти FTP та інструменти запитів до баз даних);
– інструменти та утиліти, які працюють з віддаленими системами (наприклад, інструменти управління системою та інструменти моніторингу мережі).

Інші варіанти включають в себе:

– **Системи клієнт-черга-клієнт.** Цей підхід дозволяє клієнтам спілкуватися з іншими клієнтами за допомогою черги на базі серверів. Клієнти можуть читати дані та надсилати дані на сервер, який просто працює як черга для зберігання даних. Це дозволяє клієнтам розподіляти та синхронізувати файли та інформацію.
– **Системи однорангової мережі (P2P).** Розроблений в стилі Client-Queue-Client, стиль P2P дозволяє клієнту та серверу обмінювати свої ролі, щоб розповсюджувати та синхронізувати файли та інформацію з кількох клієнтів. Це розширює архітектуру клієнт - сервер за рахунок декількох відповідей на запити, спільних даних, пошуку ресурсів та стійкості до видалення вузлів.
– **Сервери застосунків**. Спеціалізований архітектурний стиль, відповідно до якого сервер розміщує та виконує програми та сервіси, доступ до яких здійснюється через тонкий клієнт за допомогою браузера або програмного забезпечення. Прикладом може служити клієнт, який здійснює запити до додатоку, що працює на сервері.

Основними перевагами архітектурного шаблону клієнт - сервер є:

– Вища безпека. Всі дані зберігаються на сервері, що, як правило, забезпечує більший контроль над безпекою, ніж клієнтські комп'ютери.
– Централізований доступ до даних. Оскільки дані зберігаються лише на сервері, доступом та оновленням даних набагато легше керувати, ніж в інших архітектурних шаблонах.

– Простота обслуговування. Ролі та обов'язки обчислювальної системи розподіляються між декількома серверами, відомими один одному. Це гарантує, що клієнт не залежить від одного сервера і може бути обслужений іншим сервером у випадку ремонту, оновлення або переміщення першого.

Архітектуру клієнт - сервер доцільно використовувати у випадках необхідності забезпечення обслуговування великої кількості клієнтів, створення веб-додатків, що надаються за допомогою веб-браузерів, реалізації бізнес-процесів, які будуть використовуватися людьми по всій організації, або створенні служб для інших програм. Архітектурний шаблон клієнт - сервер також підходить, як і багато мережевих стилів, коли ви хочете централізувати функції зберігання, резервного копіювання та керування даними або коли система повинна підтримувати різні типи клієнтів та різні пристрої.

Тим не менше, традиційний 2-х рівневий архітектурний стиль клієнт-сервер має численні недоліки, включаючи тенденцію до повного об'єднання даних та бізнес-логіки на сервері, що може негативно вплинути на розширюваність системи та масштабованість, а також підвищити залежність системи від центрального сервера, що може негативно вплинути на надійність системи. Щоб вирішити ці проблеми, архітектурний стиль клієнт-сервер перетворився на більш загальний архітектурний стиль, що складається з трьох рівнів.

1.2.4. Архітектури типу «керівник-підлеглий»

Шаблон керівник-підлеглий – це ще одна фундаментальна архітектура, що використовується розробниками LabVIEW. Шаблон використовується у випадку наявності двох або більше процесів, які потрібно запускати одночасно та безперервно, але з різною швидкістю. Якщо ці процеси запускаються в одному циклі, можуть виникнути серйозні проблеми синхронізації. Ці проблеми синхронізації трапляються, коли одна частина циклу займає більше часу, ніж очікується. Якщо це станеться, інші частини циклу будуть затримані. Шаблон керівник-підлеглий складається з декількох паралельних циклів. Кожен з циклів

може виконувати завдання з різними темпами. Керівник контролює всі підпорядковані процеси і спілкується з ними за допомогою повідомлень.

Шаблон керівник-підлеглий найчастіше використовується для рекції на події користувацького інтерфейсу у процесі зборі даних. Наприклад, поставлена вимога написати програму, яка вимірює і записує швидкість зміни напруги кожні п'ять секунд. Програмне забезпечення ініціює замір сигналу з лінії передачі і відображає результат на графіку кожні 100 мс. Шаблон керівник-підлеглий добре підходить для такого випадку. У цьому додатку основна взаємодія буде відбуватися в користувацькому інтерфейсі. Замірювання напруги та ведення журналу відбудеться в одному підпорядкованому циклі, тоді як графічне виконання відбуватиметься в іншому.

Шаблон керівник-підлеглий забезпечує суттєві переваги при створенні багатозадачних додатків, що дає змогу організувати модульну структуру а також більший контроль над управління швидкістю виконання програми. У LabVIEW кожен паралельний цикл розглядається як окреме завдання або потік. Потік визначається як частина програми, яка може виконуватися незалежно від інших частин. Програма, яка не використовує окремі потоки, інтерпретується системою як один потік. Коли запит розділяється на декілька потоків, кожен потік обробляє запит почерзі у визначеному порядку. Це забезпечує кращий контроль з точки зору користувача над програмою. Паралельний характер програми LabVIEW досягається завдяки шаблону керівник-підлеглий.

Для вищезазначеного прикладу збору даних ми могли б обєднати вимірювання напруги та формування сигналу в одному циклі, при цьому виконувати вимірювання напруги можна лише на кожній 50-й ітерації циклу. Однак вимірювання напруги та збереження даних на диску може зайняти більше часу, ніж одне вимірювання та відображення форми сигналу. Якщо це так, то наступна ітерація вимрювання сигналу буде відкладена, оскільки вона не може розпочатися, перш ніж закінчиться весь код попередньої ітерації. Крім того, ця архітектура ускладнить зміну швидкості отримання сигналів без зміни швидкості збереження напруги на диск.

Стандартним підходом реалізації такого додатку відповідно до архітектури керівник-підлеглий було б налаштувати процеси вимірювання до двох окремих циклів (підпорядкованих циклів), які керуються основним циклом, який слідкує за обнолвення інтерфейсу користувача (UI), щоб побачити, чи були змінені параметри. Для взаємодії з підлеглими циклами основний цикл використовує локальні змінні, що забезпечує відстуність впливів процесім вимірювання один на одний, та також усуне можливість затримок, викликаних інтерфейсом користувача.

Шаблон керівник-підлеглий складається з декількох паралельних циклів. Цикл, який контролює всі інші, є керівником, а решту циклів - підлеглими. Один керівник завжди керує одним або кількома підлеглими циклами. Оскільки передача даних безпосередньо між цими циклами порушує цілісність даних, комунікацію слід реалізувати з використання архітектурних шаблонів обміну повідомленнями на основі черг за прикладом LabVIEW. На рис.1 відображено, як різні цикли з'єднані один з одним за допомогою спільні змінні.

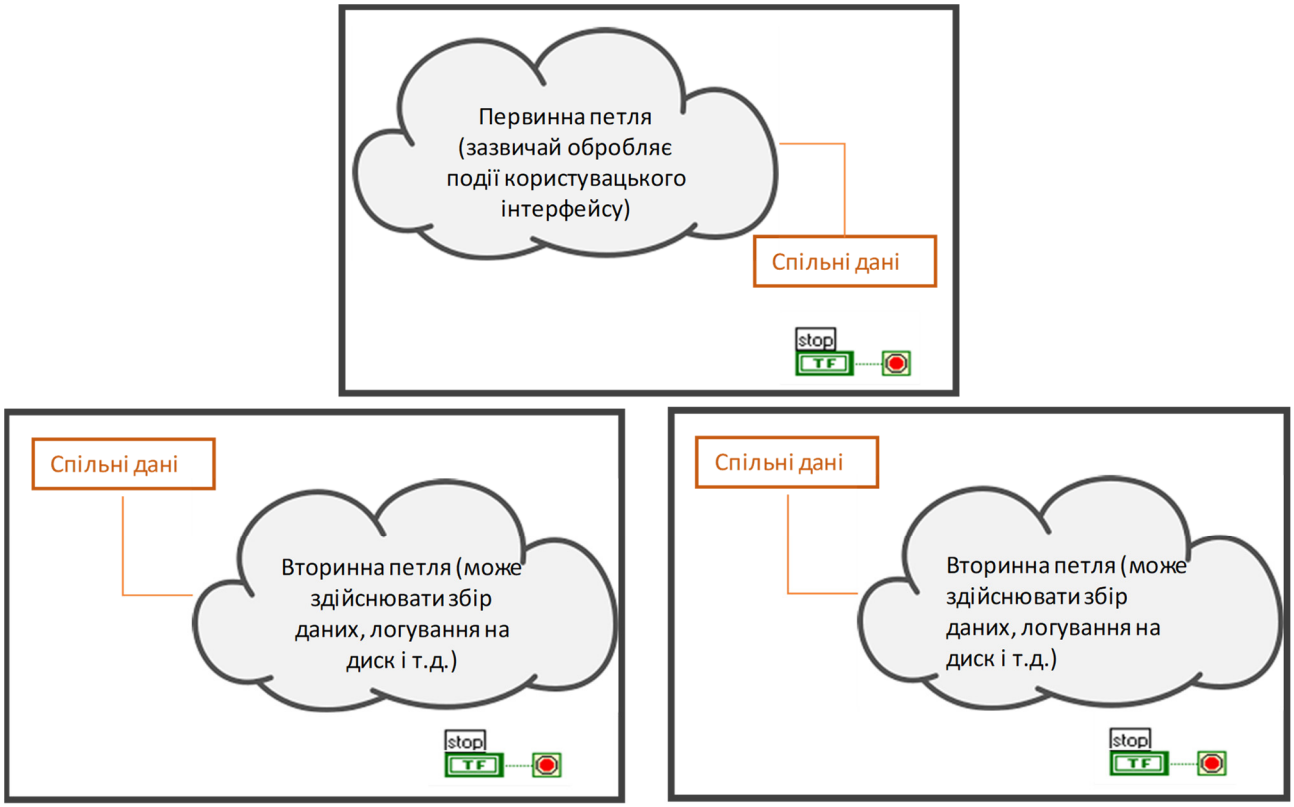


Рис. 1.6. Приклад використання шаблону «керівник-підлеглий»

У комп'ютерних мережах керівник-підлеглий – це модель для комунікаційного протоколу, в якому один пристрій або процес керує одним або декількома іншими пристроями або процесами. Після встановлення зв’язку, керування завжди здійснюється від керівника до підлеглого.

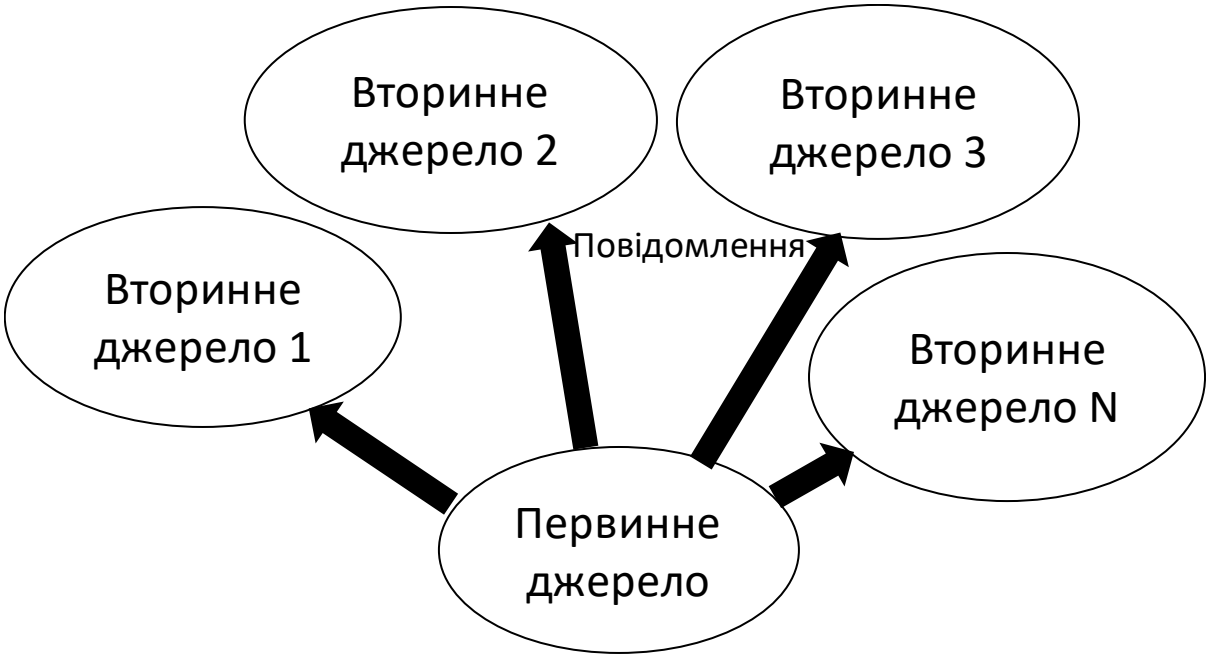


Рис. 1.7. Приклад групової комунікації на основі шаблону «керівник-підлеглий»

В області обчислювальної техніки архітектура керівник-підлеглий була започаткована у контексті реплікації баз даних. Наприклад, якщо база даних реплікується на 3-х серверах, є шанси, що кожен з них може бути не синхронізованим один з одним. З архітектурою керівник-підлеглий, одна репліка ідентифікується як керівник, а інші виконують роль підлеглих.

У випадку баз даних, клієнти будуть читати та писати в репліку керівника і всі дані будуть копіюватися у підлеглі репліки в асинхронному режимі.

Підлеглі репліки використовуються зазвичай якщо майстер зазнав невдачі.

У багатьох сучасних системах керівники використовуються як контролери, тобто роль керівника полягає лише в тому, щоб визначити, хто з підлеглих повинен обробляти запит.

В інших системах керівники виконують ті самі функції, що й усі інші репліки. У цьому випадку керівники вибираються довільно з поміж підлеглих

реплік, а якщо вибрати керівника не вдається чи керівник не відповідає, то підлеглі репліки автоматично виберуть нового керівника.

1.2.5. Архітектурні шаблони інформаційних систем на основі принципу конвеєра: обробка великих даних

Дуже проста, але потужна архітектура, що також є дуже міцною. Вона складається з довільного числа компонентів (фільтрів), які перетворюють або фільтрують дані, перш ніж передавати їх через коннектори до інших компонентів. Всі фільтри працюють одночасно. Архітектура часто використовується як проста послідовність, але вона також може бути використана для дуже складних структур [121].

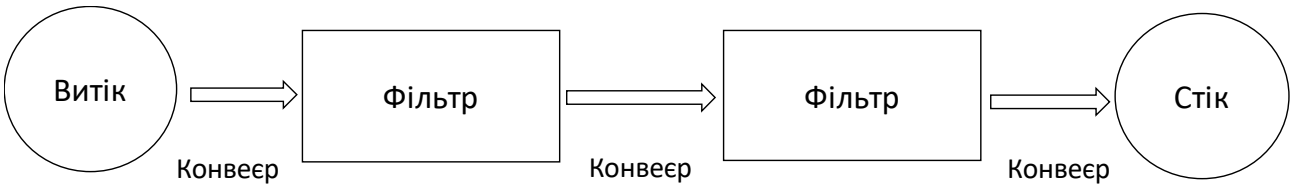


Рис. 1.8. Архітектура каналів та фільтрів

Фільтр перетворює або фільтрує отримані дані через канали, на яких він встановлений. Фільтр може мати довільну кількість вхідних каналів та довільну кількість вихідних каналів.

Канал передає дані від одного фільтра до іншого у формі спрямованого потоку даних, який, як правило, реалізується буфером даних для зберігання всіх даних, доки наступний фільтр зайнтий обробкою попереднього повідомлення.

Джерелом даних може бути статичний текстовий файл або пристрій вводу клавіатури, що постійно створює нові дані.

Споживачем може бути інший файл, база даних або екран комп'ютера [120, 122].

Приклади:

– Програми Unix. Вихід однієї програми може бути пов'язаний з входом іншої програми.

– Компілятори. Послідовні фільтри виконують лексичний аналіз, синтаксичний аналіз, семантичний аналіз та генерацію коду.

Популярність архітектури в основному пов'язана з операційною системою Unix. Вона стала популярною, тому що Кен Томсон (який створив Unix разом з Деннісом Річі) вирішив обмежити архітектуру лінійними потоками. Використання архітектури взагалі було ідеєю Дауга Макілрой, їхнього менеджера в Bell Labs.

Ця архітектура незамінна у випадку великої кількості перетворення даних, і потрібно забезпечити високу гнучкість використанні та надійність функціонування. Програма об'єднує всі входи та виходи фільтрів за допомогою каналів, а потім створює окремі пооки для кожного фільтра [156].

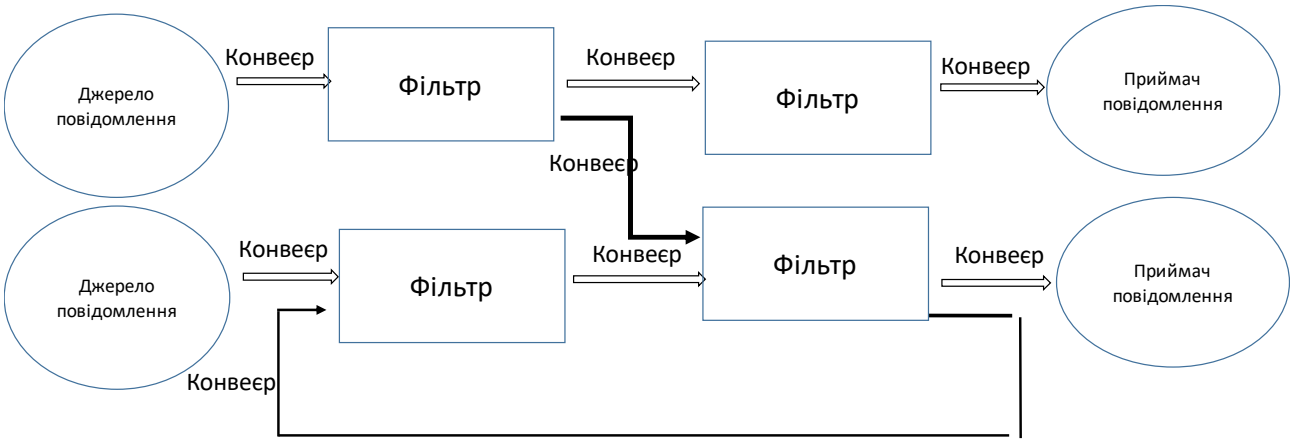


Рис. 1.9. Ілюстрація принципу незмінності архітектури каналів та фільтрів

Всі фільтри - це процеси, які запускаються (практично) одночасно. Це означає, що вони можуть працювати як різні потоки, навіть на різних серверах. Кожен канал має свою роль. Тому, якщо підключити канал, вам також потрібно вказати роль, яку цей канал буде відігравати у процесі фільтрації. Фільтри повинні бути настільки надійними, що канали можуть бути додані та видалені у процесі виконання. Щоразу, коли фільтр починає роботу з вичитування даних з вхідного каналу, виконує обробку даних та пересилає результат у вихідні канали. Якщо у вхідних каналах недостатньо даних, фільтр просто переходить в режим очікування.

Архітектура також дозволяє застосовувати рекурсивну техніку, за якої самий фільтр складається з послідовності фільтрів та каналів:

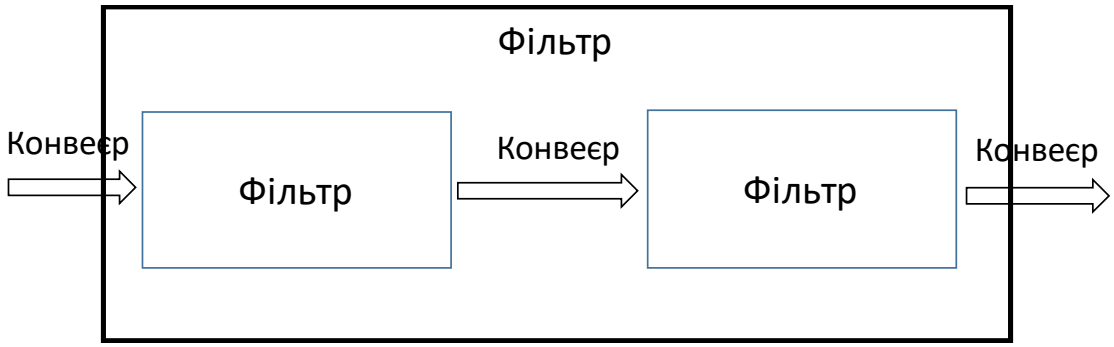


Рис. 1.10. Принцип інкапсуляції в архітектурі каналів та фільтрів

Якщо фільтр змушений очікувати на дані (наприклад, фільтр сортування), його буфер даних може стати переповненим або він заблокованим.

Якщо канали дозволяють використовувати лише один тип даних (символ або байт), фільтри повинні виконати аналіз. Це ускладнює і сповільнює обробку даних.

Фільтри зазвичай виконуються окремими потоками. Це можуть бути апаратні або програмні процеси чи потоки.

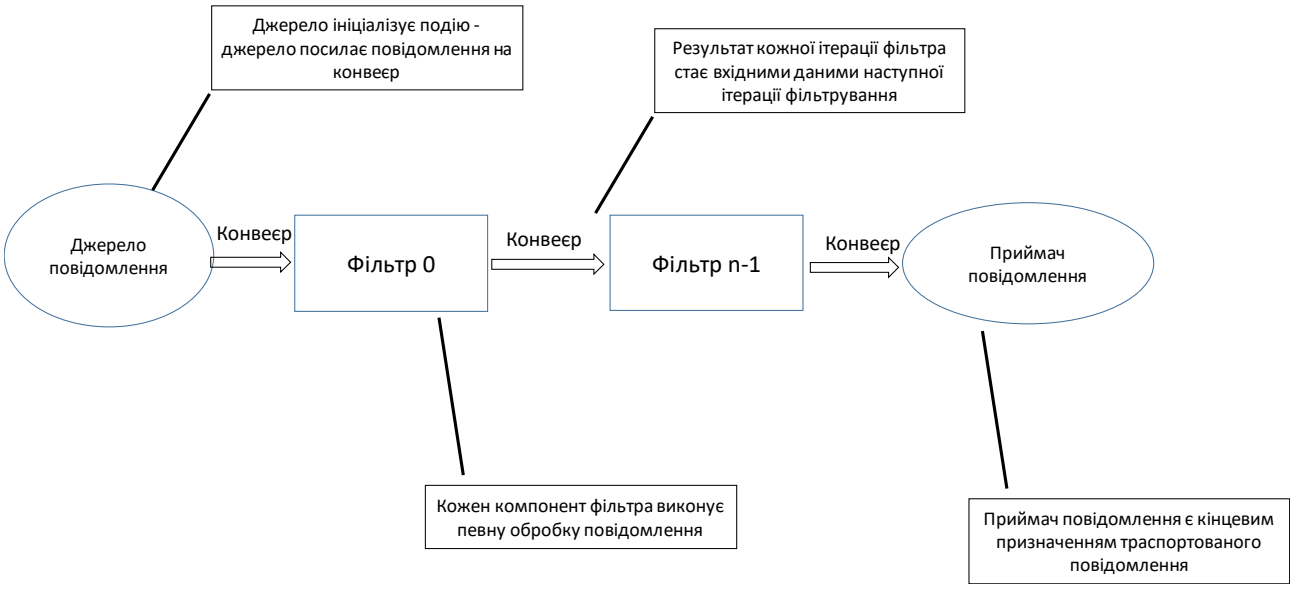


Рис. 1.11. Імплементація архітектури каналів та фільтрів

У багатьох сценаріях інтеграції, окрема подія викликає послідовність кроків обробки, кожен з яких виконує певну функцію. Наприклад, припустимо, що нове

замовлення надходить у систему у формі повідомлення. Повідомлення шифрується, щоб перешкоджати перехопленню його сторонніми особами. Повідомлення містить інформацію про автентифікацію у формі цифрового сертифіката, щоб гарантувати, що замовлення розміщується тільки довіреними клієнтами. Крім того, дублікати повідомлень можуть бути відправлені сторонніми сторонами. Щоб уникнути дублікатів повідомлень та незадоволених клієнтів, необхідно відключити можливість отримувати однакові повідомлення багаторазово, перш ніж наступні кроки обробки замовлень будуть ініційовані. Щоб задовольнити ці вимоги, необхідно перетворити потік повідомлень, що містять додаткові дані аутентифікації, у потік унікальних, простих текстових повідомлень без сторонніх полів даних [213, 214].

Соціальні мережі, такі як Facebook або Twitter, все частіше відповідають за значну частину цифрового контенту, створеного на сьогодні. Як наслідок, видавцям, зацікавленим сторонам та спостерігачам важливо розуміти та аналізувати потоки даних, що надходять з цих мереж у режимі реального часу [158, 178, 282].

Наприклад, рекламодавці спочатку опубліковували свої останні кампанії статично, використовуючи попередньо вибрані хеш-теги у Twitter. Сьогодні обробка даних у режимі реального часу відкриває двері для безперервного відстеження своєї кампанії в соціальних мережах та онлайн-адаптації опублікованого контенту для кращого взаємодії з громадськістю, наприклад, шляхом збільшення або поєднання оригінального контенту з новим контентом, або шляхом актуалізації матеріалу за допомогою нових хеш-тегів [159, 193].

Типові рішення для аналізу великих даних, такі як системи пакетної обробки даних, можуть зібльшити масштаб та тривалість очікування результату, а тому вони не підходять для онлайнового та динамічного аналізу на безперервних потоках потенційно високої швидкості.

Мета цього випадку є об'єднати та проаналізувати потоки Bitly та Twitter у режимі реального часу. Ідея полягає в тому, щоб обробляти як потоки Bitly, так і Twitter, щоб виявити тенденції та аналізувати в режимі реального часу часові

рамки твітів, ретвітів і натискань на посилання. Головним завданням є знайти систему для аналізу та змішування потоків у режимі реального часу.

Аналіз або обробка великої кількості даних паралельно, наразі, досить просте завдання завдяки розподіленим системам пакетної обробки, наприклад Apache Hadoop. Проте, Hadoop та пов'язані з ним системи переважно вирішують аспект обсягу даних і обмежуються обробкою даних за принципом окремих пакетів. Тому, обробляючи дані за допомогою Hadoop, результат роботи MapReduce, як правило, вже застарілий на кілька годин. Hadoop, безумовно, не підходить для обробки потокових даних [194, 195].

Вирішеням цієї проблеми може стати використання потокової обробки, тобто настроювання інфраструктури обробки, яка може збирати інформацію та аналізувати вхідні потоки даних безперервно та в режимі реального часу. У цьому сенсі можна використовувати кілька рішень. Системи поокових баз даних були дуже популярною темою дослідження кілька років тому. Їх комерційні партнери дозволяють користувачам створювати запити за допомогою декларативних мов (модифікований синтаксис SQL) для безперервних потоків подій. І хоча отримані системи надзвичайно ефективні, функціональні можливості таких систем суттєво обмежуються вбудованими системними операторами. Інший клас систем, що мають відношення до зазначеної проблеми це розподілені схеми обробки потоків. Ці системи, як правило, пропонують загальну цільову, розподілену платформу, яка дозволяє програмістам розробляти довільні програми для обробки безперервних і необмежених потоків даних. IBM InfoSphere7, Apache S48 або Twitter Storm – популярні приклади таких платформ.

Для описаного сценарію інтеграції та обробки потоків Twitter і Bitly буде використовуватися платформа Storm, яка підтримує дві основні функції:

– комплексний аналіз в режимі реального часу: система повинна підтримувати комплексний аналіз потоків даних безперервно та в режимі реального часу.

– розподіл та стійкість до помилок: крім того, система повинна бути здатна масштабуватися, щоб підтримувати високу швидкість обробки та забезпечувати відмовостійкість.

1.2.6. Архітектури інформаційних систем на основі підписок

Властивості архітектурного шалону на основі принципу підписок:

– Компоненти взаємодіють через оголошені події
– Компоненти можуть підписатися на сукупність подій
– Основна форма звязку це загальне середовище передавання повідомлень
– Компоненти розміщують події у середовище, присвоюючи їм певну тематику; для отримання повідомлень по певній тематиці, інші компоненти повинні зареєструвати підписку на відповідну тематику.

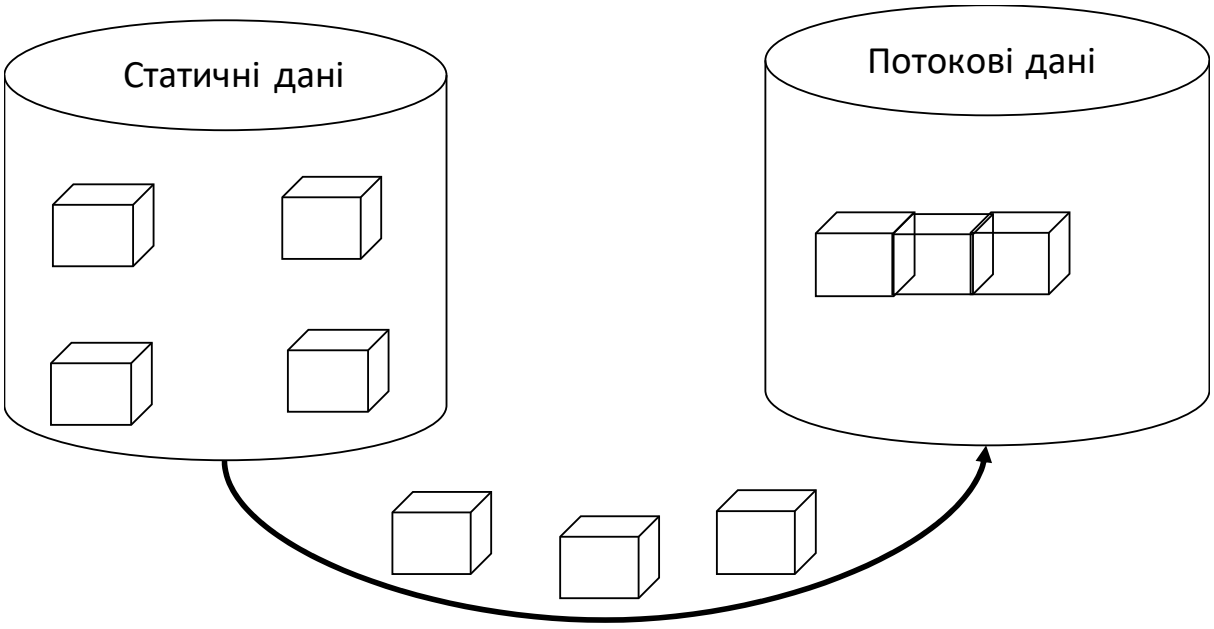


Рис. 1.12. Ілюстрація перетворення природи передавання даних у принципі підписок

Нехай існує архітектура, що складається з декількох програм. Інфраструктура зв'язку з'єднує ці додатки за допомогою спільного середовища, брокера або топології з точка-точка. Деякі програми надсилають кілька типів повідомлень. Інші програми цікавляться різними комбінаціями цих типів.

Прикладом може бути фінансова система, в якій кілька додатків підтримують інформацію про клієнтів. Система взаємодію з клієнтами (CRM) зберігає основну інформацію про кожного клієнта. Тим не менш, існує типова ситуація для інтеграційних сценаріїв, інформація про клієнтів також знаходиться в інших системах, які використовують інформацію користувача для власних потреб. Програма, з якою взаємодіє сам користувач, генерує повідомлення про оновлення інформації про клієнта, наприклад, зміна адреси клієнта. Повідомлення доставляються до системи CRM, а також до інших зацікавлених додатків, які керують інформацією про клієнтів. Однак цей тип повідомлення не має сенсу для інтегрованих програм, які не управляють інформацією про клієнтів.

Інтеграція додатків таким чином, щоб вони отримували лише повідомлення, у яких вони зацікавлені, передбачає приведення до балансу наступних умов.

Програми в складних архітектурах споживають різні типи повідомлень. Наприклад, програми, які керують інформацією про клієнтів, зацікавлені в оновленні інформації про клієнтів. Програми для торгівлі заявки зацікавлені в інформації про транзакції купівлі та продажу. Програми, які беруть участь у двофазних транзакціях, зацікавлені в повідомленні про обновлення даних в базі. Програма може надсилати кілька типів повідомлень. Наприклад, програма може надсилати інформаційні повідомлення клієнта та оперативні повідомлення про його статус. Аналогічно, програма в архітектурі інтеграції, як правило, зацікавлена лише в підмножині повідомлень, які надсилаються іншими програмами. Наприклад, менеджер портфоліо зацікавлений лише фінансовими операціями, які впливають на акції, якими він керує.

Ступінь, до якої додатки дозволяють додавати інформацію до своїх повідомлень, відрізняються. Фіксовані бінарні повідомлення зазвичай не забезпечують гнучкості або забезпечують обмежену гнучкісті в цій області.

Більшість інтеграційних архітектур інтегрують фірмові програмні рішення. Ці програми часто розроблені таким чином, щоб опрацьовувати повідомлення з невідомою структурою [279].

Відповідно до шаблону публікація-підписка на основі списку рекомендується ідентифікувати тему та підтримувати список підписників для кожної теми. Коли виникають події певної теми, необхідно негайно повідомити кожного підписника в списку підписників на цю тематику. Класичний спосіб реалізації цього дизайну описаний у шаблоні проектування Observer. Використання цього шаблону передбачає використання двох класів: об'єктів та спостерігачів. У випадку використання моделі примусового розсилання, до об'єкта додаються три методи: Attach (), Detach () та Notify (). До спостерігача додається один метод – Update ().

Для спостереження за об’єктом, всі зацікавлені спостерігачі реєструють тему, використовуючи метод Attach (). Коли відбувається зміна стану об’єкта, обєкт викликає кожного зареєстрованого спостерігача, використовуючи метод Notify (). Для детального пояснення моделі Observer див. Шаблони дизайну: елементи багаторазового об'єктно-орієнтованого програмного забезпечення.

Шаблон Observer відмінно працює у випадку створення екземплярів об'єктів, які можна чітко розділити на класи спостерігачів та об’єктів спостереження. Спостерігач особливо підходить для ситуацій, коли у вас є зв’язок між об’єктами спостереження та спостерігачами. Проте в контексті інтеграції у дуже часто є багато спостерігачів, які пов'язані з багатьма об’єктами, що ускладнює основну модель спостереження. Одним із способів реалізації таких взаємозв’язків є створення багатьох об’єктів і наявність у кожному об’єкті списку спостерігачів.

Якщо така структура об'єктів використовується для реалізації шаблону публікація-підписка, то необхідно формалізувати зв’язки між класами для можливості збереження їх у базах даних для використання в майбутньому. Для цього в реляційній базі даних необхідно додати асоціативну таблицю для збереження зв’язків між темою та спостерігачем. Після цього, отримати список підписників для довільної тематики можна прямо з таблиці бази даних.

Підтримання списків опублікованих тем (предметів) та підписників (спостерігачів), а потім інформування кожної окремо про події є основними

функціями систем відповідно до шаблону публікація-підписка на основі списків. Іншим способом досягнення такого ж результату є реалізація шаблону публікації-підписки на основі широкомовної розсилки.

У випадку використання шаблону публікація-підписка на основі широкомовної розсилки, видавець події створює повідомлення та передає його в локальну мережу (LAN). Служба на кожному вузлі перевіряє тему повідомлення. Якщо вузол підписаний на цю тему, тоді вузол обробляє повідомлення. Якщо тема не збігається, вузол ігнорує повідомлення. Тема може бути складною і сістити ієрархічнні поля, розділені комами. Вузли, які цікавляться конкретним предметом, можуть підписатися на ці поля.

Незважаючи на те, що застосування такого шаблону є ефективним способом відокремлення видавців від підписників, іноді корисно ідентифікувати підписників конкретної теми. Щоб ідентифікувати підписників теми, координуючий процес надсилає повідомлення, яке вимагає від вузлів прослуховування відповідати, якщо вони підписані на певну тему. Відповіді від кожного вузла повертаються видавцю для ідентифікації підписників. Оскільки всі повідомлення надсилаються всім вузлам прослуховування, а також тому, що кожен вузол відповідає за фільтрування небажаних повідомлень, деякі автори називають це каналом публікації/підписки з реактивною фільтрацією.

Обидва описані вище шаблони (широкомвний та на основі списку) можуть бути класифіковані як тематичні, оскільки вони обидва використовують заздалегідь визначені тематики. Останні удосконалення шаблону дають змогу визначати тематику динамічно на основі контенту, що публікується. Різниця між темами та підходами на основі змісту полягає в наступному:

У системі на основі тем, процеси обмінюються інформацією через набір попередньо визначених тем, які відповідають фіксованим логічним каналам. Системи на основі вмісту є більш гнучкими, оскільки підписки пов'язані з певним інформаційним вмістом, і тому кожна комбінація інформаційних елементів насправді може розглядатися як унікальний динамічний логічний

канал. Це експоненціальне розширення потенційних логічних каналів змінило спосіб реалізації шаблону публікація-підписка.

Практичне втілення цього підходу полягає в тому, що повідомлення направляються до кінцевого пункту призначення на основі вмісту повідомлення. Цей підхід долає обмеження системи на основі широкомовної передачі, де розподіл поєднується на основі дерева багатоадресної передачі за допомогою протоколу TCP. Це також забезпечує архітектора високою гнучкістю при прийнятті рішення щодо логіки маршрутизації на основі вмісту.

На Рис. 1.13 показано інтеграційне рішення, яке складається з чотирьох програм. Відправник (також називається видавцем) використовує підхід, що ґрунтується на темах, для публікації повідомлень на тему А та тему Б. Три підписники підписані на ці теми; один підписник підписується на тему А, інший підписується на тему В, а один підписник підписується як на тему А, так і на тему В. Стрілки показують повідомлення, що надходять від видавця до кожного підписника відповідно до цих підписок.

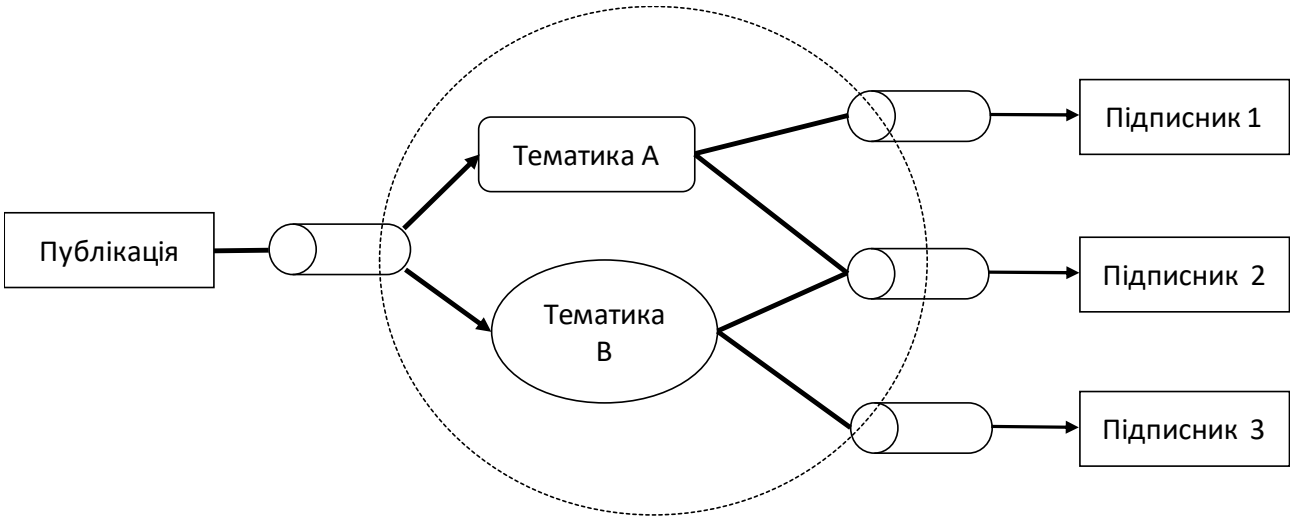


Рис. 1.13. Принцип обміну повідомленнями за архітектурою підписок

Реалізація шаблону публікації-підписки зазвичай впливає на структуру повідомлення, вбудовані програми та інфраструктуру зв'язку.

На першому етапі необхідно визначити теми або вміст, який цікавить підписників, що тягне за собою класифікацію та розбиття тематик на підкатегорії. Прикладом може бути торгова система. Деякі торговельні програми

відслідковують окермо транзакції купівлі або продажу. Інші програми відстежують обидва види транзакцій. У цьому пипадку повідомлення можна розділити на категорії купівлі та продажу, при цьому кожна категорія може мати довільне число тематик залежно від обсягу транзакції.

На наступному етапі необхідно додати інформацію до повідомлень, що вказують на тему. Іноді тему можна зберігати у невикористовуваному полі повідомлення. Крім того, можна додати нове поле для теми. Наприклад, вставити новий елемент у заголовок SOAP повідомлення. Якщо ж немає можливості використати існуюче поле чи додати нове, необхідно знайти інші способи кодування теми в повідомленні або замість цього слід використати шаблон на основі вмісту.

На наступному етапі необхідно розширити інфраструктуру зв'язку, щоб вона передавала повідомлення згідно з підпискою кожного абонента. Підхід, слід використовувати, залежить від топології рішення інтеграції. Наприклад, для інтеграції на основі спільного середовища можна реалізувати механізм підписки в самому інтерфейсі середовища передавання. Для інтеграції брокерів можна здійснити механізм підписки на основі списків. Для інтеграції типу точка-точка можна реалізувати використовуючи списки підписників у видавця.

Також необхідно змінити вбудовані програми. Видавець повинен додати інформацію, пов'язану з темою, до кожного повідомлення, яке він публікує. Наприклад, якщо тема кодується як елемент заголовка, видавець повинен вставити інформацію, пов'язану з темою, у відповідний елемент. Подібним чином, підписник повинен підписатися на теми.

Підписки можуть бути фіксованими або динамічними. З фіксованими підписками архітектор визначає теми, на які програма може підписатися. Програми не контролюють свої підписки. Зазвичай підписки визначаються на етапі дизайну інтеграційного рішення. На Рис. 1.14 відображено фіксовану підписку на тему А.

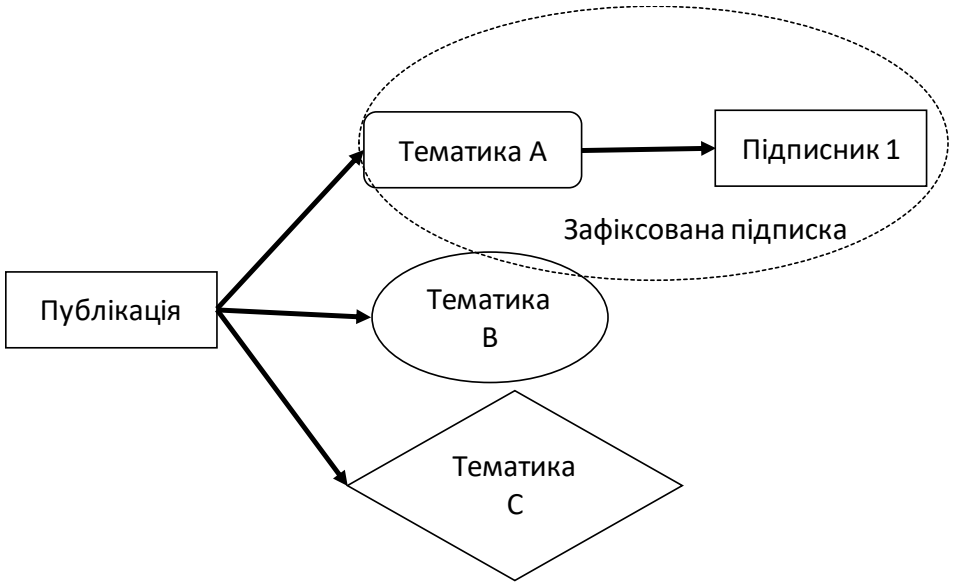
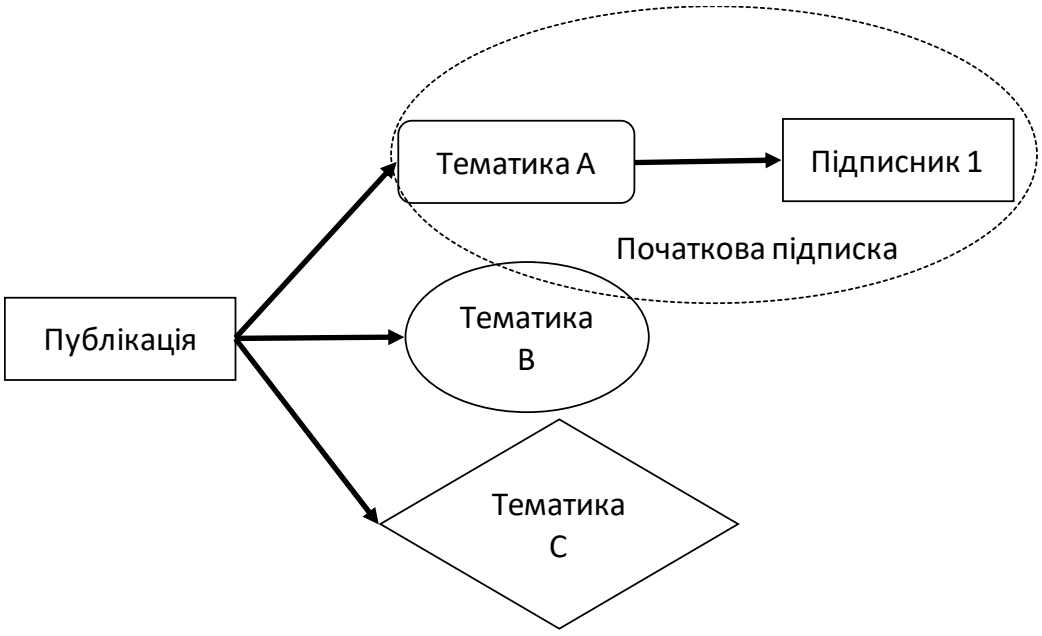
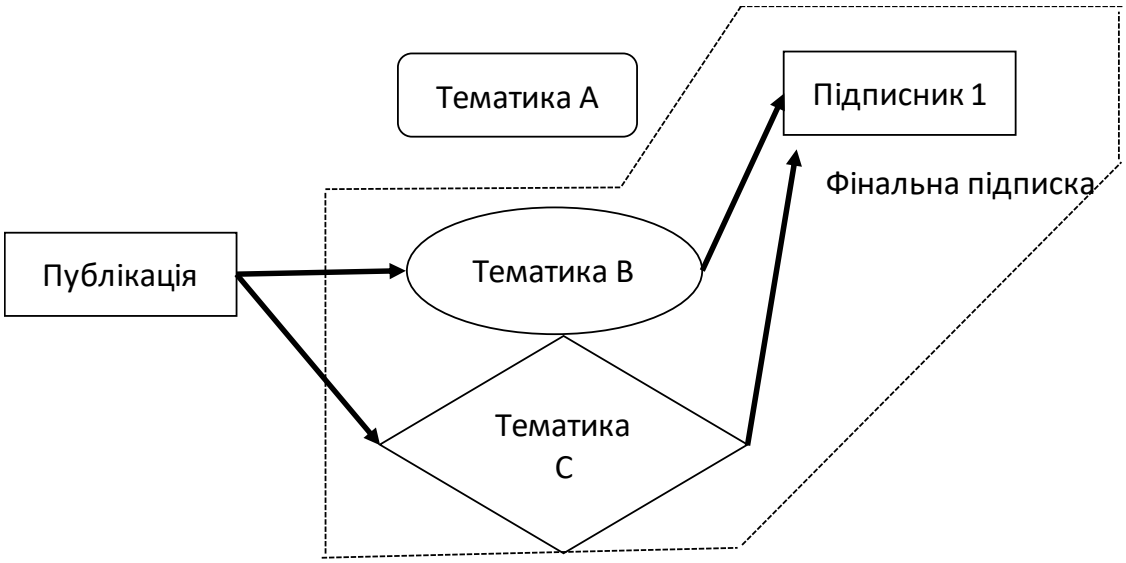


Рис. 1.14. Принцип фіксованої підписки

На противагу статичним підпискам, динамічні підписки дозволяють програмам керувати власними підписками через набір керуючих повідомлень. Програми можуть видаляти існуючі підписки, відправляючи повідомлення в інфраструктуру зв'язку, яка видаляє програму зі списку підписників. Програми можуть підписуватися на нові теми, відправляючи повідомлення інфраструктурі зв'язку, які додають програму до списку підписників. Більшість комунікаційних інфраструктур, які підтримують функції публікації і підписки забезпечують цю функцію. Однак підтримка динамічних підписок не є обов'язковою.



а)



б)

Рис. 1.15. Принцип динамічної підписки: а) початкова підписка, б) фінальна підписка

На Рис. 1.15 відображено як функціонують динамічні підписки. У верхній частині Рис. 1.15 відображено початкову підписку на тему А. Через деякий час програма надсилає повідомлення, яке видаляє її із списку підписників на тему А. Після цього програма надсилає два повідомлення, які підписують заявку на тему В та тему С. Внизу Рис. 1.15 відображено остаточну схему підписки в результаті описаних дій.

Використання шаблону публікація-підписка тягне за собою необхідність прийняття наступних рішень:

- Первинна підписка. Необхідно вирішити, яким чином підписники будуть повідомляти інфраструктуру зв'язку, коли вони вперше приєднуються до системи.
- Підписки на декілька тем. Необхідно вирішити, чи повинен механізм публікації/підписки підтримувати підписки на декілька тем.
- Статичні та динамічні підписки. Необхідно вирішити, чи повинні додатки у вашому інтеграційному рішенні динамічно змінювати свої підписки.
- Поширення тем. Необхідно вирішити, як видавці відкривають доступні теми, якщо рішення підтримує динамічні підписки.

1.2.7. Сервісно-орієнтована архітектура та принцип веб-сервісу

Сервісно-орієнтована архітектура (SOA) – це стиль розробки програмного забезпечення, де певні компоненти надають послуги іншим компонентам програми через визначений протокол комунікації. Основні принципи сервісно-орієнтованої архітектури не залежать від розробників кінцевого рішення, продуктів та технологій. Сервіс – це дискретна одиниця функціональності, до якої можна дистанційно отримувати доступ, запускати та оновлювати незалежно від інших сервісів [153, 206-212].

Сервіс має чотири властивості відповідно до одного з багатьох визначень SOA, а саме сервіс [215-217]:

- логічно представляє певгу бізнес активність із визначеним результатом.
- є самодостатнім компонентом.
- є чорнии ящиком для своїх користувачів.
- може складатися з більш ніж одного компонента.

Реалізація може використовувати один або декілька комунікаційних протоколів і використовувати файлову систему для передавання даних за відповідно до специфікації інтерфейсу між процесами, що функціонують відповідно до SOA. Ключовими елементами є незалежні сервіси з визначеними інтерфейсами, які можна викликати для виконання завдань стандартним способом, без детальної інформації про сам сервіс чи його реалізацію. SOA дозволяє розробляти додатки, побудовані шляхом поєднання вільно пов'язаних та сумісних сервісів [160].

1.2.8. Cloud архітектура інформаційних систем

Хмарні обчислення надають розробникам та IT-підрозділам можливість зосереджуватися на тому, що найбільш важливо, і уникати недиференційованої роботи, як-от закупівлі обладнання, технічне обслуговування та планування нарощування обчислювальних потужностей [119, 233]. Оскільки хмарні обчислення стали популярними, з'явилися кілька різних моделей і стратегій розгортання, які допомогли задовольнити специфічні потреби різних

користувачів. Кожен тип хмарного сервісу та метод розгортання забезпечують різні рівні контролю, гнучкості та керування. Розуміння відмінностей між Інфраструктурою як сервісом, платформою як сервісом та програмним забезпеченням як сервісом, а також стратегій розгортання, які можна використовувати, допоможе вирішити, який набір послуг відповідає потребам клієнта [112].

Приватні хмарні рішення для хостингу, також відомі як корпоративні хмари, розташовані у внутрішній мережі компанії або хостинг-центрі даних, де всі дані користувача захищені за допомогою брандмауера. Це може бути чудовим варіантом для компаній, які вже мають дорогі центри обробки даних, оскільки вони можуть використовувати свою поточну інфраструктуру [223-225]. Однак основним недоліком, який приватної хмари, є те, що відповідальність за управління, обслуговування та оновлення центрів обробки даних покладено на саму компанію. З часом сервери потрібно буде замінити, що може виявитися неекономічно. З іншого боку, приватні хмари пропонують підвищений рівень безпеки за рахунок обмежених звязків з іншими організаціями [114, 115, 155].

Основною відмінністю між публічною та приватною хмарою є те, хто відповідальний за будь-яке керування. Дані користуваів зберігаються в центрі обробки даних проайдера хмарного сервісу, і саме провайдер відповідає за управління та обслуговування центрів обробки даних. Цей тип хмарного середовища є привабливим для багатьох компаній, оскільки він скорочує час розгортання нових продуктів [143]. Проте, недоліком є те, що багато компаній відчувають, що безпека може бути під загрозою оскільки провайдер має доступ до всіх даних на своєму центрі обробки даних, чи інші компанії можуть отримати доступ до приватних не своїх даних [176, 177]. Проте, навіть якщо компанія не контролює безпеку публчіної хмари, всі дані компанії залишаються відокремленими від інших даних, що максимально мінімізує можливості порушення безпеки.

Рис. 1.16. Ідеологія Cloud обчислень

1.2.9. Агентно-орієнтована архітектура та принцип адаптації

Програмне забезпечення будується на основі програмних агентів.

Агенти мають власні інтерфейсів та можуть обмінюватися повідомленнями.

Повідомлення інтерпретуються інтерпретують агентами, що їх отримують.

Архітектрура на основі агентів є одним із класів обчислювальних архітектур для моделювання взаємодій автономних агентів (як індивідуальних, так і колективних об'єктів, таких як організації чи групи) з метою оцінки їх впливу на систему в цілому. Архітектура поєднує елементи теорії ігор, складних систем, обчислювальної соціології, мультиагентних систем та еволюційного програмування. Метод Монте-Карло використовується для введення випадковості. Огляд нещодавньої літератури, присвяченої архітектурам на основі агента та багатоагентних систем показує, що зазначені архітектури використовуються в не пов'язаних з обчисленням наукових областях, включаючи біологію, екологію та соціальні науки. Моделювання на основі агента пов'язане з концепцією багатоагентних систем, але відрізняється від цих систем тим, що метою моделювання на основі агента є поращення розуміння колективної поведінки агентів, що дотримуються простих правил, таких, які закладені у природних системах [154, 161].

1.3. Комунікація в гетерогенних інформаційно-комунікаційних системах

1.3.1. Синхронізація часу в розподілених інформаційних системах на основі часових міток Лампорта

Алгоритм Lamport timestamps – це простий алгоритм, який використовується для визначення послідовності подій у розподіленій комп'ютерній системі. Оскільки різні вузли або процеси, як правило, не ідеально синхронізовані, цей алгоритм використовується для забезпечення часткового упорядкування подій з мінімальними витратами та концептуально забезпечує відправну точку для удосконаленого методу векторного годинника. Відмітки названо на честь їх творця Леслі Лампорт [163, 275].

Розподілені алгоритми, такі як синхронізація ресурсів, часто залежать від способу упорядкування подій. Наприклад, розглянемо систему з двома процесами та диском. Процеси відправляють повідомлення одне одному, а також надсилають запит на доступ до диску [37]. Диск надає доступ у порядку надходження запитів. Наприклад, процес А посилає на диск запит на доступ до запису, а потім надсилає команду для читання для обробки процесу В. Процес В отримує повідомлення, і в результаті надсилає власний запит на читання на диск. Якщо є затримка часу, через яку диск може одержувати обидва повідомлення одночасно, він може визначити, яке повідомлення надійшло першим: повідомлення від процесу А було створено перед повідомленням від процесу В, якщо можна отримати від А до В послідовністю дій двох типів: рухатися вперед, залишаючись у тому ж процесі, та відслідковувати шлях повідомлення від його відправлення до його прийому. Алгоритм логічного годинника забезпечує механізм для визначення порядоку подібних подій.

Алгоритм. Алгоритм відповідає кільком простим правилам:

- Процес збільшує свій лічильник перед кожною подією в цьому процесі;
- Коли процес надсилає повідомлення, він включає в повідомлення значення свого лічильника;

При отриманні повідомлення лічильник одержувача оновлюється, у випадку, якщо його власний лічильник має менше значення. В іншому випадку, значення в повідомленні оновлюється значенням лічильника отримувача. Після цього лічильник збільшується на 1, і тільки тоді повідомлення вважається отриманим [291].

Алгоритм відправлення, записаний у форматі псевдо коду:

```
time = time+1;
time_stamp = time;
send(message, time_stamp);
```

Алгоритм отримання, записаний у форматі псевдо коду:

```
(message, time_stamp) = receive();
time = max(time_stamp, time)+1;
```

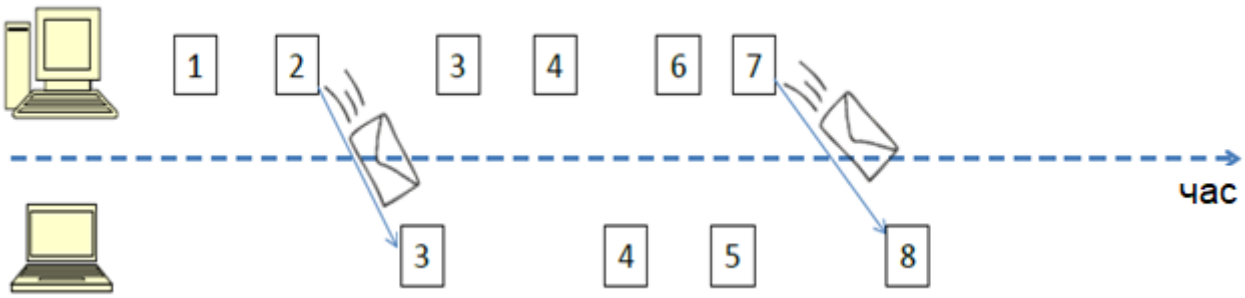


Рис. 1.17. Принцип роботи механізму часових міток Лампорта

1.3.2. Комунікація на основі принципу спільної пам’яті

У комп'ютерному програмуванні спільна пам'ять це спосіб, за допомогою якого процеси програм можуть обмінюватися даними швидше, ніж шляхом читання та запису за допомогою звичайних служб операційної системи. Наприклад, процес клієнта може містити дані для передачі на серверний процес, який повинен змінювати ці дані та повертатися результат до клієнта. Як правило, необхідно, щоб клієнт записував дані у вихідний файл (використовуючи буфери операційної системи), після чого сервер зчитував цей файл як вхідний з буферів у власний робочий простір. Використовуючи область спільної пам'яті, ці дані можна зробити безпосередньо доступними для обох процесів без використання системних служб. Щоб помістити дані в загальну пам'ять, клієнт отримує доступ до загальної пам'яті після перевірки значення семафора, записує дані, а потім

скидає семафор, щоб сигналізувати серверу (який періодично перевіряє спільну пам'ять для можливого введення), що дані очікують. У свою чергу, серверний процес записує дані назад у область спільної пам'яті, використовуючи семафор, щоб вказати, що дані готові до читання.

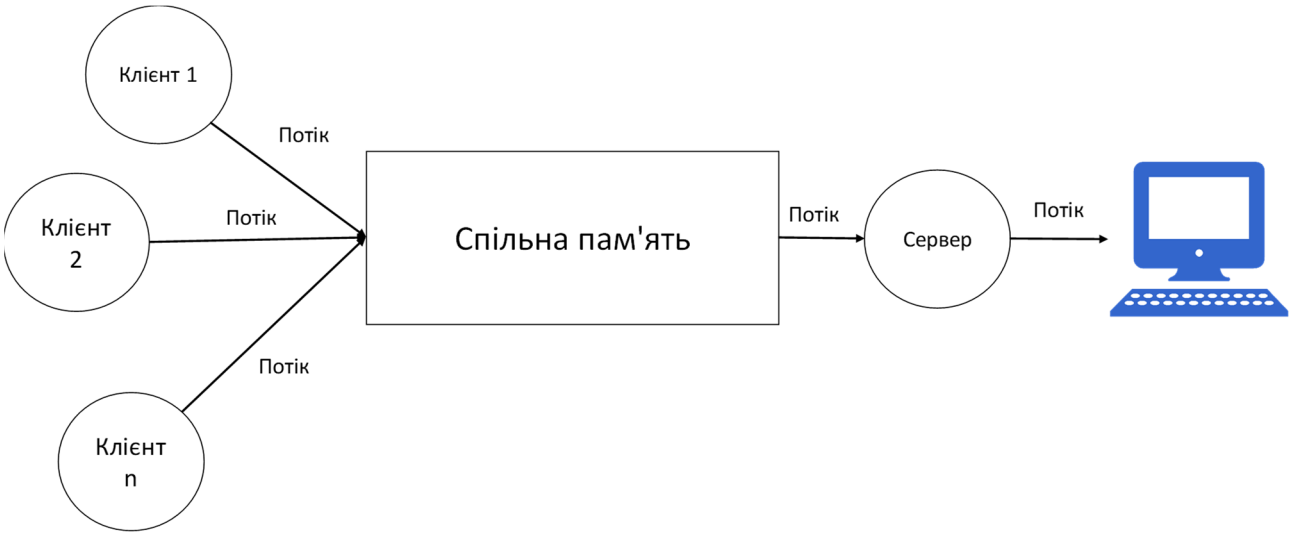


Рис. 1.18. Комунікація на основі принципу спільної пам’яті

1.3.3. Комунікаційні інтерфейси, орієнтовані на передавання даних

OpenMP (Open Multi-Processing) це інтерфейс прикладного програмування (API), який підтримує багатопроцесорне програмування з всикористанням спільної пам'яті в C, C ++ та Fortran на більшості платформ, архітектур та операційних систем, включаючи Solaris, AIX, HP -UX, Linux, macOS і Windows. Він складається з набору директив компіляторів, бібліотечних процедур та змінних середовища, які впливають на поведінку під час виконання.

OpenMP керує неприбутковий технологічний консорціум OpenMP Architecture Review Board (або OpenMP ARB), спільно визначений групою основних постачальників комп'ютерних та програмних продуктів, включаючи AMD, IBM, Intel, Cray, HP, Fujitsu, Nvidia, NEC, Red Hat , Texas Instruments, Oracle і багатьма іншими. OpenMP використовує портативну масштабовану модель, яка надає програмістам простий та гнучкий інтерфейс для розробки паралельних додатків для платформ, починаючи від стандартного настільного комп'ютера до суперкомп'ютера.

Система на основі дощок є підходом штучного інтелекту на основі архітектурної моделі дошки, в якій загальна база знань, "чорна дошка", ітеративно оновлюється різноманітною групою спеціалізованих джерел знань, починаючи з специфікації проблеми та закінчуючи рішенням. Кожне джерело знань оновлює дошку з частковим рішенням, коли його внутрішні обмеження співпадають зі станом дошки. Таким чином фахівці працюють разом для вирішення проблеми. Модель дошки була спочатку спроектована як спосіб обробки складних, погано визначених завдань, де рішення являє собою суму його частин.

Додаток на основі дошки складається з трьох основних компонентів. Модулі програмного забезпечення, які називаються джерелами знань (KSs). Подібно до експертів в конкретній галузі, кожне джерело знань забезпечує специфічну кваліфікацію.Сама чорна дошка, яка виступає загальним сховищем проблем, часткових рішень, пропозицій та отриманої інформації. Чорна дошка може розглядатися як динамічна "бібліотека" внесків інформації, які нещодавно були "опубліковані" іншими джерелами знань щодо поточної проблеми. Керуюча оболонка, яка контролює потік вирішення проблем в системі. Подібно до того, як команда людей потребує модератора, KSs потребує механізму для упорядкування їх дій. У цій архітектурі це забезпечується керуючою оболонкою.

Потокове передавання даних це передавання даних зі стійкою високою швидкістю, достатньою для підтримки таких програм, як телебачення високої чіткості (HDTV) або безперервне резервне копіювання на носій даних. Потокове передавання даних вимагає наявності достатньої пропускної здатності мережі та можливості переконатись, що дані передаються відповідно до вимог і параметнри потоку не відхиляються від норми.

У програмуванні комп'ютера програмування потоків даних являє собою парадигму програмування, яка моделює програму як спрямований потік даних, що протікає між операціями, тим самим впроваджуючи принципи потоку даних та архітектуру. Мови програмування потоків даних поділяють деякі функції

функціональних мов, і, як правило, розробляються для того, щоб привести деякі функціональні поняття до мови, більш придатної для числової обробки.

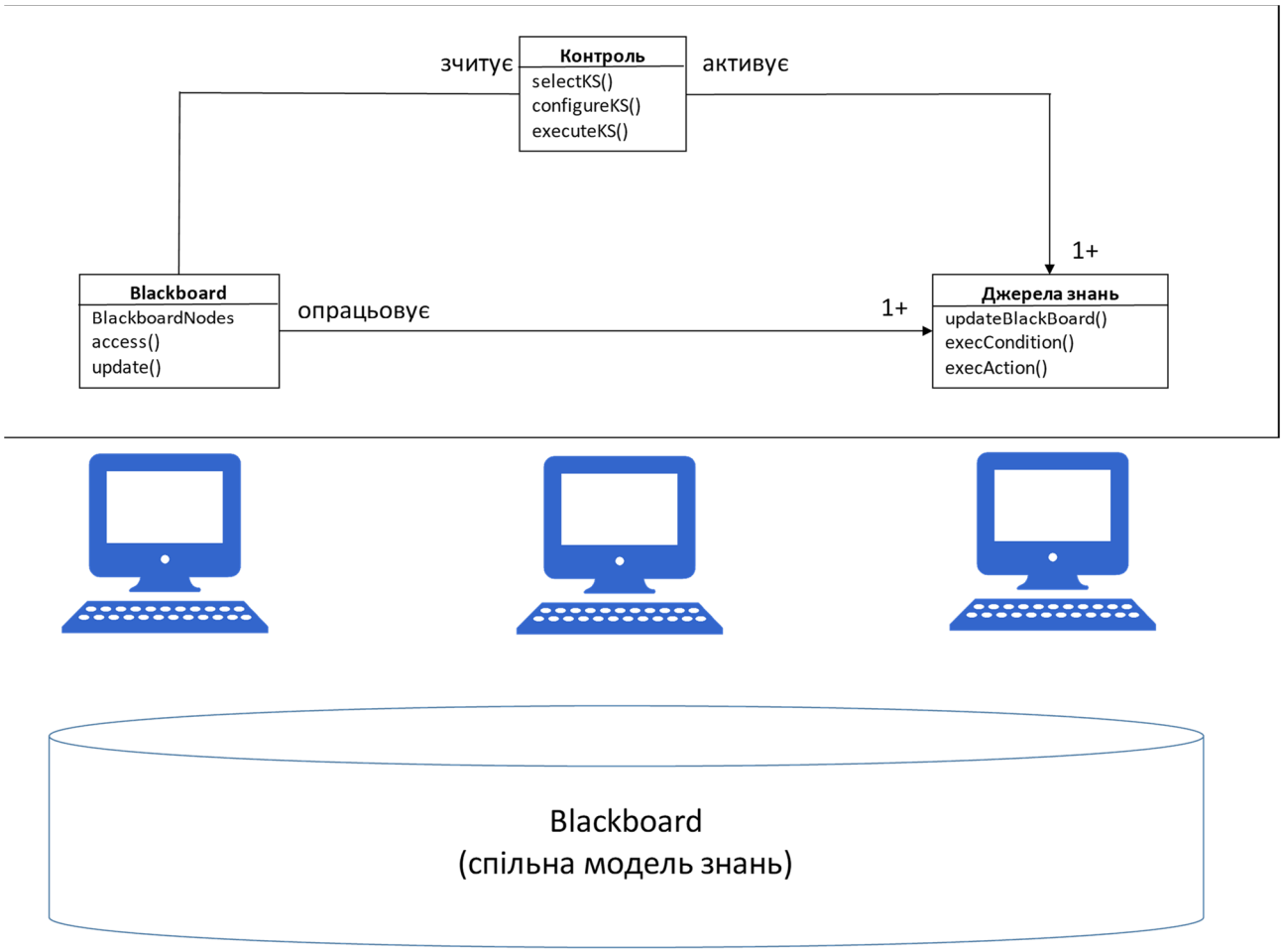


Рис. 1.19. Принцип комунікації на основі спільної дошки

Деякі автори використовують термін Datastream замість потоку даних, щоб уникнути плутанини з архітектурою Dataflow Computing або Dataflow на основі недетермінованої машинної парадигми [31].

Необхідно здійснювати перетворення, використовуючи послідовність фільтри, де кожен фільтр отримує вхідне повідомлення, застосовує просте перетворення та надсилає перетворене повідомлення на наступний фільтр. Передавання повідомлень відбувається через канали, які з'єднують виходи та входи фільтрів. Канали також можуть забезепчувати функцію буферизації. У лівій частині рис.1 відображено схему, що складається з двох фільтрів. Програма джерело даних генерує повідомлення і передає його через канал у перший фільтр. Фільтр перетворює отримане повідомлення, а потім посилає його на

вихід у настпний канал. Канал переносить перетворене повідомлення до другого фільтру.

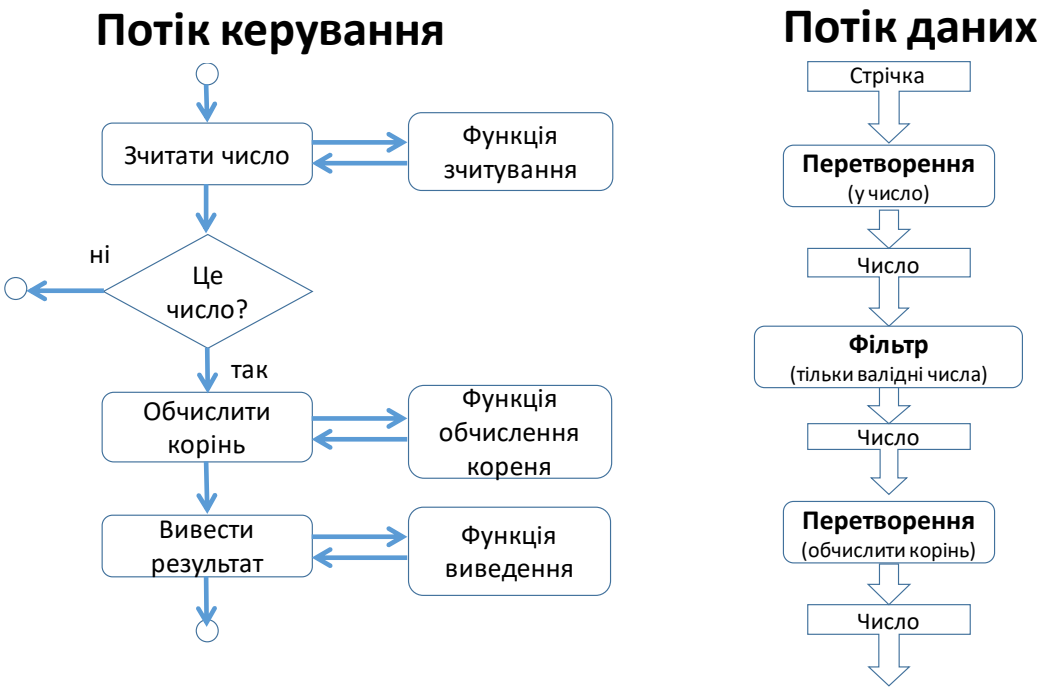


Рис. 1.20. Принцип потокового програмування

Канал також буферизує повідомлення, які надсилає фільтр 1, у випадку, якщо фільтр 2 зайнятий обробкою попередніх повідомлень. Другий фільтр застосовує своє перетворення і передає повідомлення через канал до програми, яка очікує на філнальний результат перетворення. Така схема вимагає наступного:

- Формат вихідного повідомлення джерела повинен відповідати формату вхідного повідомлення першого фільтру.
- Формат вихідного повідомлення першго фільтра повинен відповідати формату вхідного повідомлення другого фільтру.
- Формат вихідного повідомлення другого фільтру повинен відповідати формату вхідного повідомлення отримувача результату фільтрування.

1.3.4. Комунікація в інформаційних системах на основі принципу обміну повідомленнями

Загалом, обмін повідомленнями (також називаються електронними повідомленнями) це створення, зберігання, обмін та управління текстом, зображеннями, голосом, факсом, електронною поштою, пейджинговим зв'язком та електронним обміном даними (EDI) через комунікаційну мережу.

У програмуванні обмін повідомленнями це обмін повідомленнями (спеціально відформатованими даними, що описують події, запити та відповіді) на сервер повідомлень, який діє як програма обміну повідомленнями для клієнтських програм. Є дві основні моделі сервера повідомлень: модель "точка-точка" та модель опублікування / підписки. Обмін повідомленнями дозволяє програмам спільно використовувати загальний код обробки повідомлень, щоб виділити ресурси та взаємозалежність, а також легко керувати збільшенням кількості повідомлень. Обмін повідомленнями також полегшує взаємодію між програмами, що виконуються на різних платформа, оскільки формат і протокол обміну повідомленнями є іпльним для всіх.

IBM MQSeries та служба обробки повідомлень Java від Sun Microsystems (JMS) - це приклади продуктів, які забезпечують обмін повідомленнями.

Комунікація на основі повідомлень дозволяє відкривати сервіси для публічного доступу, описуючи інтерфейс сервісу, методи якого клієнт може викликати, передаючи повідомлення у форматі XML через транспортний канал. Такі виклики, як правило, виконуються з віддалених клієнтів, але інтерфейси на основі повідомлень також можуть підтримувати локальні виклики. Стиль спілкування на основі повідомлень добре підходить для таких сценаріїв:

Реалізація бізнес-системи, яка представляє середньо та довгострокові інвестиції; наприклад, сервісу, який протягом тривалого часу використовуватиметься партнерами.

Реалізація великомасштабної системи, яка мають забезпечувати високу доступність або повинна працювати через ненадійні мережі. У цьому випадку

механізм зберігання повідомлень і переадресація може забезпечити підвищену надійність.

Створення сервісу, який необхідно ізолювати від інших сервісів, які основний сервіс буде використовувати. Використання службових інтерфейсів на основі повідомлень, які поширюють деталі інтерфейсу для клієнтів, дозволяє будь-яким клієнтам легко користуватися бажаним сервісом, не потребуючи конкретних реалізацій для окремих клієнтів.

Реалізація бізнес процесів, які використовують асинхронну модель.

Необхідно врахувати наступні аргументи при використанні комунікації на основі повідомлень:

Підключення до мережі може бути відстунім, і повідомлення, можливо, доведеться зберігати, а потім надсилати, коли з'єднання відновиться.

Необхідно врахувати випадок, коли відповідь на ваідправлене повідомлення не не надходить. Щоб керувати станом бесіди, бізнес логіка може зареєструвати відіслані повідомлення для подальшої обробки, якщо відповідь не буде отримана.

Обдумати використання підтверджень, щоб забезпечити правильну послідовність та цілісність повідомлень.

Варто використовувати стандартні протоколи, такі як HTTP для комунікації через Інтернет та TCP для внутрішнього зв'язку. Не варто реалізовувати власний канал комунікації, якщо немає чіткого опису протоколу та інтерфейсу, що відповідає потребам бізнес логіки.

Якщо тривалість очікування на відповідь є критичною для спілкування, варто розглянути модель синхронної комунікації, в якій клієнт очікує відповідь на кожне повідомлення [104]. Якщо ж клієнти можуть продовжувати виконання під час очікування відповіді, варто розглянути асинхронну модель.

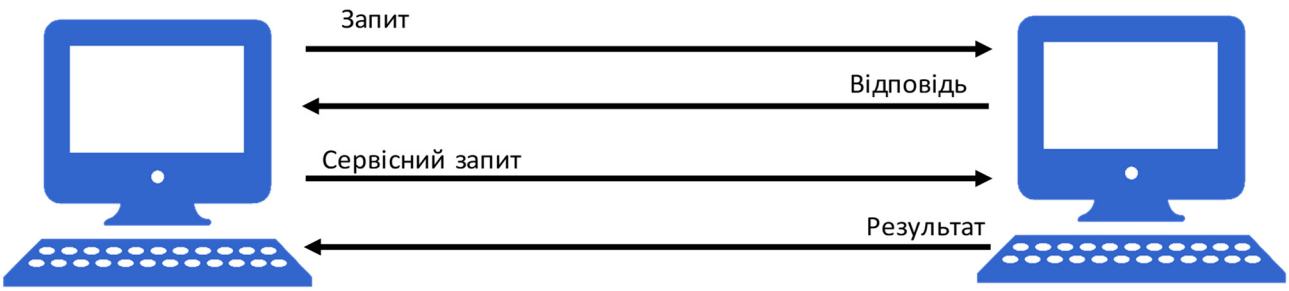


Рис. 1.21. Ілюстрація процесу обміну повідомленнями

1.3.5. Інтерфейс передавання повідомлень

Інтерфейс передачі повідомлень (MPI) - це стандартизована та портативна система передавання повідомлень, розроблена групою науковців та дослідників для роботи на різноманітних паралельних обчислювальних архітектурах. Стандарт визначає синтаксис та семантику ядра програми, що має на меті розпаралелювати процеси виконання між декількома процесорами. Існує кілька добре перевірених та ефективних реалізацій MPI, які є відкритими. MPI сприяло розвитку багатопотокового програмного забезпечення та сприяло розробці портативних і масштабованих розподілених додатків.

Інтерфейс MPI призначений для забезпечення основи віртуальної топології, синхронізації та зв'язку між набором процесів (які були віднесені до вузлів / серверів) незалежно від мови. Програми на основі MPI завжди працюють з процесами, але програмісти під процесом зазвичай мають на увазі процесор. Як правило, для максимальної продуктивності кожен процесор (або ядро в багатоядерній машині) буде призначено лише одному процесу. Це завдання відбувається під час виконання через агент, який запускає програму MPI, яка зазвичай називається mpirun або mpiexec.

Функції бібліотеки MPI включають в себе реалізацію комунікації типу точка-точка, вибір топології розподілення процесів, обмін даними між процесами (операції відправлення / отримання), об'єднання часткових результатів обчислень (збирання та зменшення операцій), синхронізацію вузлів. Бібліотека MPI також відповідає за отримання інформації, пов'язаної з мережею, такою як кількість процесів в обчислювальній сесії, тотожність ідентифікатора

процесора, сусідні процеси, доступні в логічній топологія тощо. Операції «точка-точка» включають синхронні, асинхронні, буферизовані та готові форми, що дозволяють як відносно сильнішу, так і слабку семантику для аспектів синхронізації. Багато важливих операцій доступні в асинхронному режимі в більшості реалізацій.

MPI-1 та MPI-2 уможливлюють реалізації, які покривають функції спілкування та обчислення, але практика та теорія відрізняються. MPI також визначає інтерфейси безпечні для викуористання багатьма потоками. Багатопотокове колективне спілкування найкраще виконується кількома копіями комунікаторів, як описано нижче.

Ряд важливих функцій MPI передбачає зв'язок між двома специфічними процесами. Популярним прикладом є MPI_Send, який дозволяє одному зазначеному процесу відправити повідомлення на другий вказаний процес. Операції "точка-точка", як їх називають, є особливо корисними у розподіленій архітектурі, в якій кожен процесор регулярно обмінюється даними з іншими процесорами в процесі виконання своїх завдань.MPI-1 визначає механізми блокування та уникнення блокування комунікації точка-точка, а також так званий механізм готового відправлення, за допомогою якого запит на відправлення може бути здійснений лише тоді, коли запит на отримання відповідного запиту вже було зроблено.

1.3.6. Проміжне програмне забезпечення інформаційних систем, орієнтоване на обмін повідомленнями

Оскільки підприємства, установи та технології постійно змінюються, програмні системи, які обслуговують їх, повинні мати можливість враховувати такі зміни. Саме у цій критичній точці потрібно інтегрувати нові компоненти або масштабувати існуючі, наскільки це можливо. Найпростіший спосіб інтегрувати гетерогенні компоненти це не адаптувати їх для взаємодії, а надати їм проміжний компонент для комунікації, який дозволить їм спілкуватися, незважаючи на їх відмінності. Цей компонент називається проміжним програмним забезпеченням та дозволяє взаємодіяти між собою програмними

компонентами (додатками, корпоративними Java-компонентами, сервлетами та іншими компонентами), які розробляються незалежно та працюють на різних мережевих платформах.

Як відображено на рисунку, концептуально проміжне програмне забезпечення знаходиться між самою програмою та операційною системою [70].

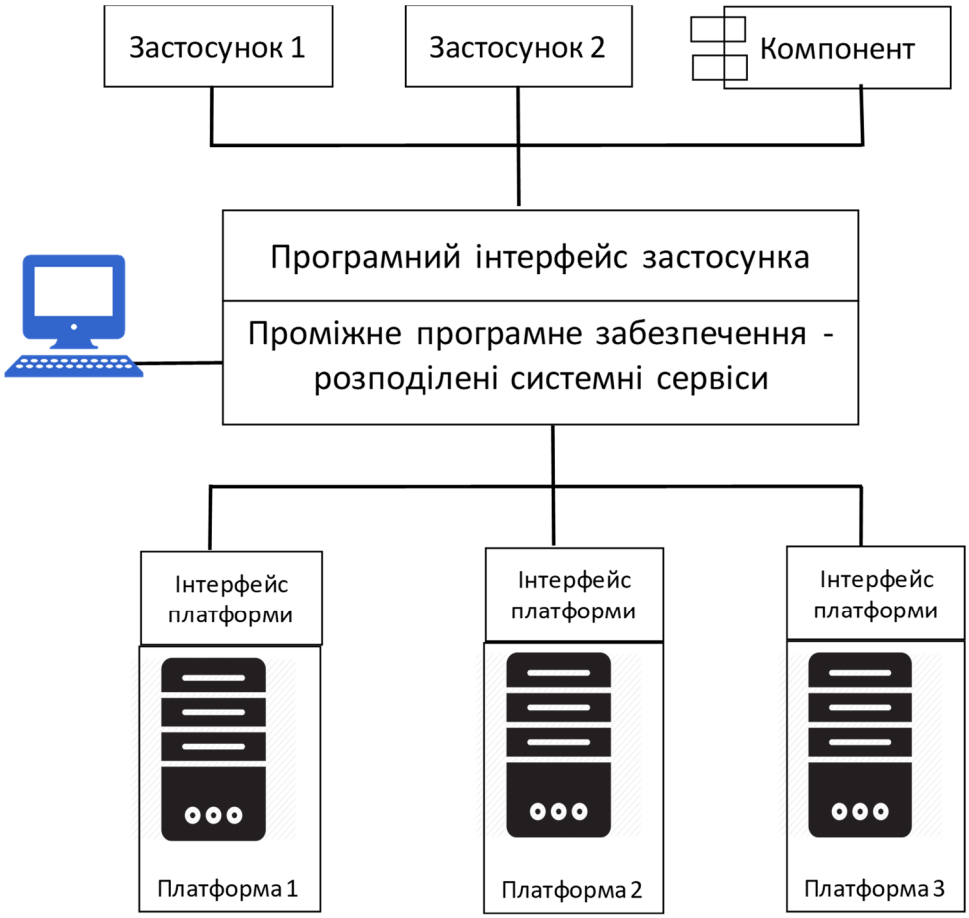


Рис. 1.22. Роль проміжного програмного забезпечення, орієнтованого на обмін повідомленнями

Програми, що встановлені на різних мережних вузлах, використовують інтерфейси для спілкування, при цьому не вникаючи в деталі платформ на яких вони виконуються. Така віртуальна система взаємопов'язаних програм може бути надійною та захищеною. Її продуктивність може бути виміряна та налаштована і її можна масштабувати без втрати функціональності.

Віддалений виклик процедури RPC. Дає змогу процедурам в одній програмі викликати процедури в віддалених програмах, так ніби останні виконуються на тій самій машині. Проміжне програмне забезпечення реалізує механізм зв'язку,

який знаходить віддалені процедури та робить їх доступними для абонента. Традиційно цей тип проміжного програмного забезпечення використовується процедурними програмами, проте тепер також включає об'єктні компоненти. Broker Request Object або проміжне програмне забезпечення на основі ORB, яке дозволяє розподіляти та ділитися об'єктами програми через мережу. Проміжне програмне забезпечення дозволяє розподіленим додаткам здійснювати зв'язок та обмін даними, надсилаючи та отримуючи повідомлення.

Всі ці моделі дозволяють одному програмному компоненту впливати на поведінку іншого компонента через мережу. Вони відрізняються тим, що проміжне програмне забезпечення RPC та ORB створює системи тісно пов'язаних компонентів, тоді як системи на базі MOM дозволяють більш слабкий звязок компонентів. У системі RPC або ORB, коли одна процедура викликає іншу, вона повинна зачекати повернення викликаної процедури, перш ніж зможе зробити іншу дію. У синхронних моделях повідомлень проміжні програми функціонують як посередники, що перехоплюють виклик передають його мережею та повертають результат.

У розподілених обчисленнях віддалений виклик процедури (RPC) це коли комп'ютерна програма викликає процедуру (підпрограму) для виконання в іншому адресному просторі (зазвичай на іншому комп'ютері), без необхідності явної імплементації деталей віддаленої взаємодії. З точки зору програміста необхідно написати такий самий код, як у випадку виклику процедури локально. Така форма взаємодії між клієнтом і сервером, як правило, реалізується через систему передачі запитів-відповідей на основі повідомлень. Об'єктно-орієнтованим аналогом є віддалений виклик методу (RMI). Модель RPC забезпечує певний рівень прозорості розташування, а саме, процедури викликів в основному однакові незалежно від того виклики локальний чи віддалений, проте є відміності за якими локальні виклики можна відрізняти від віддалених. Віддалені виклики зазвичай набагато повільніші і менш надійні ніж локальні.

RPC це форма зв'язку між процесами (IPC) і полягає в тому, що різні процеси мають різні адресний простір: на одній і тій же машині вони можуть

мати різний віртуальний адресний прострі, навіть якщо фізичний адресний простір однаковий; а якщо вони знаходяться на різних машинах та і фізичний адресний простір відрізняється. Для реалізації концепції було використано багато різних (часто несумісних) технологій.

RPC це свого роду протокол відповіді-запиту. RPC ініціюється клієнтом, який надсилає запит до віддаленого сервера для виконання певної процедури з наданими параметрами. Віддалений сервер надсилає відповідь клієнту, і програма продовжує виконання. Поки сервер обробляє виклик, клієнт блокується (він чекає, поки сервер не закінчить обробку, перш ніж відновити виконання). Якщо клієнт надсилає асинхронний запит на сервер, наприклад, виклик XHTTP, то виконання клієнта продовжиться одразу після відправлення повідомлення.

Важливою відмінністю між віддаленими викликами та локальними викликами є те, що віддалені виклики можуть зазнати невдачі через непередбачувані мережні проблеми. Тому такі програми повинні вміти обробляти подібні випадки, наприклад якщо віддалена процедура не змогла бути викликаною.

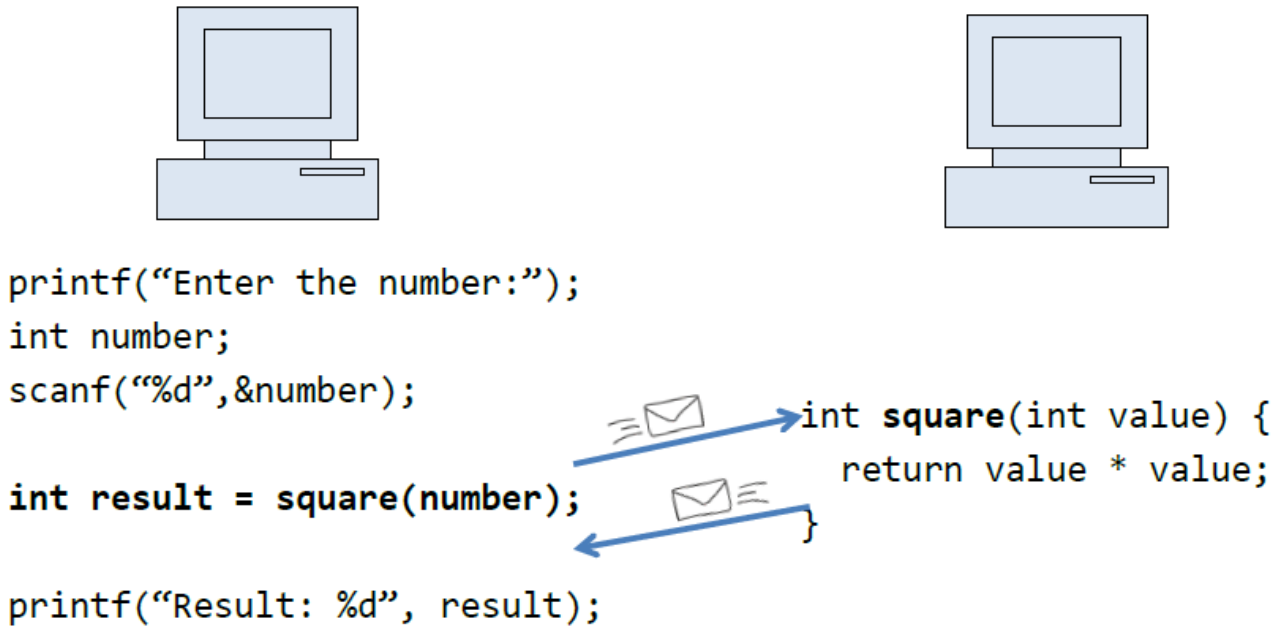


Рис. 1.23. Ілюстрація принципу роботи RPC

Java Remote Method Invocation (Java RMI) це API, що виконує виклик віддаленого методу, і є об'єктно-орієнтованим аналогом викликів віддалених процедури (RPC) з підтримкою прямої передачі класів Java та розподіленої збору та видалення непотрібних обєктів. Реалізація залежить від механізмів представлення класів у Java Virtual Machine (JVM) а тому підтримує лише виклики з однієї машини на іншу. Протокол, що лежить в основі цієї реалізації, називається Java Remote Method Report (JRMP). Для підтримки коду, що виконується не в віртуальній машині JVM, було розроблено проміжне програмне забезпечення CORBA. Використання терміну RMI може означати виключно інтерфейс програмування або може означати як API, так і JRMP, IIOP чи іншу реалізацію, тоді як термін RMI-IIOP спеціально позначає інтерфейс RMI, який делегує більшість функцій реалізації CORBA.

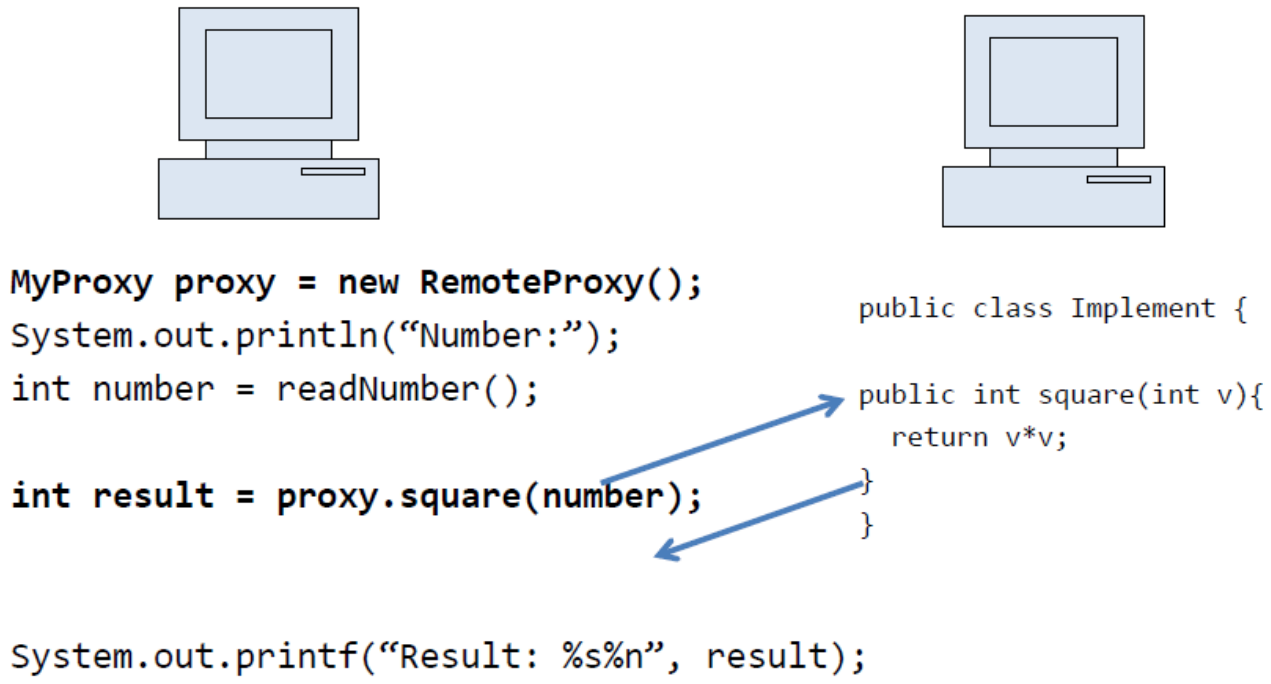


Рис. 1.24. Ілюстрація принципу роботи RMI

CORBA це стандарт, визначений групою OMG та призначений для полегшення зв'язку систем, що розгортаються на різних платформах. CORBA дає змогу співпрацювати між різними операційними системами, мовами програмування та обчислювальною технікою. CORBA використовує об'єктно-

орієнтовану модель, хоча системи, які використовують CORBA, не повинні бути об'єктно-орієнтованими. CORBA є прикладом парадигми розподілених об'єктів.

Основою CORBA є брокер запиту об'єктів (ORB). Підтримка ORB в мережі клієнтів і серверів на різних комп'ютерах означає, що клієнтська програма (яка може бути об'єктом) може запитувати послуги з серверної програми чи об'єкта, не розуміючи, де сервер знаходиться в розподіленій мережі, або який інтерфейс реалізує серверний обєкт. Для надсилання запитів або повернення відповідей між ORB програми використовують протокол General Inter-ORB (GIOP), а для Інтернету його аналог Inter-ORB (IIOP).

Корпорації Майкрософт має свій аналог CORBA, а саме модель розподілених компонентів (DCOM). Проте CORBA та Microsoft домовились про спосіб інтеграції, при якому клієнтський об'єкт, розроблений відповідно до Component Object Model може зв'язатися з сервером CORBA (і навпаки).

Розподілена обчислювальна обстановка (DCE), архітектура розподіленого програмного забезпечення, яка передувала тенденції до об'єктно-орієнтованого програмування та CORBA, в даний час використовується рядом великих компаній. DCE, можливо, продовжуватиме існувати разом з CORBA.

REST або RESTful Web сервіси це один із способів забезпечення сумісності між комп'ютерними системами в Інтернеті. REST сервіси забезпечують взаємодію на основі текстових моделей реальних обєктів та підтримують чітко визначений протколом HTTP набір операцій.

Архітектурні властивості, на які впливають обмеження архітектурного стилю REST:

- Продуктивність - комунікація між компонентами може мати ключовий вплив на продуктивність сервісу з точки зору користувача.
- Масштабованість - можливість підтримки великої кількості компонентів та взаємодії між компонентами.

Рой Філдінг, один з головних авторів специфікації HTTP, описує вплив REST на масштабованість наступним чином:

Відокремлення клієнта та сервера відповідно до специфікації REST спрощує реалізацію компонентів, зменшує складність семантики коннекторів, підвищує ефективність налаштування продуктивності та підвищує масштабованість чистих серверних компонентів. Системні особливості дозволяють посередникам-проксі-серверам, шлюзам і брандмауерам брати учать в обміні інформацією між компонентами без зміни їх інтерфейсів. REST дозволяє здійснювати проміжну обробку оскільки повідомлення містять весь необхідний опис для їх обробки посередниками.

- Простота уніфікованого інтерфейсу
- Модифікованість компонентів для задоволення змінних потреб (навіть якщо програма працює)
- Видимість зв'язку між компонентами за допомогою службових агентів
- Мобільність компонентів шляхом переміщення програмного коду з даними
- Надійність та стійкість до відмов на рівні системи при наявності збоїв у компонентах, конекторах чи даних.

Корпоративна шина даних (ESB) реалізує систему зв'язку між взаємозалежними програмними додатками в сервіс орієнтованій архітектурі (SOA). Вона реалізує програмну архітектуру, як зображено з правого боку на рис. Оскільки вона реалізує архітектуру програмного забезпечення для розподілених обчислень, вона також реалізує спеціальний варіант більш загальної моделі клієнт-сервер. В цілому будь-який додаток, що використовує ESB, може поводитися як сервер або клієнт. ESB забезпечує логічність та гнучкість комунікації на високому рівні між додатками. Основною метою комунікації на високому рівні є інтеграція корпоративних додатків (EAI).

Рис. 1.25. Ілюстрація принципу роботи корпоративної шини даних

1.4. Висновки до першого розділу

У першому розділі дисертації проведено аналіз впливу системної архітектури розподілених гетерогенних інформаційно-телекомунікаційних систем на значення їх метрик якості. Проаналізовано цілі розроблення архітектур, здійснено порівняння архітектурних шаблонів інформаційно-комунікаційних систем. Встановлено, що архітектура системи ключовим чином впливає на значення метрик якості надання послуг. Кожна з архітектур характеризується окремим набором метрик, що дає змогу найбільш точно оцінити процес надання послуг.

Проаналізовано процеси комунікації у розподілених гетерогенних інформаційно-комунікаційних системах з урахуванням рівня застосунків:

- Методи синхронізації часу;
- Особливості комунікації у системах зі спільною пам’яттю;
- Подано приклад комунікаційних інтерфейсів, орієнтованих на передавання даних;

– Проведено аналіз комунікації у розподілених гетерогенних інформаційно-телекомунікаційних систем на основі черг повідомлень;
– Проаналізовано функціонування інтерфейсів передавання повідомлень;
– Проведено аналіз проміжного програмного забезпечення, орієнтованого на обмін повідомленнями.

Проведений аналіз є основою для подальшого виконання завдань розроблення моделей компонентів досліджуваних інформаційно-телекомунікаційних систем та методів надання послуг в таких системах.

РОЗДІЛ 2. МЕТОДИ ПОБУДОВИ ТА МОДЕЛІ ГЕТЕРОГЕННИХ РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМ

2.1. Принцип багаторівневої взаємодії в гетерогенних інформаційно-телекомунікаційних системах

Сьогодні швидкий розвиток інформаційних технологій нерозривно пов'язаний з телекомунікаційними системами, які забезпечують передавання інформації між компонентами розподілених інформаційних послуг. Оскільки частка децентралізованих послуг, що не обмежується єдиною інформаційною системою, суттєво перевищує частку централізованих послуг, роль телекомунікацій має особливе значення. Поряд з клієнтом-серверною та одноранговою архітектурою розподіленої інформаційної та телекомунікаційної систем функціонують хмарні і «туманні» системи обробки, які за своєю природою є дуже неоднорідні [232].

Враховуючи той факт, що поява та впровадження цих технологій стимулюють розвиток, ймовірно, необмеженого набору послуг, потрібен новий підхід до керування ресурсами та потоками даних на телекомунікаційній площині. Як наслідок, необхідно враховувати взаємодію послуг на прикладному рівні, який формує вимоги до взаємодії на рівні телекомунікаційної мережі [35].

Також сьогодні існує суперечність між розвитком інформаційних технологій та телекомунікаційними системами, що полягає у використанні ресурсів та управління потоками даних, які не враховують вимоги до зв'язку між елементами розподілених сервісів на рівні додатків [4].

Таким чином, розробка методів та моделей надання послуг у гетерогенних розподілених ІТС для підвищення якості обслуговування, покращення структурних та функціональних параметрів, а також удосконалення характеристик таких систем для координації взаємодії підсистеми обслуговування та мережевих ресурсів є актуальною науковою проблемою [267, 293].

Макромодель ІТС представлена у вигляді трьох підсистем: підсистема користувача, підсистема мережі та підсистема обслуговування (Рис. 2.1). На межі підсистем реалізується взаємне керування ресурсами [15, 162].

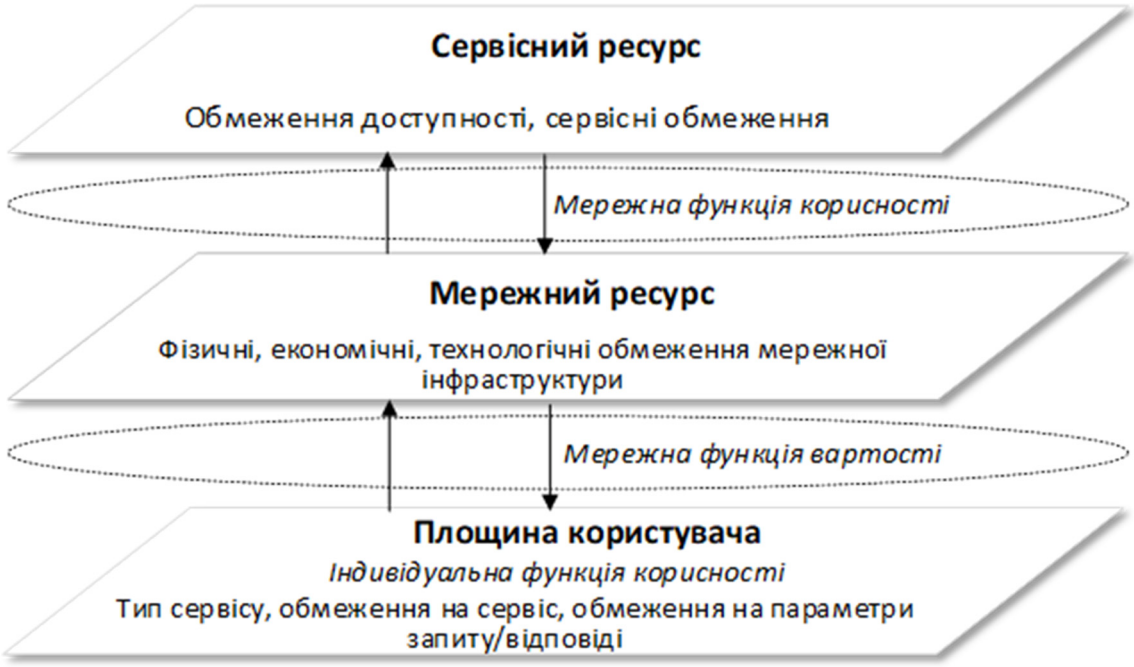


Рис. 2.1. Декомпозиція сервісної архітектури гетерогенної мережі

Сервісний процес складається з двох фаз: етапу координації та фази доставки. На етапі координації здійснюється формування агрегованих запитів на обслуговування від користувачів до певних сервісних ресурсів. Кількість запитів залежить від популярності послуги. Кожен запит характеризується обмеженнями по часу доставки та відповіді, обмеженнями сесії та типом сервісу. Запит посилається мережевою інфраструктурою на інтерфейс для формування відповідної відповіді, яка містить інформацію про параметри створеного сервісу. Нарешті, відповідь повертається користувачеві, який приймає рішення, щоб розпочати сеанс для передачі сервісного потоку.

Визначимо типові сценарії етапу координації запитів.

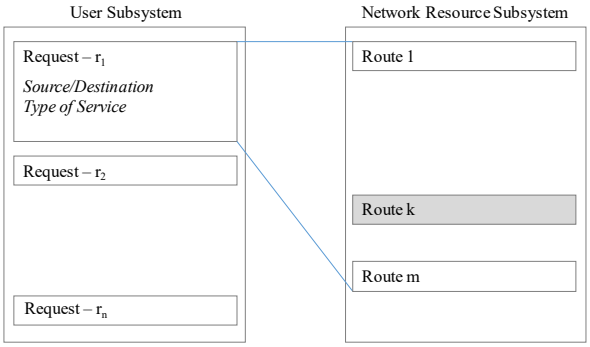


Рис. 2.2. Вибір маршруту запиту

Перша фаза – передавання запиту на інтерфейс сервісу маршрутом, що дозволяє забезпечити всі обмеження запитів / відповідей (маршрутизація за одним критерієм для кінцевої ланки) (Рис. 2.2).

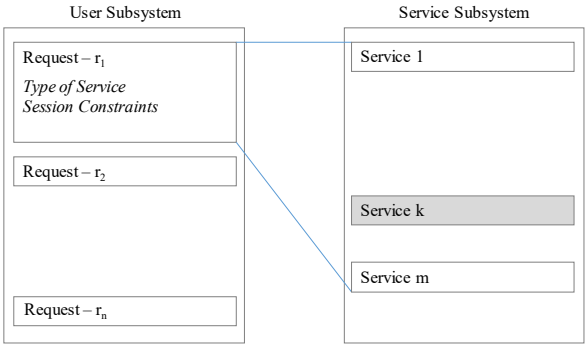


Рис. 2.3. Вибір сервісу

Після доставки запиту на інтерфейс сервісу виконується вибір композитного сервісу (Рис. 2.3). Відповідна компонента повинна тісно відповідати вимогам користувача за типом та обмеженнями сесії, з урахуванням його функціональних можливостей (функціонально одна і та ж послуга може бути інтегрована різними способами).

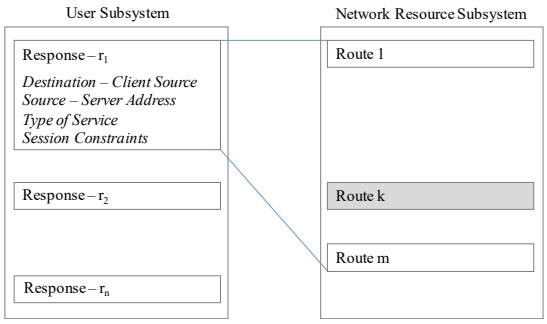


Рис. 2.4. Вибір потоку сервісу

На підставі інформації про відповідь виконується вибір шляху передадавання даних (Рис. 2.4). Набір доступних шляхів у вигляді крайових вузлів (на границі до наступного серверу) підсистеми мережевих ресурсів визначається шляхом постійного відображенн за критеріями джерел та адрес призначення, типом критеріїв обслуговування та обмеженнями сеансів.

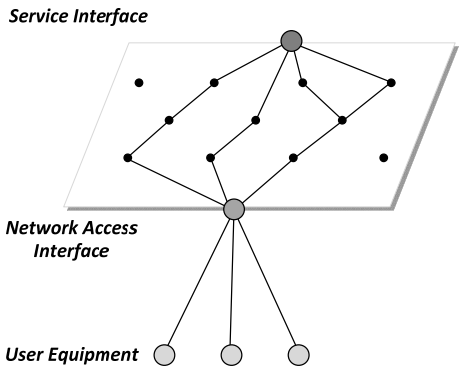


Рис. 2.5. Набір шляхів між NAI та SI

При цьому тип обслуговування визначає допустиму затримку, джиттер, ймовірність втрати пакетів та швидкість передавання даних, пов'язану з цим сервісом, а обмеження на сеанс визначають вимоги до обслуговування в рамках певного типу послуги.

Виходячи з існуючого набору шляхів, підсистема мережних ресурсів вибирає один або декілька шляхів (для цілі балансу навантаження) у якості маршруту для передавання потоку сервісу з урахуванням асимптотичної близькості запиту користувача та параметрів маршруту, щоб забезпечити максимальну корисність мережі та мінімальну її вартість (Рис. 2.5).

Підсистема мережних ресурсів є універсальною площиною доставки контенту. Ми розглядаємо це як багатошаровий граф, конфігурація якого залежить від технології передавання даних (кількість шарів, множина вершин, множина ребер).

У процесі перенаправлення потоку сервісу ми фактично формуємо набір шляхів передавання в підсистемі мережних ресурсів відповідно до вимог певного потоку, заснованого на обчисленні корисності мережі та розрахунку її вартості. Передача здійснюється для задоволення вимог службового потоку, а кількість

неефективно використовуваних мережних ресурсів не повинна перевищувати прийнятного рівня. З огляду на те, що кількість потоків послуг, які одночасно передають дані, як правило, є значною, завдання формування множини шляхів передавання не може бути розв'язане для кожного потоку, і, таким чином, воно стає завданням розподілу мережних ресурсів залежно від їх вартості та може бути записане як завдання нелінійного цілочисельного програмування або завдання нелінійного змішаного програмування.

Враховуючи специфіку сучасних мережних технологій, підсистема мережних ресурсів вимагає вертикальної та горизонтальної декомпозиції. Вертикальна декомпозиція забезпечує незалежний розподіл функцій між шарами багатошарового графа, в той час як горизонтальна декомпозиція забезпечує реалізацію однакових функцій в окремому шарі для багатошарових вузлів графа у вигляді розподілених обчислень [71].

На етапі координації запиту структура мережі розглядається як граф на площині, а на етапі надання послуг вона розглядається як багатошаровий граф.

2.1.1. Математична модель передавання запитів на послуги в розподілених інформаційно-комунікаційних системах

Припустимо, що існують шляхи K^s від інтерфейсів доступу до мережі (NAI) до сервісного інтерфейсу (SI), представлені у матриці $\mathbf{H}^s$ розмірністю $L \times K^s$, де

$$h_{lj}^s = \begin{cases} 1, \text{ if path } j \text{ to SI } s \text{ uses link } l; \\ 0, \text{ otherwise.} \end{cases}, \tag{2.1}$$

L – кількість каналів у мережевій інфраструктурі.

Нехай H_H^s - сукупність всіх стовпчиків матриці, яка представляє всі доступні шляхи до s.

Нехай $\mathbf{w}^s$ містить вектор $K^s \times 1$, в якому j-й компонент представляє собою частку потоку сервісу до s-го SI, який передається по j-ому шляху:

$$w_j^s \geq 0 \ \forall j \quad \text{і} \quad \mathbf{1}^T \mathbf{w}^s = 1, \tag{2.2}$$

де **1** – вектор відповідного розміру, всі елементи якого задані одиницями. Це необхідно для одношляхової маршрутизації

$$w_j^s = \begin{cases} 1, \text{ all data to } s\text{-th SI by } j\text{-th path,} \\ 0, \text{ otherwise.} \end{cases} \tag{2.3}$$

і для багатошляхової маршрутизації:

$$w_j^s \in [0,1], \tag{2.4}$$

Таким чином, $\mathbf{H}^s$ визначає множину шляхів без контурів до s-го SI, і $\mathbf{w}^s$ визначає частку навантаження на кожному шляху до s-го SI. Складові $\mathbf{H}^s$ і $\mathbf{w}^s$ визначають $L\times 1$ вектор маршрутизації $\mathbf{r}^s = \mathbf{H}^s\mathbf{w}^s$, так як ці елементи представляють частку s-го потоку в кожній ланці l.

Матриця $\mathbf{R}$ з розмірами $L\times S$ може бути представлена як:

$$\mathbf{R} = [\mathbf{r}^1 \ldots \mathbf{r}^S], \tag{2.5}$$

і є матрицею маршрутизації для всіх SI.

Різниця між одношляховими та багатошляхавими маршрутами полягає в обмеженнях елементів матриць.

Тому, в матриці для одношляхової маршрутизації:

$$r_{ls} = \begin{cases} 1, \text{ if } l \text{ belongs to path to } s; \\ 0, \text{ otherwise.} \end{cases} \tag{2.6}$$

для багатошляхової:

$$r_{ls} = \begin{cases} \neq 0, \text{ if } l \text{ belongs to path to } s,\ r_{ls} < 1; \\ = 0, \text{ otherwise.} \end{cases} \tag{2.7}$$

Шлях до SI s визначається як:

$$p^s = (r^s)^T \tag{2.8}$$

2.1.2. Декомпозиція завдання розподілу ресурсів у процесі надання множини інформаційних послуг групам користувачів

Сучасні інформаційно-комунікаційні системи вважаються гетерогенними з точки зору географічного розподілу та технологічної складності. Також такі системи створюють проблему багатокомпонентного потоку через різні вимоги

користувачів до надання послуг (різні об'єктивні параметри - доступність, пропускна спроможність, затримка тощо) та велику кількість доступних та перспективних послуг [44].

Важливим завданням у цьому випадку є розподіл загальної ємності системи між потоками послуг. Загалом, цей виклик призводить до завдання нелінійної оптимізації через специфічні властивості трафіку в сучасних інформаційно-комунікаційних системах.

Багато вчених досліджують проблеми оптимізації таких систем. Деякі з них використовували функцію корисності як засіб розподілу пропускної здатності. Їх модель продуктивності мереж заснована на агрегованих функціях корисності з метою виділення системних ресурсів. Більшість з цих досліджень розглядають гомогенні системи, де кожен потік послуг орієнтований на той же об'єктивний параметр (наприклад, пропускну здатність) [87, 88].

Запропоновано інший підхід, що базується на перетворенні об'єктивного параметра для кожного потоку сервісу на відповідне значення на основі функції корисності. Це дозволяє агрегувати попит різних користувачів із однаковими вимогами QoS (Quality of Service).

Для зменшення складності завдання нелінійної оптимізації для розподілу пропускної здатності мережі між потоками сервісів пропонуємо багатошарову декомпозицію та координацію розв’язків на основі принципу оптимальності Беллмана [16].

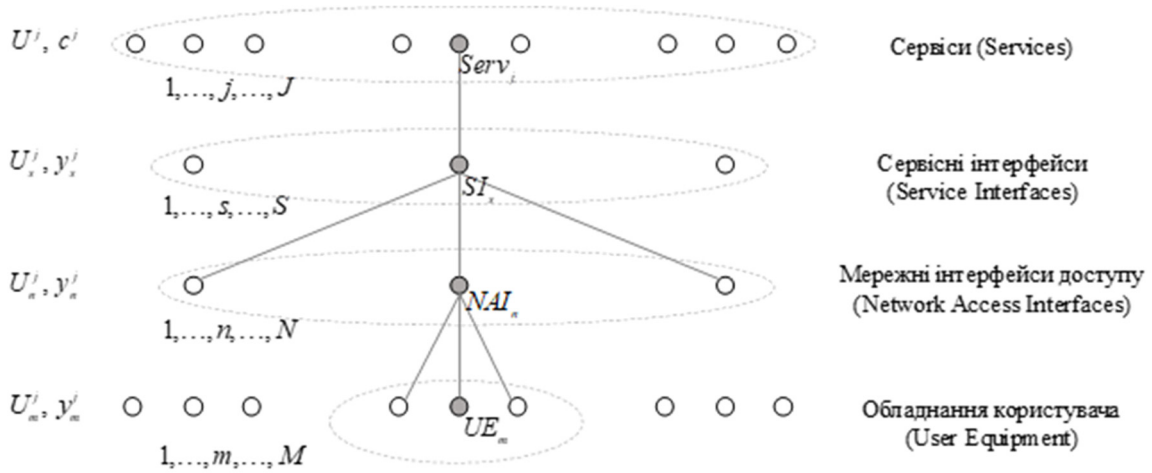


Рис. 2.6. Декомпозиція гетерогенної інформаційно-комунікаційної системи

Нехай користувач m подає заявку на сервіс j і має функцію корисності $U_m^j(y_m^j)$, де y_m^j – пропускна здатність для надання послуги користувачеві. Перевага використання функцій корисності – це можливість визначати загальний показник у довільних одиницях (утилях) для сервісів, надання яких залежать від різних мережних ресурсів [123].

Правильна інтерпретація функцій корисності від користувачів є складним завданням. Необхідно знати, яка із них має найбільше значення для правильного застосування пріоритетів. При цьому ж, може вирішуватися завдання пріоритезації користувачів.

Всі користувачі повинні періодично повідомляти про значення своїх функцій корисності центральному контроллеру. Після цього він повинен вирішити завдання складної нелінійної оптимізації, щоб визначити, яку пропускну здатність можна виділити для передавання потоку послуг (завдання резервування).

Вирішення завдання оптимізації є доволі складним завданням, враховуючи потенційну кількість користувачів та географічний розподіл мережних об'єктів. Щоб спростити завдання нелінійної оптимізації, на кожному модельному рівні ми ініціюємо агрегування відповідних функцій корисності. Таким чином, функції корисності є вхідними даними у завданні розподілу ємності мережі [292].

Наведемо етапи формулювання завдання.

Функція корисності $U_m^j(y_m^j)$ від M_n^j користувачів, де $M_n = \sum_j M_n^j$, при надсиланні сервісного потоку j, агрегується до єдиного інтерфейсу доступу до мережі n (NAI) із загальною функцією корисності $U_n^j(y_n^j)$.

Загальна функція корисності $U_n^j(y_n^j)$ є сумарним значенням загальних функцій корисності $U_s^j(y_s^j)$, що відображає значення корисності $M_s^j = \sum_{j,n} M_n^j$ для всіх користувачів сервісу j, які потребують доступу до s-ого SI.

Крім того, ми утворюємо функцію корисності U^j, тобто суму функцій корисності $U_s^j(y_s^j)$, яка є спільною для всіх користувачів сервісу j, $M^j = \sum_{j,s} M_s^j$ чи $M^j = \sum_s \sum_n \sum_m m_{n,s}^j$

Завдання розподілу ємності мережі зводиться до визначення набору значень ємності c^j на основі агрегованих функцій корисності U^j кожного сервісного потоку.

У той же час можливі наступні випадки:

$$c^j = \sum_s \sum_n \sum_m y_{m,n,s}^j , \tag{2.9}$$

де $y_{m,n,s}^j$ - ресурс, необхідний користувачеві m, який підключений до NAI, і шукає доступ до SI s для надання послуги j.

Коли умова (2.9) виконується, ми знаходимо c^j, яка задовольняє потреби користувачів, тому ця послуга може бути надана користувачам із гарантованою якістю.

$$c^j < \sum_s \sum_n \sum_m y_{m,n,s}^j ; . \tag{2.10}$$

Якщо виконується умова (2.10), ми повинні визначити, які запити користувачів на сервіс j повинні обслуговуватися, в умовах обмеженості пропускної здатності c^j.

Розглянемо підсистему мережевих ресурсів з N вузлів та L спрямованих каналів, де кожен канал має пропускну здатність c^l. Є сервісні потоки J (послуги); кожен з них характеризується трьома параметрами (відправним вузлом, цільовим вузлом та класом трафіку). У цьому випадку клас трафіку визначає максимальну затримку трафіку, джитер та максимальну межу швидкості передавання для відповідного сервісного потоку. Відповідно до необхідних параметрів QoS та класу трафіку, існує множина шляхів передавання, які представляються у вигляді матриці $\mathbf{P}^j$ для кожного сервісного потоку $j \in [1, J]$, що використовується для розподілу навантаження сеансів користувача. $\mathbf{P}^j$

складається з підмножини шляхів $\mathbf{P}_s^j$ до кожного сервісного інтерфейсу s. Нехай $\{P_{p^j}\}$ – це сукупність всіх стовпчців матриці $\mathbf{P}^j$.

Для кожного сервісу обчислюється матриця $L \times K$. Її елементи визначають канали, що відносяться до шляху:

$$p_{k,l}^j = \begin{cases} 1, \text{ if } l \text{ belong to the } k\text{-th path for } j\text{-th service;} \\ 0, \text{ otherwise.} \end{cases}, \tag{2.11}$$

де l – один з каналів .

Опишемо функцію розподілу ємності мережі між сервісними потоками наступним чином:

$$\max_{x_k^j} \sum_{j=1}^{J} U^j \left(\sum_{k=1}^{\left|\{P_{p^j}\}\right|} x_k^j \right), \tag{2.12}$$

з урахуванням обмежень:

$$\sum_{j=1}^{J} \sum_{k=1}^{\left|\{P_{p^j}\}\right|} p_{k,l}^j x_k^j \le c^l, \; l = 1, \ldots, L. \tag{2.13}$$

$$\sum_{k=1}^{\left|\{P_{p^j}\}\right|} x_k^j \le \sum_{s} \sum_{n} \sum_{m} y_{m,n,s}^j, \; j = 1, \ldots, J. \tag{2.14}$$

$$x_k^j \ge 0, \; k = 1, \ldots, \left|\{P_{p^j}\}\right|, \; j = 1, \ldots, J. \tag{2.15}$$

У (2.14) потік y^j замінюється сумою вкладених потоків $y_{m,n,s}^j$ для кожного шляху k.

Нерівність (2.13) встановлює умову, що потік в кожному каналі не перевищує його пропускної здатності c^l.

Передавання здійснюється для задоволення вимог сервісного потоку, а кількість неефективно використовуваних мережних ресурсів не повинна перевищувати прийнятного рівня.

Нерівність (2.14) вводить обмеження при багатошляховій маршрутизації: частка потоку послуг по шляху не може перевищувати пропускну здатність. Різниця між y^j та сумою розрахункових потоків $\sum_{k=1}^{\left|\{P_{p^j}\}\right|} x_k^j$ відображає частку втраченого трафіку.

Таким чином, ми завжди отримуємо прийнятний розв'язок, навіть якщо запитана пропускна здатність перевищує пропускну здатність мережі.

Пропускна здатність мережі розраховується як:

$$c^j = \sum_{k=1}^{\left|\{P_{pj}\}\right|} x_k^j , \tag{2.16}$$

що є оптимальним розв'язком завдання розподілу ємності мережі між сервісними потоками за умови, що функції функції корисності U^j відомі.

2.1.3. Моделі користування інформаційно-комунікаційними послугами на основі сімейства функцій корисності

Нижче наведено кілька прикладів окремих функцій корисності [10], які використовують для розв'язку завдань відображення параметрів передавання інформаційних потоків [221, 270].

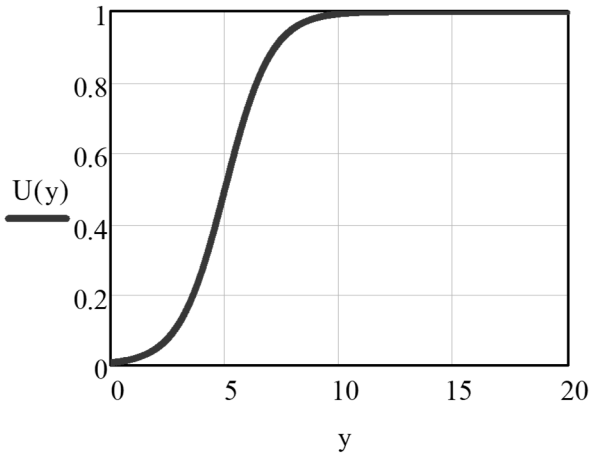


Рис. 2.7. Логнормальна функція корисності

Цей тип функції може бути використаний для визначення корисності сервісу через швидке збільшення корисності, яке близьке до бажаного значення. Логарифмічна функція корисності підходить для оцінки пропускної здатності [241, 242].

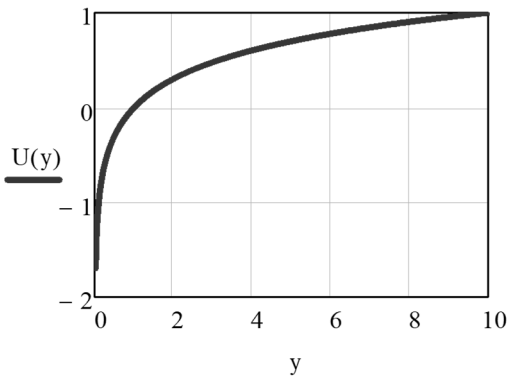


Рис. 2.8. Логарифмічна функція корисності

Коли ви отримуєте більшу пропускну здатність, ніж запитана, корисність цього додаткового значення буде більшою, однак різниця між двома наступними корисностями буде меншою кожного разу.

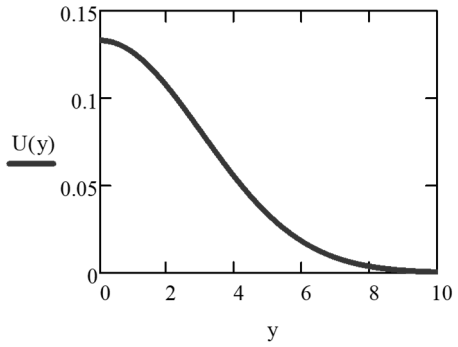


Рис. 2.9. Гаусівська функція корисності

Функція корисності Гауса підходить для випадку виявлення компромісу між потоком послуг, що не використовуються, або буферизацією протягом відносного довгого часу. Цим способом можна визначити корисність тарифів на послуги.

На кожному рівні запропонованої декомпозиції (Рис. 2.1) ми пропонуємо агрегувати основні значення функцій корисності, щоб зменшити складність розв’язку завдання. Тому нам потрібно знайти механізм агрегації, навіть якщо ми маємо справу з потоками від різних сервісів з різними вимогами до процесу доставки. Сукупні значення зазначеної сервісної корисності – це вхідні дані для завдання розподілу ємності серед потоків послуг.

У випадку індивідуальних функцій корисності, наведених у роботі, розглядаємо випадок їх лінійності і єдиний цільовий параметр – пропускну

здатність. У випадку багатошляхової маршрутизації, потоки послуг орієнтовані на різні цільові параметри. Тому ми можемо підсумувати не функції, а лише їхні значення у відповідні часові моменти. Значення функції корисності вимірюються в утилях. Тому ми можемо використовувати ці значення незалежно від об'єктивного параметра (доступність, пропускна здатність або затримка). Це дозволяє нам розподіляти ємність системи відповідно до максимізації якості сприйняття послуг.

2.2. Територіально-залежна модель балансування навантаження, створеного у процесі надання інформаційно-комунікаційних послуг з урахуванням переміщення користувачів

Бурхливі процеси конвергенції та інтеграції на системному рівні, які характерні для інформаційно-телекомунікаційних систем, спричинили появу спільної технологічної основи для обслуговування користувачів з диференційованими вимогами до якості надання послуг. Такі системи дають змогу адаптації нових видів послуг із дотриманням спільних принципів обслуговування для всіх типів користувачів. З урахуванням того, що нові інформаційно-телекомунікаційні послуги потребують значної пропускної здатності системи, виникає завдання забезпечення необхідного телекомунікційного ресурсу у процесі обслуговування [273], особливо в періоди стрибкоподібного зростання попиту на послуги в окремих територіально-залежних сегментах мережі [47, 50, 58, 245, 280].

Сучасні методи та засоби керування інформаційно-телекомунікаційними системами [26, 117, 135, 136] реалізовані на базі розподілу зон відповідальності за прийняття рішень, тобто розподіл ресурсів має локальний характер та не враховує завантаженості всієї територіально-залежної системи [74, 75, 76]. Для їх вивчення слід провести територіально-залежне моделювання структури інформаційно-телекомунікаційної системи [42, 48, 49], змоделювати поведінку користувачів інформаційно-телекомунікаційних послуг, дослідити розподіл користувацього навантаження в системі, а також запропонувати метод його балансування, враховуючи інформацію про завантаження всієї системи.

Завдання балансування користувацького навантаження у інформаційно-телекомунікаційних системах представляє інтерес для широкого кола науковців. Дослідження роботи таких систем можна згрупувати у дві великі групи: балансування навантаження на основі імовірності відмови в обслуговуванні запитів та з урахуванням ступеня завантаженості інформаційно-телекомунікаційної системи [72, 73, 77]. Перша група характеризується меншим обсягом службової інформації, адже процес балансування ініціюється перевищенням порогу імовірності відкидання запитів на обслуговування. Тобто, якщо поріг досягнутий, балансування навантаження між пунктами надання послуг здійснюється шляхом зміни радіусу дії перевантаженого і сусідніх пунктів надання послуг через, наприклад, регулювання потужності випромінювання базових станцій у випадку безпровідної інофрмаційно-телекомунікаційної інфраструктури [226].

Друга група методів балансування є кращою з точки зору практичності, бо дає змогу врахувати міру використання пропускної здатності і балансування навантаження як у процесі вибору пункту надання послуг, так і при здійсненні перемикання на інший пункт надання послуг. Процес балансування слід розпочинати з максимально завантаженого пункту надання послуг з метою досягнення рівномірного завантаження інформаційно-телекомунікаційної системи в цілому [181].

Однак, сьогодні немає методів та засобів, які дають змогу примати рішення про балансування навантаження на підставі інформації про поточне завантаження всієї інформаційно-телекомунікаційної системи, бо для цього слід забезпечити централізований моніторинг стану мережі та перемикання користувачів між пунктами надання послуг [90, 174]. Тобто, розроблення моделей та методів балансування користувацького навантаженя в інформаційно-телекомунікаційних системах на підставі їх повної територіальної структури за рахунок централізованого керування вибором пунктів надання послуг є актуальним. Для того, щоб розв’язати це завдання, слід урахувати територіальну структуру інформаційно-телекомунікаційної системи, характеристику

активності та переміщення користувачів, що забезпечується на підставі запропонованої імітаційної моделі поведінки користувачів інформаційно-телекомунікаційної системи [175].

Перед тим, як розпочати моделювання процесу балансування між пунктами надання послуг, спочатку слід розв'язати завдання розроблення територіально-залежної моделі інформаційно-телекомунікаційної системи, першим етапом роботи якої є формування множини координат пунктів надання послуг (ПНП) [126]. Нехай вважатимемо територію рівнинною та прямокутною областю, де ПНП розміщені рівномірно, а зони обслуговування представлені шестикутниками (Рис. 2.10).

З Рис. 2.10 бачимо, що представлений спосіб розбиття території на зони обслуговування характеризується періодом $3 \cdot R$ по горизонталі. Відстань між сусідніми ПНП, що розташовані на одній горизонталі, становить $3 \cdot R$. Кількість повних періодів по горизонталі n_x становитиме:

$$n_x = \left\lfloor \frac{X}{3 \cdot R} \right\rfloor, \tag{2.17}$$

де X – довжина території покриття; R – радіус комірки.

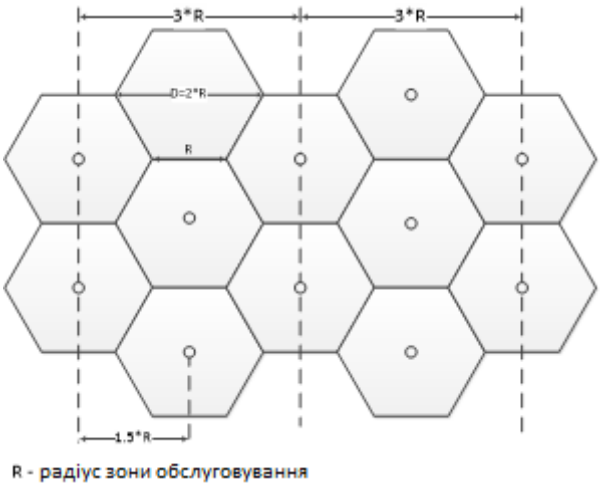


Рис. 2.10. Фрагмент територіальної структури інформаційно-телекомунікаційної системи

$$z_x = \frac{X}{3 \cdot R} - n_x, \tag{2.18}$$

де z_x – різниця між довжиною території та довжиною періодичної структури по горизонталі.

У випадку, коли $X > R$, то кількість стовпців зон обслуговування та координата x ПНП першого стовпця визначаються як

$$n_c = \begin{cases} 2 \cdot n_x, & z_x \le \frac{1}{3} \\ 2 \cdot n_x + 1, & \frac{1}{3} < z_x \le \frac{5}{6} \\ 2 \cdot n_x + 2, & z_x > \frac{5}{6} \end{cases}, \tag{2.19}$$

де n_c – загальна кількість стовпців зон обслуговування.

$$X_1 = \begin{cases} R \cdot \left(1 + 1.5 \cdot \left(z_x - \frac{1}{6}\right)\right), & z_x \le \frac{1}{3} \\ R \cdot 1.5 \cdot z_x, & \frac{1}{3} < z_x \le \frac{5}{6} \\ R \cdot \left(1 + 1.5 \cdot \left(zz_x - \frac{7}{6}\right)\right), & z_x > \frac{5}{6} \end{cases}, \tag{2.20}$$

де X_1 – координата x ПНП першого стовпця.

Якщо $X \le R$, то справедливі такі співвідношення

$$n_c = 1 . \tag{2.21}$$

$$X_1 = 1.5 \cdot R \cdot z_x = \frac{X}{2} . \tag{2.22}$$

Для ПНП кожного наступного стовпця зон обслуговування координата x визначається як

$$X_n = X_0 + 1.5 \cdot R \cdot (n-1), \qquad n = \overrightarrow{2, n_c} , \tag{2.23}$$

де X_n – координата x ПНП n-го стовпця зон обслуговування.

Рис. 2.11 представляє геометричні відстані між сусідніми ПНП шестикутної територіальної структури, що знаходяться на різних горизонтальних осях зон обслуговування. Ця інформація використовується для обчислення вертикальних координат ПНП у процесі формування територіальної структури системи на заданому периметрі.

З Рис. 2.11 видно, що кількість ПНП у парних та непарних стовпцях зон обслуговування відрізняється на одиницю.

$$n_y = \left\lceil \frac{Y}{R \cdot \sqrt{3}} \right\rceil, \tag{2.24}$$

де Y – ширина території розгортання системи; n_y – кількість ПНП у парних стовпцях зон обслуговування ($n_y - 1$ – кількість ПНП у непарних стовпцях зон обслуговування).

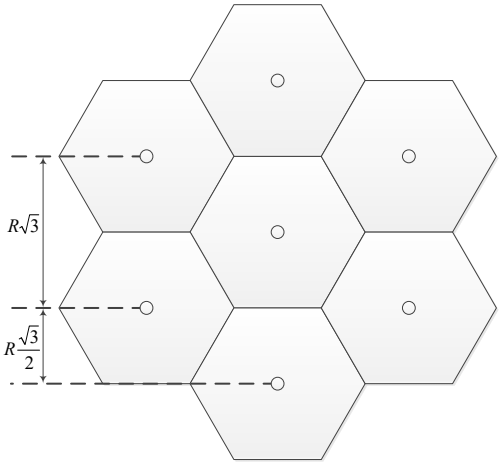


Рис. 2.11. Ілюстрація відстаней між сусідніми ПНП на різних горизонтальних осях зон обслуговування

$$z_y = n_y - \frac{Y}{R \cdot \sqrt{3}}, \tag{2.25}$$

де z_y – різниця між шириною території та шириною періодичної структури по вертикалі.

Стартова координата y ПНП у парних стовпцях зон обслуговування:

$$Y_{1П} = R \cdot \frac{\sqrt{3}}{2} \cdot \left(1 - z_y\right). \tag{2.26}$$

Стартова координата y ПНП у непарних стовпцях зон обслуговування:

$$Y_{1Н} = R \cdot \frac{\sqrt{3}}{2} \cdot \left(2 - z_y\right). \tag{2.27}$$

Координата y для решти ПНП розраховується як:

– для парних стовпців зон обслуговування

$$Y_{nП} = R \cdot \sqrt{3} \cdot \left(n - \frac{z_y - 1}{2} \right), \qquad n = \overrightarrow{2, n_y}, \tag{2.28}$$

– для непарних стовпців зон обслуговування

$$Y_{nH} = R \cdot \sqrt{3} \cdot \left(n - \frac{z_y}{2} \right), \quad n = \overrightarrow{2, \left(n_y - 1 \right)}, \tag{2.29}$$

де $Y_{nП}, Y_{nH}$ – координати y n-х ПНП парного ($_{nП}$) та непарного ($_{nH}$) стовпців зон обслуговування.

Число ПНП становить:

$$N_{БС} = n_x \cdot n_y - \left\lfloor \frac{n_x}{2} \right\rfloor. \tag{2.30}$$

Результати, представлені на Рис. 2.12, демонструють сформовану територіальну структуру розподіленої гетерогенної інформаційно-телекомунікаційної системи з такими параметрами:

– довжина території розгортання – 1000 м;
– ширина території розгортання – 1000 м;
– радіус зони обслуговування – 100 м.

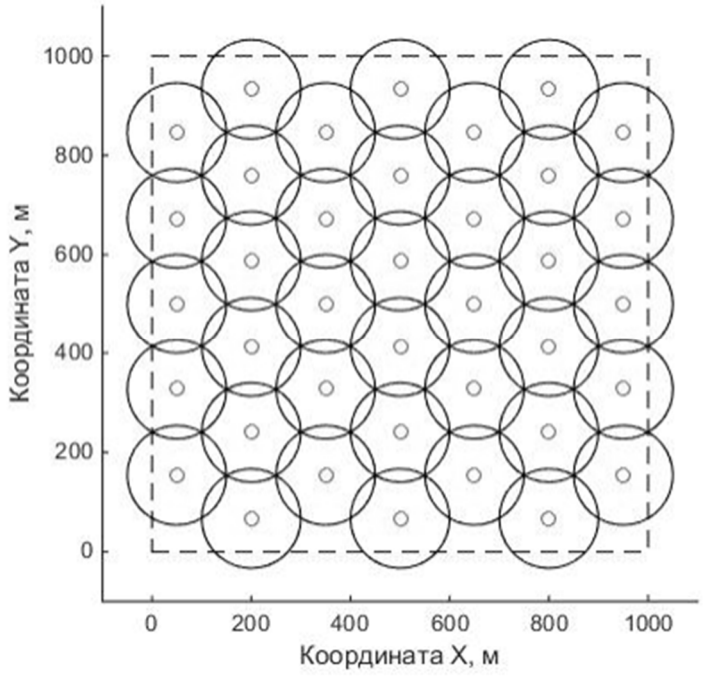


Рис. 2.12. Приклад територіальної структури інформаційно-телекомунікаційної системи

Бачимо, що на Рис. 2.12 помітні елементи території без обслуговування на границях периметру, що є особливістю запропонованого підходу. Приймемо, що ця такі фрагменти обслуговуються ПНП із сусідніх територіальних елементів інформаційно-телекомунікаційної системи.

2.2.1. Вплив переміщення користувачів на процес надання послуг

У моделі переміщення користувачів враховано довжину переміщення r (м) та кут переміщення ϕ (радіан). Територіальне переміщення користувача ІТС залежить від швидкості його руху $V_{аб}$ (м/с) [21]

$$V_{аб} = V_{\max} \cdot k \,, \tag{2.31}$$

де $V_{\max}$ – максимальна передбачена моделлю швидкість переміщення (у моделі прийнято, що $V_{\max} = 33 \ м/с$); k – коефіцієнт, що є добутком K рівномірно розподілених випадкових чисел на проміжку від 0 до 1.

Таким чином, середнє значення розрахованих швидкостей

$$V_{аб\_сер} = \frac{V_{\max}}{2^K}. \tag{2.32}$$

Варіація напряму переміщення користувача визначається кутом переміщення [113]:

$$\phi = \alpha + \beta \cdot (0.5 - b), \tag{2.33}$$

де α –складова-константа кута переміщення, радіан; β – величина, що визначає діапазон допустимої зміни кута переміщення, радіан; b – випадкове число на проміжку від 0 до 1, що підкоряється рівномірному закону розподілу.

$$\alpha = \pi \cdot (1 - 2 \cdot d), \tag{2.34}$$

де d –випадкове число на проміжку від 0 до 1, що підкоряється рівномірному закону розподілу.

$$\beta = 2 \cdot \pi \cdot g \cdot \phi_V \,, \tag{2.35}$$

де g –випадкове число на проміжку від 0 до 1, що підкоряється рівномірному закону розподілу; ϕ_V – коефіцієнт, який визначає ступінь впливу швидкості руху на зміну напряму переміщення, значення ϕ_V знаходиться на інтервалі від 0 до 1. Зі зростанням значення $V_{аб}$, зменшується ϕ_V і, таким чином, кута переміщення варіюється у меншому діапазоні.

$$\phi_V = \left(1 - \frac{V_{ab}}{V_{\max}}\right)^m, \tag{2.36}$$

де m – показник ступеня, зростання якого забезпечує зменшення варіації напряму переміщення.

Переміщення користувача r розраховується зі співвідношення

$$r = \Delta t \cdot V_{ab} \cdot \left((1-p) + 2 \cdot p \cdot a\right), \tag{2.37}$$

де Δt – інтервал часу, що відповідає крокові моделювання (с); p – коефіцієнт, який характеризує максимальну варіацію швидкості руху користувача на інтервалі Δt (у моделі прийнято $p = 0{,}1$, тобто максимальна варіація швидкості абонента становить ±10% на інтервалі моделювання), a –випадкове число на проміжку від 0 до 1, що підкоряється рівномірному закону розподілу.

Стартові координати користувача X_0, Y_0 визначаються випадково із використанням рівномірного закону розподілу в межах території розташування інформаційно-телекомунікаційної системи.

Подальші точки розташування абонента розраховуються у процесі моделювання із значень r та ϕ :

$$X_n = X_{n-1} + r \cdot \cos(\phi). \tag{2.38}$$

$$Y_n = Y_{n-1} + r \cdot \sin(\phi). \tag{2.39}$$

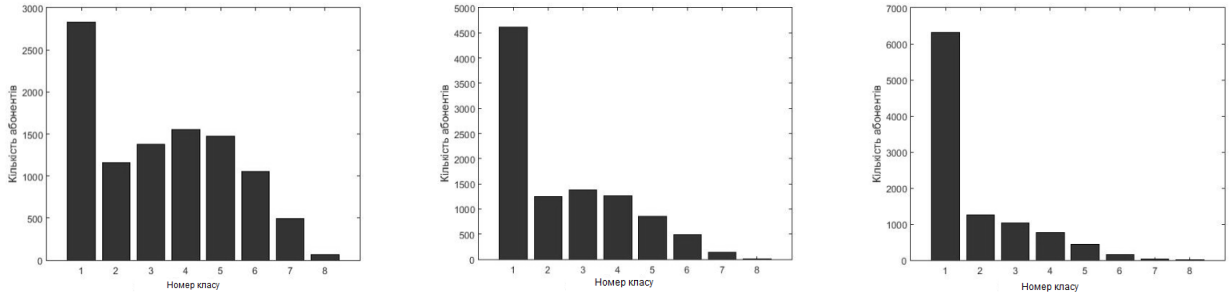
Значення $V_{аб}$, $V_{\max}$, k, α, β та ϕ_V розраховуємо на старті моделювання для кожного абонента, а r, ϕ, X_n та Y_n розраховуємо на кожному кроці моделювання.

Користувачі системи за швидкістю переміщення розділені на класи (Таблиця 2.1).

Таблиця 2.1. Класифікація користувачів за їх швидкістю переміщення

Номер класу	Швидкість переміщення користувача, м/с
1	$V_{аб} < 0.25$
2	$0.25 < V_{аб} \le 0.5$
3	$0.5 < V_{аб} \le 1$
4	$1 < V_{аб} \le 2$
5	$2 < V_{аб} \le 4$
6	$4 < V_{аб} \le 8$
7	$8 < V_{аб} \le 16$
8	$V_{аб} > 16$

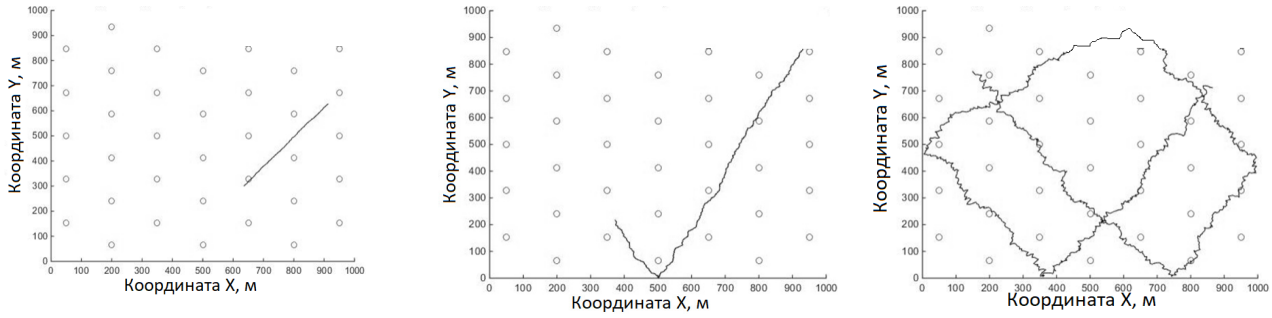
Рис. 2.13 представляє приклади розподілу користувачів ІТС за їх швидкістю переміщення. Вони характеризують особливості поведінки користувачів в різні періоди доби. Максимальне число користувачів завжди належить першому класу, що відповідає майже нерухомому положенню з незначними переміщеннями. У періоди доби з 8:00 до 10:00 і з 16:00 до 19:00 суттєвою є частка користувачів класів з 2 до 6, що характеризує їх переміщення між місцями постійного перебування [152] (Рис. 2.13, а). Протягом решти періодів, за вийнятком нічного часу, розподіл користувачів може бути охарактеризований Рис. 2.13, б. У цей час характерним є зменшення інтенсивності переміщення користувачів (Рис. 2.13, в).



а б в

Рис. 2.13. Типові розподіли користувачів за швидкістю переміщення для різних класів користувачів K: а) $K=4$, б) $K=5$, в) $K=6$.

Приклади траєкторії переміщення користувачів на основі представленої моделі подано на Рис. 2.14. У користувачів із більшою швидкістю руху спостерігається довша траєкторія переміщення.



а б в

Рис. 2.14. Траєкторії переміщення користувачів на території надання послуг: а) для абонента класу 2; б) для абонента класу 4; в) для абонента класу 6

2.2.2. Показник активності користувацьких сесій

Модель активності користувацьких сесій базується на імітаційному моделюванні процесу обслуговування запитів на надання послуг [105, 185-187]. Кожен користувацький сеанс у моделі характеризується масивом тривалостей ($Length$) комунікаційних сесій на основі розподілу Пуассона та інтервалів ($Interval$) між ними. Таким чином, моменти початку ($Start$) та завершення (End) n-ого сеансу можуть бути розраховані за такими співвідношеннями

$$\begin{aligned} Start_i &= End_{n-1} + Interval_n, \qquad n \in [2; N] \\ End_i &= Start_n + Length_n, \qquad n \in [1; N] \end{aligned}, \tag{2.40}$$

де n – номер комунікаційної сесії; N – максимальна кількість сесій користувача на інтервалі моделювання; $Start_1 = Interval_1$.

На кожному інтервалі моделювання i визначаємо такі параметри:

- кількість запитів на надання послуг – $Arrival\_density_i$;
- кількість сесій, які успішно розпочалися – $Start\_density_i$;
- кількість сесій, що завершилися – $End\_density_i$;
- кількість сесій, під час яких відбулася відмова в обслуговуванні – $Loss\_density_i$;
- кількість сесій, що перебувають в активному стані – $Load_i$.

Число запитів на надання послуг прирівнюється до кількості сесій, що мають початися в $_i$ -ий інтервал моделювання:

$$Start_{nk}^{k} \equiv i \Rightarrow Arrival\_density_i = Arrival\_density_i + 1, \quad (2.41)$$

де $Start_{nk}^{k}$ – момент початку наступної комунікаційної сесії користувача k; nk – порядковий номер цієї сесії.

Число завершених сесій в $_i$-ий інтервал моделювання прирівнюється до кількості сесій, моменти завершення яких співпадають з $_i$:

$$End_{nk}^{k} \equiv i \Rightarrow \begin{cases} End\_density_i = End\_density_i + 1 \\ nk = nk + 1 \\ Load_i = Load_i - 1 \end{cases}, \quad (2.42)$$

де End_{nk}^{k} – момент завершення наступної комунікаційнї сесії користувача k.

Число комунікаційних сесій, які розпочалися в $_i$-ий інтервал моделювання, відповідає кількості сесій, моменти початку яких співпадають з $_i$, але не перевищують максимально можливу кількість активних сесій Max_Load:

$$Start_{nk}^{k} \equiv i \Rightarrow \begin{cases} Load_i < Max\_Load \Rightarrow \begin{cases} Start\_density_i = Start\_density_i + 1 \\ Load_i = Load_i + 1 \end{cases} \\ Load_i \equiv Max\_Load \Rightarrow \begin{cases} End_{nk}^{k} = i - 1 \\ nk = nk + 1 \\ Loss_i = Loss_i + 1 \end{cases} \end{cases}. \quad (2.43)$$

Для кроку моделювання $i = 1$ значення параметру $Load_i$ приймається рівним 0, а $nk = 1$. Для всіх наступний кроків $_i$ початкове значення параметру $Load_i$ приймається таким, що відповідає значенню $Load_{i-1}$, яке обчислене на попередньому кроці.

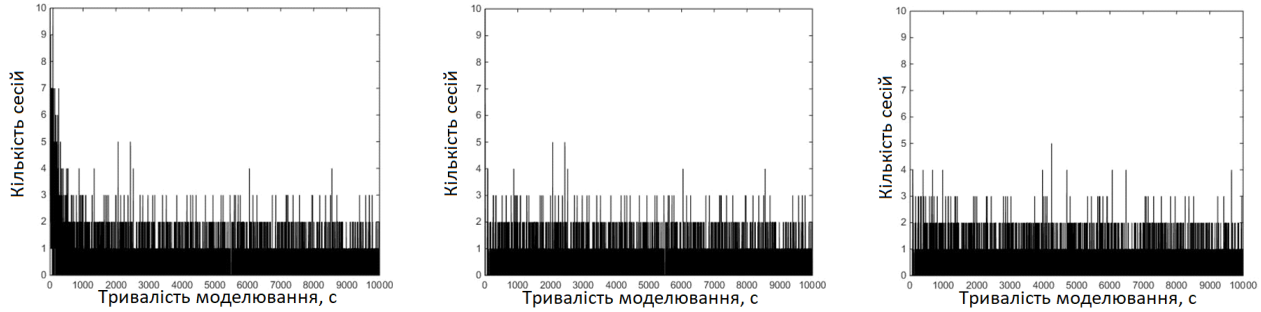
Вхідні параметрами моделювання мережної активності абонентів такі [21]:

- середнє значення інтервалу між сеансами, $\lambda = 200$ с;
- максимальна кількість сеансів для кожного абонента, $N = 50$.
- середнє значення тривалості викликів, $T = 81$ с;
- максимальна кількість викликів $N = 50$;
- максимальна кількість одночасних комунікаційних сеансів в системі $Max\_Load = 40$;

– тривалість інтервалу моделювання $Time\_life = 10000\,c$;

– загальна кількість користувачів $N\_user = 1000$.

Результати, отримані за підсумками моделювання активності абонентів, подано на Рис. 2.15.

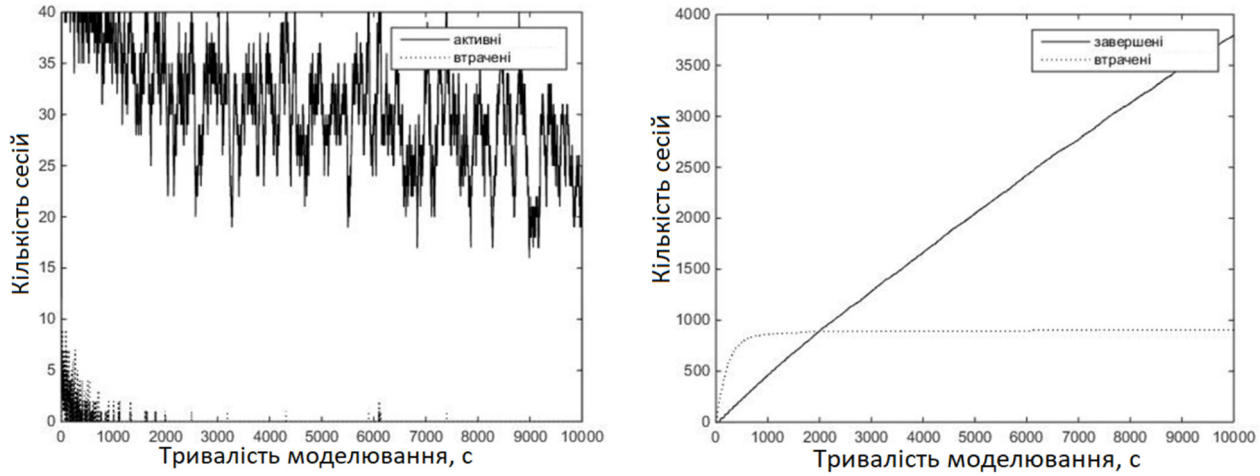


а б в

Рис. 2.15. Кількість запитів на надання послуг (а), початок (б) та завершення (в) обслуговування комунікаційних сесій

Кількість запитів на надання послуг, кількість сесій, які почалили обслуговуватися, та кількість сесій, які завершили обслуговуватися подано на Рис. 2.15, а, б та в, відповідно. Шляхом порівняння їх величин в однаковий момент моделювання можна визначичти кількість активних та втрачених сесій (Рис. 2.16, а).

На Рис. 2.16, б подано узагальнені показники роботи ІТС за комунікаційними сесіями у процесі моделювання.

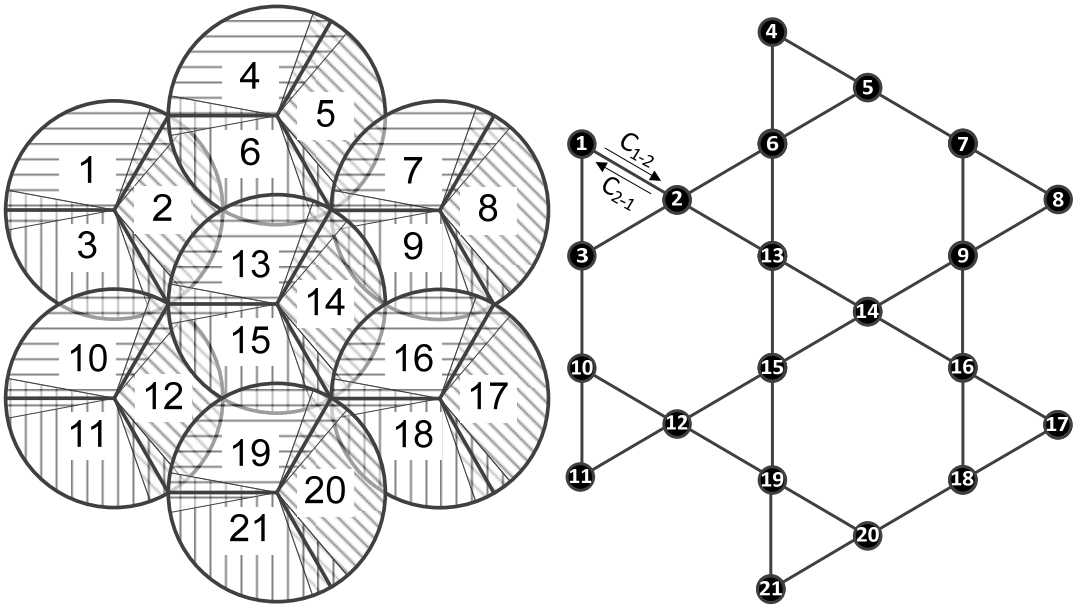


а б

Рис. 2.16. Моделювання користувацьких сесій: а) миттєві значення; б) кумулятивні значення

2.2.3. Балансування навантаження у процесі надання інформаційно-комунікаційних послуг на основі територіально-залежної моделі

Метод балансування навантаження з багатоетапним хендовером базується на використанні користувацького навантаження, яке знаходиться на краях секторів. Це допомагає збільшити доступність мережних ресурсів. Цей метод представляє структуру ІТС (Рис. 2.17, а) як граф (Рис. 2.17, б). Вузли представляють сектори зон обслуговування. Ребра вказують на наявність користувацького навантаження між секторами. Це означає, що існує спільна область обслуговування [127]. Ці ребра є обов'язковою умовою для алгоритму балансування навантаження з багатоетапним хендовером [51-57, 102, 288].



а) б)

Рис. 2.17. Кластер секторів (а) ІТС
як мережний граф (б)

Ваговий коефіцієнт, наприклад, ребра $_{1-2}$, вказує на пропускну здатність C_{1-2}, яка використовується користувачами сектора $_1$, за умови, що рівень сигналу сектора $_2$ дорівнює або перевищує мінімальне робоче значення.

Кожен вузол графа має свій власний коефіцієнт K_i, що казує на завантаженість i -го сектора:

$$K_i = C_i / C_{i\max}, \quad i = \overrightarrow{1,2..N}, \tag{2.44}$$

де C_i, $C_{\max i}$ – зайнята і максимальна пропускна здатність *i*-го сектору; N - номер сектору (вузла).

На Рис. 2.18 показана узагальнена блок-схема методу балансування користувацького навантаження, яка реалізує багатоетапний хендовер [6]. Алгоритм здійснює моніторинг завантаженості секторів та застосовує процедуру балансування навантаження при перевищенні порогу [13, 85, 222].

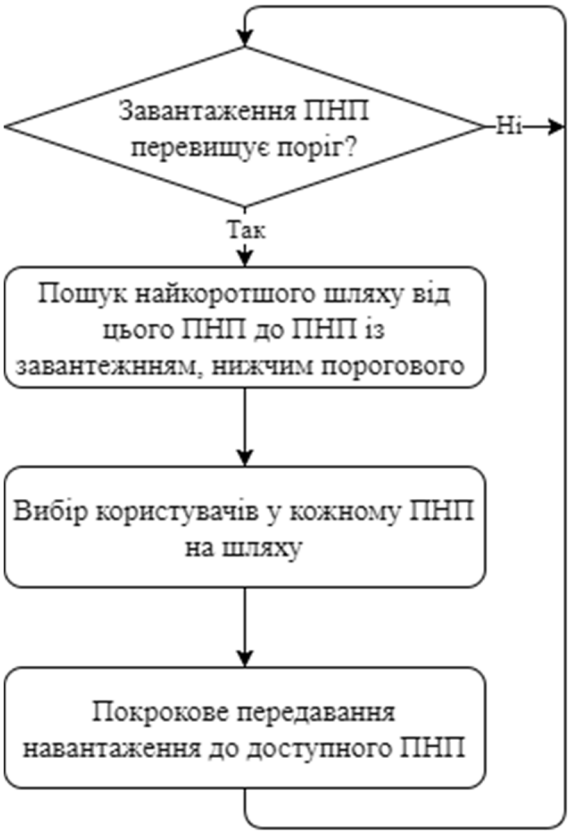


Рис. 2.18. Узагальнена блок-схема методу балансування навантаження з багатоетапним хендовером

Балансування навантаження починається з пошуку наявного сектора з рівнем завантаження, що є нижчим порогового. Сектор доступний, коли є шлях до нього з перевантаженого сектора. Пропускна здатність маршруту повинна перевищувати інтенсивність навантаження, яке необхідно передати з перевантаженого сектору, щоб зменшити використання ресурсів нижче порогового рівня.

Значення пропускної здатності маршруту менше або рівне пропускної здатності його мінімального ребра. Це означає, що можна вибрати користувачів для передачі обслуговування в наступному секторі на маршруті. Користувачі вибираються на основі максимального рівня потужності сигналу на приймачі від ПНП цільового сектора та на основі мінімального значення їх швидкості руху [84, 201].

Після цього проводимо балансування навантаження на основі даного маршруту. Передачі починаються від передостаннього сектора до останнього і закінчуються від першого до другого. У цьому методі набір передач обслуговування на маршруті називається багатоетапним хендовером.

Для проведення моделювання вважалося, що для кожного підключеного терміналу одночасно повинно бути не більше однієї сесії. Усі сесії вимагають однакової пропускної здатності і мають однаковий пріоритет. Сеанс успішно завершено, якщо його тривалість завершена, і, навпаки, сеанс втрачено, коли його не вдалося запустити або закінчено достроково.

Територіальна структура однакова з трьома секторами в кожній зоні обслуговування. Територія має прямокутну форму. Користувачі не залишають територію у процесі переміщення під час моделювання. Сектор перевантажений, коли його рівень завантаження перевищує порогове значення.

Вхідні параметри для моделювання:

- 3000 користувачів у системі;
- 100 с - середній інтервал між сесіями;
- 110 с - середня тривалість сесій;
- 50 сесій - максимальна кількість сесій з одного терміналу;
- 24 сесії - максимальна кількість активних сесій для кожного сектору;
- 1000 метрів – геометричні розміри території;
- 2000 секунд - тривалості моделювання;
- 33 м / с - максимальна швидкість руху користувачів;
- 90% - поріг звантаженості сектора.

Перед початком моделювання генеруємо::

– індивідуальні траєкторії для кожного користувача;
– заплановану мережну активність для кожного користувача;
– структура ІТС;

Вхідні дані для двох режимів моделювання:

– Нормальний режим, який використовує лише базове балансування навантаження, тобто лише сусіди беруть участь у процесі балансування;
– З LB – в цьому режимі використовується запропонований метод балансування навантаження з багатоетапним хендовером.

2.2.4. Характеристика стану процесу надання послуг на основі територіально-залежної моделі

На Рис. 2.19 показано кількість активних сесій в системі кожної секунди у двох режимах: без (звичайний режим) та з використанням (з LB) методу балансування навантаження з багатоетапним хендовером [256]. Оскільки моделювання системи починається без активних сеансів, то в діапазоні від 0 с до 100 с їх число збільшується. Цей інтервал однаковий для обох режимів, як показано на Рис. 2.19 – Рис. 2.21.

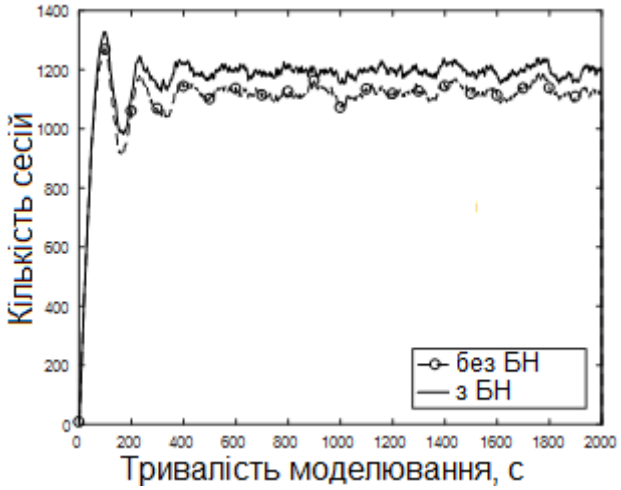


Рис. 2.19. Кількість активних сесій в системі без (нормальний режим) і з використанням (з LB) методу балансування навантаження

Далі, починаючи з 200 с, система входить у режим насичення, і ми можемо змінювати кількість активних сесій (Рис. 2.19) та рівень втрат сесій [7] (Рис. 2.20)

для обох режимів роботи. Використання запропонованого методу балансування навантаження дозволяє зменшити кількість втрачених сесій на 14% [244].

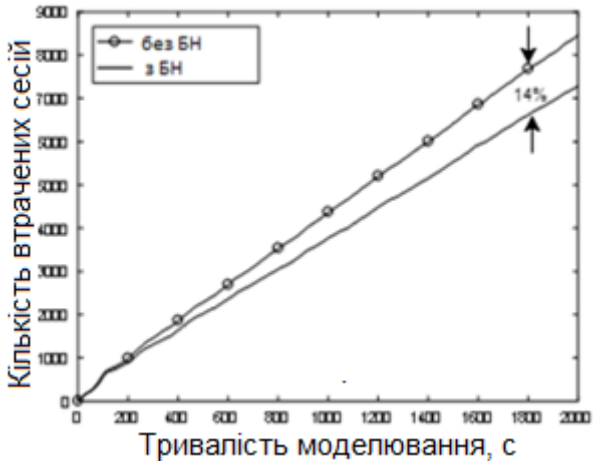


Рис. 2.20. Кількість втрачених сесій без та з використанням алгоритму балансування навантаження (з БН)

Вивільнення ресурсів в перевантажених секторах, які використовують цей метод, супроводжується збільшенням кількості хендоверів (Рис. 2.21). В результаті збільшується об'єм службових даних, що може знизити продуктивність мережі.

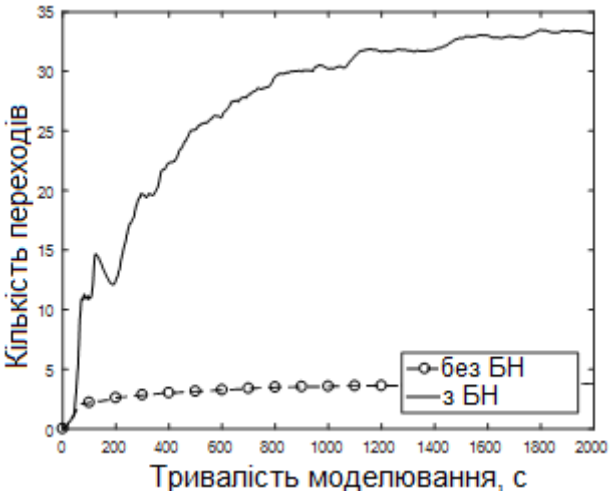


Рис. 2.21. Середня миттєва кількість хендоверів для двох режимів моделювання

На Рис. 2.22 показано співвідношення між середньою кількістю миттєвих хендоверів та середньою кількістю хендоверів, які використовуються лише для балансування навантаження.

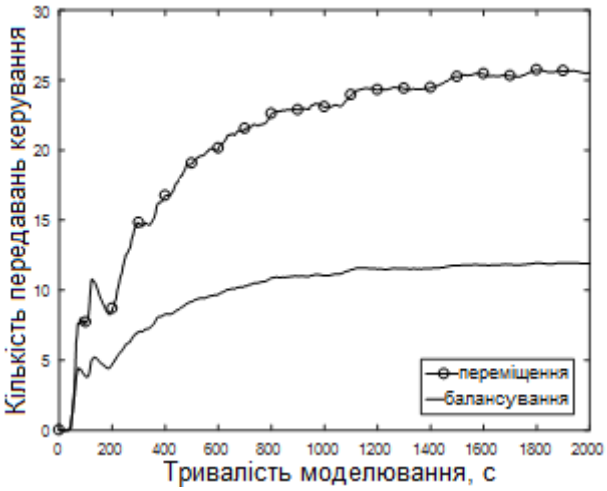


Рис. 2.22. Середнє значення хендоверів і кількість передавань, які використовуються для балансування навантаження

Найдовше передавання обслуговування протягом часу моделювання включало 17 хендоверів. Це означає, що для зниження рівня завантаженості в одному секторі був найближчий сектор із навантаженням, нижчим від порогового значення, лише за 17 кроків за визначеним мережним графом.

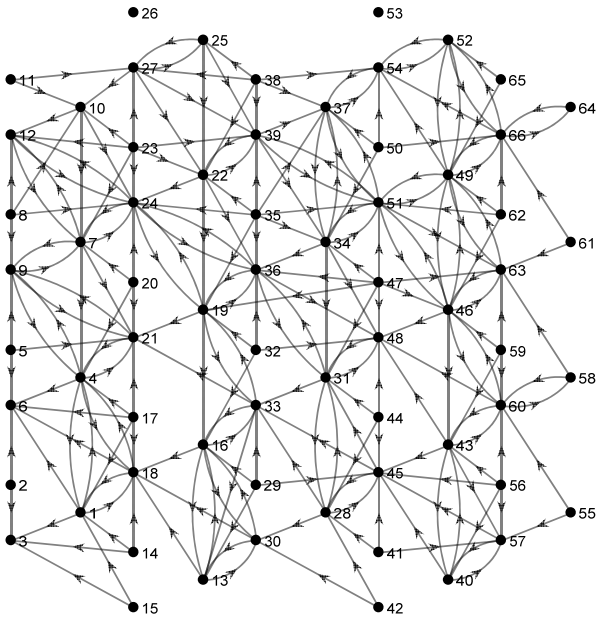


Рис. 2.23. З'єднання вузлів на останній секунді моделювання

На Рис. 2.23 показаний напрямлений граф мережі, який визначається на останній секунді моделювання. Він включає в себе всі 66 секторів (вузлів) згенерованої ІТС.

2.3. Методи побудови та надання послуг у розподілених інформаційно-комунікаційних системах

2.3.1. Представлення системи у вигляді платформи надання послуг

Розглянемо матрицю суміжності $\|c\|$, що описує топологію службових взаємозв'язків в інфраструктурі SDP. Ми можемо проаналізувати деяку мережну сервісну систему з чергою з N-вузлами. Зазначимо, що, з одного боку, ця матриця може описувати структурні властивості мережної інфраструктури SDP, а з іншого боку – розподіл потоків у мережі. Отже, припустимо, що структурний опис SDP відповідає матриці $\|c\|$. Відповідно, опишемо розподіл сервісних процесів між інтерфейсами сервісних вузлів структури SDP $\|c\|$ за допомогою матриці потоків λ_{ij}. У загальному випадку попередньо визначено, що інфраструктура сервісу SDP розроблена відповідно до вимог SLA споживачів послуг.

Зрозуміло, що для сервісних інтерфейсів вузлів і потоків транзитних вузлів необхідні ресурси, які характеризуються [81, 164]:

- Адекватністю розміру буфера та кількості ресурсів для обробки потоків, які запитують користувачі;
- Обмеженою їх кількістю – це особливість реальної інфраструктури SDP.

Після цього відзначимо, що для кожного потоку в SDP справедливе рівняння, що описує затримку сервісного потоку між суміжними вузлами i та j:

$$t_{ij} \approx k\Lambda_0\lambda_{ij}, \tag{2.45}$$

де $k = 1$ - пропорціональний індекс, Λ_0 – параметр, який залежить від логіки обслуговування та відповідного алгоритму, який оцінює буферний ресурс сервісного вузла. У цьому можемо визначити:

$$\Lambda_0 = \left[t_{proc} + t_{wait}\left(\frac{\overline{d_q}}{d_{max}}\right)\right] t_{proc}^{-1} = 1 + \frac{t_{wait}}{t_{proc}}\frac{\overline{d_q}}{d_{max}}, \tag{2.46}$$

де t_{proc} – тривалість оброблення одного запиту в сервісному потоці з незавантаженим сервісним вузлом (ресурси не використовуються, а буфер порожній); t_{wait} – час затримки, поки ресурс службового вузла може бути готовий обробляти запит обслуговування потік в очікуванні у черзі, часто вказується в більшості SLA, переважно для потоків реального часу, перевищення цього часу може відповідати відмові типу DoS); $\overline{d_q}$ – середнє робочеп завантаження сервісного вузла (середня тривалість черги сервісних запитів, це завантаження може бути оцінено або вимагатиметься SLA і фіксоване для розрахунків); d_{max} – це максимально можливе завантаження сервісного вузла (довжина черги сервісних запитів, як правило, фіксується як d_{max} параметр основного обладнання SDP).

Припустимо, що для ефективної роботи SDP не слід перевантажувати. Тому сервісні вузли інфраструктури SDP описано моделлю черг в нотації Кендалла *M/M/*1/*N* де $N \to \infty$ (буфер буде порожній або не використовуються сервісні ресурси).

Для розрахунку середньої затримки потоку інфраструктури SDP ми використовували вилучення формули Клейнкрека з закону Літла:

$$T = \frac{1}{\gamma} \sum_{k=1}^{N} \sum_{l=1}^{N} \lambda_{kl} \cdot t_{kl}, \tag{2.47}$$

де γ – повне завантаження розподіленої мережі SDP, яке може бути розраховане за наступною формулою :

$$\gamma = \sum_{i=1}^{N} \sum_{j=1}^{N} \gamma_{ij}, \tag{2.48}$$

де γ_{ij} – це сума сервісних потоків між вузлами i та j .

З урахуванням збалансованого симетричного характеру запитів на обслуговування та відповіді через взаємозв'язок між інтерфейсами та транзитним розподілом сервісних потоків ми можемо змінити (2.48) таким чином [125]:

$$T = \frac{1}{\gamma} \sum_{k=1}^{N} \sum_{l=1}^{N} \lambda_{lk} \cdot t_{kl} . \quad (2.49)$$

Що стосується (2.45), значення затримок сервісних потоків опишемо за допомогою такого співвідношення:

$$T = \frac{\Lambda_0}{\gamma} \sum_{k=1}^{N} \sum_{l=1}^{N} \lambda_{lk} \cdot \lambda_{kl} . \quad (2.50)$$

Оптимізація конфігурації сервісної інфраструктури за критерієм мінімальної затримки сервісного потоку може бути досягнути, якщо навантаження системи вважається пропорційно залежним від субтопологічних властивостей SDP (обчислення при суміжному перетворенні матриці для прогнозування робочого навантаження), яке виникло внаслідок підняття матричних елементів $\|c\|$ до степеня N-1. Як правило, співвідношення для сервісного потоку може бути записано у вигляді такого функціонального співвідношення:

$$T_{min} = \frac{\Lambda_0}{\gamma} \sum_{k=1}^{N} \sum_{l=1}^{N} \lambda_{lk} \cdot \lambda_{kl} - \frac{\beta_{-1}}{\gamma} \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij}^{N-1} \cdot c_{ji}^{N-1} . \quad (2.51)$$

Критерій мінімальної затримки доставки сервісних потоків:

$$T \to T_{min} .$$

де β_{-1} – це індекс пропорційності робочого навантаження в формулі (2.51), узагальнений і може бути представлений як::

$$\beta_{-1} = \frac{c^{N-1}}{\max\left\{c^{N-1}\right\}} = N^{2-N} .$$

Для гомогенної уніфікованої архітектури SDP ми припускаємо що $\forall \beta_{-1} = \text{const}$ Давайте визначимо:

$$S(\lambda_{ij}, \lambda_{ji}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_{ij} \cdot \lambda_{ji} ,$$

$$D(c_{ij}^N, c_{ji}^N) = \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}^N \cdot c_{ji}^N .$$

Мінімізацію часу доставки сервісного потоку $T \to T_{min}$, а в обмежених випадках $T \to 0$, при оптимальному управлінні сервісним потоком можна описати умовами мінімізації (2.51). З урахуванням введених позначень ця умова набуде вигляду:

$$\lim_{\|\lambda\| \to \|c\|} T = 0 ,$$

і відповідно:

$$\lim_{\|\lambda\| \to \|c\|} \left[\frac{\Lambda_0}{\gamma} \sum_{k=1}^{N}\sum_{l=1}^{N} \lambda_{lk} \cdot \lambda_{kl} - \frac{\beta_{-1}}{\gamma} \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}^{N-1} \cdot c_{ji}^{N-1} \right] =$$
$$= \lim_{\|\lambda\| \to \|c\|} \left[\frac{\Lambda_0}{\gamma} S(\lambda_{lk}, \lambda_{kl}) - \frac{\beta_{-1}}{\gamma} D(c_{ij}^{N-1}, c_{ji}^{N-1}) \right] = 0.$$

Залучаючи індекси сусідніх матричних елементів $\|c\|$ до опису сервісних потоків з індексами $\|\lambda\|$ у випадку $\|\lambda\| \to \|c\|$, $S(\lambda_{lk}, \lambda_{kl}) \to D(c_{ij}^{N-1}, c_{ji}^{N-1})$ ми отримуємо наступне поліноміальне функціональне рівняння:

$$\Lambda_0 S(\lambda_{ij}, \lambda_{ji}) - \beta_{-1} \left[S(\lambda_{ij}, \lambda_{ji}) \right]^{\frac{1}{N-1}} = 0 . \qquad (2.52)$$

Це рівняння можна інтерпретувати як стан оптимальності управління сервісними потоками SDP. Замінивши функцію (2.52), ми отримаємо розв'язок для завдання оптимізації розподілу сервісних потоків у інфраструктуруі SDP:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_{ij} \cdot \lambda_{ji} = \left(\frac{\Lambda_0}{\beta_{-1}} \right)^{\frac{2-N}{N-1}} . \qquad (2.53)$$

Отримавши значення λ для сервісних потоків та деякі технічні вимоги, зазначені в SLA, ми можемо обчислити всі необхідні параметри SDP в деякій масштабованій сервісній інфраструктурі за рівнянням з формулою (2.46), заміненою на (2.53). Таким чином, для правильного та ефективного управління сервісними потокамим ми представляємо наступний критерій:

$$\sum_{k=1}^{N}\sum_{l=1}^{N}\lambda_{kl}\cdot\lambda_{lk} \Leftrightarrow \sum_{k=1}^{N}\sum_{l=1}^{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\Delta\gamma_{ij}\cdot x_{kl}^{(i,j)}\right)^2 - \\ -\left(\frac{\Lambda_0}{\beta_{-1}}\right)^{\frac{2-N}{N-1}} \to \min \tag{2.54}$$

де $x_{kl}^{(i,j)}$ – частка сервісного потоку γ_{ij} між сервісними вузлами i та j що розподілені від вузла k до l, ця сума може бути розрахована на кожен сервісний вузол в SDP. Умови захисту сервісного повинні виконуватися $x_{kl}^{(i,j)}$ повинні враховувати (2.54):

$$\sum_{k=1}^{N} x_{kl}^{(i,j)} - \sum_{k=1}^{N} x_{lk}^{(i,j)} = \begin{cases} -1, l = i, \\ 0, l \neq i, j, \\ 1, l = j. \end{cases}.$$

Таким чином, для $N \to \infty$ керування сервісними потоками здійснюється з мінімальним використанням ресурсів $\overline{d_q}$.

2.3.2. Метод проектування фізичної структури інформаційно-комунікаційної системи

Одним з найпоширеніших інструментів дослідження мереж будь-якого типу чи класу мереж є теорія графів. У роботі пропонується використовувати теорію графів. Для цього введено наступні позначення:

- $G = (V, E)$ - граф що відповідає структурі досліджуваної мережі;
- V - множина вершин, що відповідають мережним вузлам;
- E – множина ребер, що відповідають мережним каналам.

Теорія графів дає змогу в деталях описати як саму структуру мережі так і її поточний стан, за рахунок введення станів ребер та вузлів, що в процесі функціонування мережі дає змогу моделювати вплив завантаження ребер чи вузлів на доступність каналів та якість обслуговування потоків у мережі. По суті параметри ребер є ключовими параметрами, які відображають наскільки ефективно функціонує мережа та дають змогу робити висновки про необхідність внесення змін в функціонування мережі.

Однією з основних характеристик графа є його зв’язність. У повнозвязному графі між кожною парою вершин існує ребро. У реальних телекомунікаційних мережах таке правило швидше має місце при дослідженні логічної топології. Для дослідження фізичної топології достатньо щоб виконувалася умова, що між будь якими двома вершинами графа повинен існувати шлях, при цьому кожна вершина повинна мати не менше ніж одне з’єднання з бідь якою іншою вершиною в графі. Тому першим етапом дослідження мережі є створення графу, параметри якого відповідають структурі фізичної мережі, після чого можна застосовувати метод аналізу ієрархій для дослідження якості обслуговування потоків в існуючій мережній структурі.

Метод аналізу ієрархій дає змогу на основі порівняння двох суміжних варіантів прийняти рішення. Недоліком цього методу є те, що вхідні дані формуються експертами, а отже не завжди є повністю об’єктивними, а тому він не може використовуватися для порівняння структури графів. Натомість, вхідні дані визначаються на основі вибору максимального значення якості надання сервісу у вибраній мережній структурі. Спочатку для дослідження обраної мережі необхідно сформувати структуру завдання (Рис. 2.24). Найвищий рівень цієї структури буде відповідати цілі завдання. Ціллю у цьому завданні є знайти таку структуру мережі, яка дасть змогу забезпечити максимальну якість обслуговування для усіх потоків. На наступному рівні зібрано критерії для оцінювання якості сервісу. У роботі використано два наступні критерії: інтенсивність потоку I та завантаженість мережі L. Варто зауважити, що завантаженість не є показником рівня сервісу, проте завантаженість є основним фактором, який може призвести до низької якості сервісу та відображає ефективність використання ресурсів мережних каналів. Інтенсивність потоку це максимальна швидкість передавання даних між двома обраними вершинами по обраному шляху з урахування поточного завантаження каналів [170].

Рис. 2.24. Розроблення ієрархічного представлення задачі дослідження мережі на основі методу аналізу ієрархій

Для подальшого дослідження необхідно забезпечити уніфікацію формування інформаційних потоків та пропускної здатності ребер.

Спільний принцип формування потоків та пропускної здатності ребер забезпечує порівняння структур між собою, що також уможливить заміну маршрутів у логічній структурі мережі. Тому у роботі прийнято ряд наступних обмежень:

- абсолютно всі потоки паралельно протікають в мережі, при цьому по обраному маршруту протікає лише один ненаправлений потік $j, i \neq j$;
- максимальна пропускна здатність усіх каналів є рівною;
- пропускна здатність каналу пропорційна розподіляється для всіх потоків;
- обмеження по швидкості для потоку між вершинами i та $j, i \neq j$ є найменша пропускна здатністю в каналі в обраному маршруті.

За таких умов кожен потік змагається за своє існування, в результаті чого система стабілізується і потоки займають потрібну їм пропускну здатність в залежності від їх пріоритету та вимог щодо якості обслуговування. Таким чином у якості першого критерію використано матрицю інтенсивності потоків I. Проте, недостатнє завантаження мережі може виникнути, якщо в різних каналах є різна кількість маршрутів, що пояснюється введеним обмеженням про пропускну здатність відповідно до теореми про максимальний потік. Другим критерієм обрано завантаженість кожного окремого ребра L, представлений матрицею завантаженості.

На третьому рівні визначеної ієрархії розміщені різноманітні альтернативні структури графів. Введені критерії використовуються при аналізі та оцінці всіх потоків що існують в мережній структурі. Для кожної альтернативи визначається показник ефективності P для визначення якого використовується наступний вираз:

$$P = \max\left[\sqrt[|V|]{\prod_{i=1,\, j=1}^{|V|} I_{i,j}^{w_1}} + \sqrt[|V|]{\prod_{i=1,\, j=1}^{|V|} L_{i,j}^{w_2}} \right],\ i \neq j\ , \tag{2.55}$$

де $_P$ – показник, що визначає ефективність структури, w_k – відносна вага k-го критерію, $|V|$ – кількість вершин у графі, $_I$ - матриця, яка визначає інтенсивність потоків, $_L$ - матриця завантаженості графа.

Мережна структура, у якій обслуговування всіх потоків відбувається з максимальним рівнем якості надання сервісу і є цільовою функцією задачі, що представлена як $\max(P)$. Відповідно до описаного методу вибору мережної структури, здійснюється вибір фізичної структури, що найбільше впливає на мережі з низькою зв'язністю.

2.3.3. Оптимізація логічної структури інформаційно-комунікаційної системи у процесі надання інформаційно-комунікаційних послуг

Чим більше ребер існує в графі, тим графі є більш зв'язним, а отже існує більше варіантів маршрутизації потоків та розподілу ресурсів мережних каналів, що призводить до того що фізична структура уже не є основним обмеженням при плануванні передавання даних у мережі.

Логічна структура створюється на основі розподілу існуючої пропускної здатності каналу між двома і більше потоками у процесі вибору оптимального маршруту. При перерахунку маршрутів з урахуванням критерію максимального використання вільних мережних ресурсів, у результаті розв'язку оптимізаційної задачі буде отримано структуру, в якій пропускна здатність усіх каналів використовується максимально ефективно [64, 272, 284].

Знайти розв'язок для поставленого завдання можна за наступним алгоритмом. Спочатку необхідно знайти усі шляхи, сформувати матриці найкоротших шляхів W і завантаженості ребер потоками R. Завантаженість ребра відображається матрицею R. Варто зазначити, що кількість потоків у ребрі і визначають сумарну завантаженість ребра [128, 165].

Нехай дано граф $G=(V,E)$, де $e_{i,j}=(i,j),\ i\neq j,\ i,j\in V$ описує ребро між вершинами i та j. Слід використати алгоритм Флойда для знаходження матриці найкоротших шляхів W. В цій матриці кожен елемент є шляхом $\mu(i,j)=\{e_{i,l},e_{l,m},\ldots,e_{k,j}\},\ i\neq j,\ i,l,m,k,j\in V$, що включає набір ребер $e_{l,m}$, з яких формується найкоротший шлях. Очевидно, що для випадку з повнозв'язним графмо кожен елемент буде представлений одним ребром між обраними вершинами. В інших випадках, одне і те саме ребро може зустрічатися в різних маршрутах, а отже і в різних елементах матриці. Для випадку з неорієнтованим графом, беруться до уваги лише шляхи, які знаходяться по одну сторону від діагоналі [24].

Матриця R дає змогу оцінити число маршрутів, що проходить через кожне ребро графу. Елементи матриці можуть набувати значень від 0 до $|E_{\max}|$. Нуль означає, що через у відповідному ребрі не існує маршрутів. Ця матриця формується на основі матриці найкоротших шляхів, за рахунок додавання всіх потоків, що протікають в ребрі $e_{l,m}$ $R_{l,m}=\begin{cases}R_{l,m}+1,\ e_{l,m}\in\mu\\ R_{l,m},\ e_{l,m}\notin\mu\end{cases},\ \mu\in W_{rout},\ \mathrm{l,m}\in\mathrm{V}$.

Значення усіх елементів матриці R будуть характеризувати завантаження ребра. Якщо значення усіх елементів є рівними, то завантаження усіх каналів є однаковим, а канальні ресурси рівномірно розподілені між наявними маршрутами. Для підбору логічної мережної структури використовується критерій $\min[\max(R)-\min(R)]$, різний рівень завантаженості різних ребер описується елементом в квадратних дужках. Цей елемент також вказує на проблемні ділянки в мережі, де може виникнути вузьке місце у процесі передавання даних, за умови що інтенсивність потоків між кожною парою вузлів не є заданою. Проте, існують

виключення. Наприклад, в IP мережах граничний пристрій має один фізичний зовнішній канал [130], у якому проходять безліч логічних, проте сам канал не є потенційно перевантаженим. Незважаючи на це, основною функціює матриці завантаженості є визначення сумарної завантаженості каналу, за рахунок визначення та сумування швидкостей потоків, що проходитимуть через досліджуване ребро. Якщо інформація про інтенсивність потоків між будь-якою парою вузлів відсутня, до уваги береться лише сам факт існування потоку.

У мережах з великою кількістю ребер існує можливість знайти альтернативні маршрути для передавання даних. Такі маршрути зазвичай утворюються ребрами, пропускна здатність яких використовується не ефективно. Проте тут слід звертати увагу не лише на самі ребра, а в цілому на підграф, що формується набором таких ребер, оскільки будь який маршрут складається не менше ніж з одного ребра. Таким чином необхідно вибрати всі ребра, пропускна здатність яких використовується неефективно і сформувати з них зв'язний підграф, для якого застосувати спосіб визначення найменшого зв'язного дерева (MST, англ. Minimum Spanning Tree). Отримане в результаті дерево $G_{MST}(V,E_{\min})$ міститиме всі вузли графа G. В отриманому графі прокладено лише один маршрут через недовантажені ребра мережі, який і є обхідним маршрутом по відношенню до основного маршруту.

Здійснювати пошук обхідні маршрутів слід за допомогою алгоритму Флойда до знайденого найменшого зв'язного дерева $G_{MST}(V,E_{\min})$ та сформувати матрицю шляхів W_{MST}. В результаті пошуку сформована матриця W_{MST} містить не тільки найкоротші шляхи, але і обхідні шляхи, максимальний ранг яких становить $r=|V|-1$, що зумовлено відсутністю петель в отриманій структурі.

Для максимального покращення розподілу ресурсів у мережі слід виключити дублюючі шляхи та шляхи з максимальним рангом. Також доцільно проводити заміну шляхів, для яких різниця значень метрик є мінімальною, оскільки додаткове навантаження на ребра обхідного шляху буде мінімальним.

Кожен обхідний шлях може по своєму впливати на ефективність розподілу ресурсів у досліджуваній логічній структурі, а тому необхідно вибрати такий обхідний шлях, який забезпечить максимальну ефективність розподілу ресурсів. Для вирішення цієї задачі використовується показник ефективності розподілу ресурсів K (2.56). Значення показника K зменшується зі зростанням частки вивільненого ресурсу та підвищенням рівня якості надання сервісу.

$$K = \min\left(\sum_{i,j}\left(\frac{\sum_{i,j} R_{i,j}}{|E|} - R_{i,j}\right)\right) \tag{2.56}$$

Для визначення оптимального обхідного шляху необхідно по черзі для всіх можливих обхідних шляхів порахувати значення показника K, використовуючи формулу (2.56). Найменше значення показника дасть змогу визначити обхідний шлях, що забезпечить максимально ефективний розподіл ресурсів.

Очевидно, що різні маршрути по різному будуть впливати на перерозподіл ресурсів, особливо, якщо будуть задані інтенсивності потоків між вершинами, що в свою чергу створить додаткові обмеження. В цьому дослідженні значення інтенсивносте не використовуються, а отже показник відображає покращення рівня сервісу з точки зору перерозподілу кількості потоків в одному ребрі. У випадку заміни маршруту, весь вільний ресурс буде використано для підвищення якості обслуговування потоків, що паралельно проходять в тих самих ребрах. Тому показник K відображає кількісну зміну, а критерій мінімуму буде відображати збільшення об'єму ресурсів для решти потоків.

Вибір оптимальної логічної структури ускладняється зі зростанням кількості вузлів, оскільки зростає число можливих варіантів логічної структури. Використовуючи алгоритм MST, проводить вибір альтернативних шляхів. Як наслідок, кількість замін шляхів залежить від кількості незалежних мінімальних зв'язних дерев. Для структур, в яких кількість ребер перевищує $|E| = y \cdot |E_{\min}| + 1$, $y = 2,3 \ldots N$ існуватиме y незалежних дерев. У кожному дереві буде вибрано маршрут для заміни.

2.4. Висновки до другого розділу

1. Запропоновано математичну модель першого етапу надання послуг в гетерогенних сервісно-орієнтованих системах – пересилання запиту. Новизна пропонованого підходу полягає в прозорості використання мережевих ресурсів. Це означає, що мережа адаптована до вимог надання послуг.

2. Сформульовано завдання розподілу ресурсів гетерогенних інформаційно-комунікаційних систем між сервісними потоками як нелінійну задачу оптимізації. Його вирішення базується на використанні методу розкладання на основі агрегації функцій корисності. Крім того, сформульовано підзадачу визначення та агрегації функцій корисності. Перевага вищезазначеного підходу полягає в тому, що ми можемо об'єднати вимоги до послуг, орієнтовані на різні цільові параметри. Тому задача розподілу може бути вирішена незалежно від задачі передавання багатокомпонентного потоку з урахуванням миттєвих значень функцій корисності послуг.

3. На основі територіально-залежної моделі обслуговування отримано множину координат абонентів у процесі моделювання, класифіковано користувачів за критерієм швидкості руху, що забезпечує можливість централізації керування мобільністю в моменти перевантажень фрагментів мережі. За результатами моделювання сформовано приклади траекторій руху користувачів за відомою їх швидкістю та напрямом руху, на основі множини яких виділено основні варіанти переміщення користувачів.

4. Проведено моделювання процесу надходження запитів на ініціацію комунікації з використанням моделі активності клієнтів з використанням інформації про характеристики сеансів, отриманої шляхом імітаційного моделювання (Рис. 2.15). Отримано значення завантаження та втрат у системі (Рис. 2.16, а), де спостерігаються перевантаження фрагментів мережі. Ці перевантаження у подальшому знівельовано шляхом використання розробленого методу балансування навантаження. Отримані параметри процесів комунікації дають змогу охарактеризувати процес надання послуг у системі шляхом розрахунку імовірності втрат і системної доступності.

5. Проведено моделювання поведінки абонентів у мережі коміркового зв'язку з урахуванням їх активності та переміщення.

6. Розроблений метод розбиття території на зони обслуговування забезпечується мінімізацію кількості точок присутності послуг.

7. Моделювання переміщення споживачів послуг забезпечує можливість прогнозування завантаження окремих точок присутності послуг, що дає змогу підвищити ефективність балансування навантаження.

8. Модель активності користувачів послуг дає змогу врахувати тривалості комунікаційних сеансів, інтенсивності надходження запитів, процеси обслуговування та максимальну кількість конкурентних сеансів у системі, що забезпечує підвищення адекватності моделювання завантаженості системи точок присутності послуг.

9. Запропонований метод балансування навантаження з багатоетапним перемиканням точок присутності послуг збільшує доступність інформаційних ресурсів у періоди пікових навантажень. Збитки від втрачених комунікаційних сеансів скоротились на 14% у порівнянні із звичайним режимом роботи системи. Найдовша ланцюжок перемикання точок присутності під час моделювання містив 17 перемикань.

10. Запропоновано структурну та функціональну схеми центру керування та первинної обробки даних індикативного моніторингу визначено його склад та завдання. Розроблено універсальну систему інтерпретації команд центру керування у складі мобільного модуля індикативного моніторингу, яка, на відміну від відомих, забезпечує використання теоретично необмеженого числа різних мобільних платформ.

11. Глобальне масштабування та ускладнення розподіленої структури SDP, потребує забезпечення достатньої кількості ресурсів обробки сервісів для відповідної взаємодії між користувацькими та сервісними інтерфейсами під час виконання деяких застосунків. Ефективна і правильна конфігурація SDP (як структури взаємоз'єднання фізичних інтерфейсів, так і управління потоком) здійснена шляхом розрахунку необхідних параметрів для максимізації

продуктивності службової інфраструктури та забезпечення процесів передавання сервісних потоків з мінімальними затримками, що значно підвищує показники якості надання послуг масштабованої розподіленої сервісної системи та унеможливлює відмову в обслуговуванні коли кількість активних користувачів та їх вимоги різко зростають. Представлено відповідні критерії вирішення завдань у суворій відповідності з сформульованою гіпотезою математичними перетвореннями з отриманням практично реалізованих формул та підходів.

12. Завдання забезпечення потоків даних задовільним сервісом полягає у оптимальному поєднанні фізичного та логічного шарів структури ІТС. Їх важко розділити, а неоптимальне їх поєднання згумно відображається на ефективності всієї системи. Ступінь зв’язності мережної структури визначає вплив цих шарів на якість надання послуг. У зв’язку з цим у роботи запропновано здійснювати вибір фізичної структури та адаптувати логічну структуру, визначену шлязом маршрутизації, з метою підвищення якості обслуговування користувачів.

РОЗДІЛ 3. МЕТОДИ СТРУКТУРНО-ФУНКЦІОНАЛЬНОГО СИНТЕЗУ ТЕЛЕКОМУНІКАЦІЙНОЇ ПІДСИСТЕМИ СИСТЕМИ ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМ

3.1. Структурний синтез оверлейних інформаційно-телекомунікаційних систем

Якість надання сервісу в телекомунікаційних мережах суттєво залежить від методів управління та передавання трафіку. Зазвичай в мережах невеликих розмірів маршрути можуть конфігуруватися системним адміністратором. Такі маршрути називаються статичні і не змінюються впродовж функціонування мережі. В таких випадках системний адміністратор може конфігурувати інші мережні параметри, зокрема алгоритми буферизації та обробки пакетів з урахуванням пріоритетів потоків, що встановлюються на основі загально відомої класифікації сервісів. Ручний спосіб налаштування доцільно використовувати при невеликій кількості користувачів та різноманітності потоків. Проте, коли мова заходить за мережі мобільних операторів ситуація кардинально змінюється. В таких мережах, зокрема мережах транспортного класу, де кількість потоків та їх різноманітність невпинно зростають, статична конфігурація не може забезпечити ефективного функціонування мереж. За умов статичної конфігурації при зростанні навантаження від користувачів дуже швидко постане питання уникнення перевантаження магістральних каналів, забезпечення необхідної якості обслуговування пріоритетним потокам у порівнянні з непріоритетними та ефективний розподіл мережних ресурсів з для забезпечення ефективного використання мережної інфраструктури. Тому у таких мережах використовуються протоколи динамічної маршрутизації, які здійснюють прокладання маршруту з урахуванням великої кількості критеріїв. Це дає змогу не тільки адаптувати мережі під вхідне навантаження, але і розширювати можливості динамічної маршрутизації за рахунок введення додаткових критерії та налаштування таких протоколів конкретно під потреби оператора мережі.

Найпоширенішим інструментом для дослідження мереж в телекомунікаціях є теорія графів. Зокрема використання теорії графів дає змогу визначати ефективність вибору маршруту та провести балансування навантаження з урахуванням параметрів ребер та вершин графа. Більше того, за рахунок введення додаткових параметрів, що характеризують ребра та вузли, виникає можливість дослідження пропускної здатності та затримки передавання пакетів з урахування поточного стану мережі.

3.1.1. Показник ефективності формування логічної структури інформаційно-телекомунікаційної системи

Основним недоліком існуючих протоколів динамічної маршрутизації є їх децентралізованість. Для знаходження оптимального маршруту передавання даних чи навіть просто для незначної реконфігурації існуючого маршруту, всі мережні пристрої повинні отримати обновлення про стан мережі, і допоки цей процес не завершиться мережа не в змозі гарантувати правильну поведінку та ефективне функціонування. Для впровадження нових механізмів маршрутизації та покращення існуючих є можливість вдосконалювати існуючі алгоритми за допомогою нових критерії, які дали б змогу здійснити оптимальніший вибір маршрутів. Проте, зазвичай зі збільшення кількості критеріїв зростає складність алгоритму знаходження оптимального рішення та час необхідний для реконфігурації. Більше того в умовах мультисервісного трафіку, стан мережі може суттєво змінитися ще до закінчення роботи алгоритму маршрутизації, і вже буде не актуальним на момент закінчення його роботи. Тому алгоритм необхідно повторювати заново. Очевидно, що здійснювати оптимізацію функціонування мережі в умовах її функціонування це задача не тривіальна, а тому одним з рішень такої проблеми є використання математичних моделей та оптимізаційних задача для визначення оптимальних маршрутів з урахуванням різного рівня навантаження мережних каналів. Прорахунок наперед усіх можливих варіантів, дасть змогу підготувати конфігурацію, яка може бути впроваджена в мережі негайно при виникненні відповідних умов.

Для реалізації такого підходу необхідно мати інструменти для прогнозування стану мережі в будь який момент часу, зокрема для прогнозування інтенсивності навантаження. Навантаженням описаних мереж є мультисервісний трафік, який складається з безлічі дрібних потоків, кожен з яких має власні вимоги для якості обслуговування. Як наслідок, для прогнозування навантаження необхідно проводити дослідження характеристик мультисервісного трафіку, що створюється абонентами в конкретній мережі, та сформувати адекватні імовірнісні моделі такого трафіку.

Часто науковці розглядають структурно-функціональні параметри мережних вузлів, які можуть бути використані для дослідження мультисервісного трафіку та сформувати стратегію маршрутизації. У таких стратегіях основним критерієм вибору маршруту є сукупність вимог до якості обслуговування конкретних потоків. Тоді, здебільшого, зосереджуються на двох параметрах при дослідженні якості обслуговування, а саме: затримка передавання даних по ребру та пропускна здатність кожного ребра.

Описані підходи вимагають витратити велику кількість часу для спостереження за мережею, аналізу параметрів мережі в різні моменти часу та виявлення чітких шаблонів поведінки, які і ляжуть в основу системи управління мережею. Отримані характеристики використовують при подальшому дослідженні маршрутизації на основі теорії графів. У роботі також враховано наступні обмеження:

- пропускна здатність маршруту обмежується найменшою пропускною здатністю ланки маршруту.
- пропускна здатність ребра розподіляється рівномірно між маршрутами, що проходять через нього.

В результаті застосування першого обмеження виникає недовантаження мережних каналів, в той час як застосування другого обмеження вирівнює навантаження між усіма вузлами, що на практиці зустрічається досить не часто. У роботі увагу зосереджено на аналізі мережних топологічних структур щодо

ефективного вибору маршрутів, який здійснюється на основі порівняння маршрутів за критеріями затримки передавання та пропускної здатності.

В багатьох випадках на вибір фізичної топології мережі впливає фінансовий фактор. Після встановлення інфраструктури, системні адміністратори проводять конфігурацію оптимальних маршрутів та протоколів динамічної маршрутизації.

Для дослідження мережі у роботі використовується метод аналізу ієрархій. В результаті застосування цього методу для кожного варіанту топології отримується її оцінка відповідно до встановлених критеріїв. Ця оцінка дає змогу зробити висновок про можливість та доцільність використання того чи іншого протоколу динамічної маршрутизації на конкретній топології.

Дослідження мережної топології проводиться на основі моделі, яка включає два основні параметри, а саме кількість вузлів мережі та пропускна здатність каналів. У загальному випадку пропускні здатності всіх каналів будуть рівні, проте це правило може змінюватися залежно від досліджуваної топології.

Напершому етапі дослідження формується матриця сумжіності та матриця пропускної здатності мережних каналів. Мінімальна кількість ребер в досліджуваній мережі (r_{min}) може бути на одиницю меншою від кількості вузлів. Максимальна кількість ребер визначається за допомогою виразу (3.1).

$$r_{\max} = \sum_{i=1}^{r_{\min}} (N - i), \tag{3.1}$$

У кожній топології необхідно обчислити маршрути для кожного вузла. Для таких задач на практиці зазвичай використовують алгоритми Дейкстри та Беллмана-Форда. Результат роботи обох алгоритмів при однакових вхідних параметрах графу є однаковим, проте в алгоритму Дейкстри тривалість пошуку результату є довшою ніж у алгоритму Беллмана-Форда [169].

На наступному етапі формується матриця, кожен елемент якої відповідає ребру графа, а значення цього елемента відповідає кількості маршрутів, що проходять через це ребро. Попередній аналіз цієї матриці дає змогу встановити можливість підвищення ефективності вибраних шляхів. Оптимальним

розв’язком вважається розв’язок в якому всі значення всіх елементів в цій матриці рівні, тобто кількість маршрутів у всіх ребрах графа є однаковою. Після цього можна здійснити коректування оптимальності, що полягає у врахуванні характеристик потоків. Після цього можна стверджувати, що протоколи динамічної маршрутизації будуть функціонувати ефективно та обирати оптимальні маршрути передавання даних.

Якщо в отриманій матриці, одне із значень не є таким як інші, знайдений розв’язок вважається не оптимальним. В результаті використання такого розв’язку при імплементації протоколів маршрутизації, може виникнути перевантаження основного шляху в мережі при великій кількості недовантажених обхідних шляхів. Щоб виправити таку ситуацію, необхідно внести зміни в фізичну топологію, що на практиці зазвичай важко реалізувати.

На основі отриманої матриці можна отримати пропускну здатність кожного маршруту, а саме пропускна здатність маршруту буде дорівнювати пропускній здатності каналу розділеній порівну між всіма маршрутами, що проходять через цей канал. Якщо ж отримана матриця буде неоднорідною, то це призведе до того, що різні ланки шляху будуть мати різну пропускну здатність, і як наслідок пропускна здатність маршруту буде обмежуватися ланкою, значення пропускної здатності якої є мінімальним. Як наслідок, маршрут буде недовантаженим, що в цілому буде означати недовантаження ланок, через які проходить цей маршрут.

Елементи результуючої матриці пропускних здатностей мережної топології отримуються в результаті додавання пропускних здатностей всіх маршрутів, що проходять через ребро, якому відповідає вибраний елемент матриці. Якщо у всіх елементах значення дорівнює номінальному значенню пропускної здатності, заданому в умові, прокладені маршрути вважаються оптимальними.

Для дослідження кості обслуговування потоків у мережах використовують матрицю затримок передавання по мережних каналах, з урахуванням маршрутів отриманих в результаті виконання кроків, описаних вище. Елементи цієї матриці визначаються як сума обернених значень пропускної здатності в кожній ланці. Недоліком такого підходу до формування матриці затримок, є те що основна

частина затримки спричинюється обробкою в мережних вузлах, і не настільки залежить від вільної пропускної здатності каналів. Пропускна здатність переводиться у відносні одиницю, значення якої коливається від 0 до 1. Те саме проводиться і для матриці пропускних здатностей.

Отримані матриці пропускних здатностей та затримок виступатимуть критеріями при оцінці мережних топологій. Використовуючи метод аналізу ієрархій формують ще одну матрицю, яка дає змогу визначити важливість критерії на основі порівняння. У роботі приймається, що обидва критерії є однаково важливими і порівняльна матриця не впливає на подальший результат.

Для того, щоб застосувати всі матриці-критерії для отримання єдиної оцінки, необхідно визначити вектор локального пріоритету (B_i), який обчислюється як середнє геометричне значення рядка матриці, яке нормується по відношенню до всіх інших пріоритетів рядків матриці. Кінцеву оцінку топології отримують з використання наступного виразу:

$$P = A_1 \cdot B_{(1,1)} + A_2 \cdot B_{(1,2)} + \ldots + A_n \cdot B_{(m,n)}, \qquad (3.2)$$

де P – оцінка структури, яка досліджується, m – номер матриці за одним із критеріїв, n – номер локального пріоритету матриці-критерію m.

3.1.2. Порівняльна оцінка варіантів структур інформаційно-телекомунікаційних систем

Для дослідження було обрано мережу, що складається із шести вузлів. Для такої кількості вузлів можна створити 30827 унікальних варіантів топологій, прте не кожна так топологія буде описувати реальну телекомунікаційну мережу. Для дослідження будуть вибрані лише зв’язні варіанти.

Для оцінки оптимальності вибраної топології використовуватиметься цільова функція (3.3). Ця функція враховує інтегральну оцінку графа, отриману в результаті застосування методу аналізу ієрархій. Враховуючи той факт, що значення затримки були отримані на основі пропускної здатності каналів, можна зробити висновок, що найкращим маршрутом буде той, для якого оцінка буде мати мінімальне значення.

$$\min(P), \tag{3.3}$$

В результаті моделювання отримано інтегральну оцінку для кожної топології. На Рис. 3.1 подано статистичний розподіл отриманих оцінок.

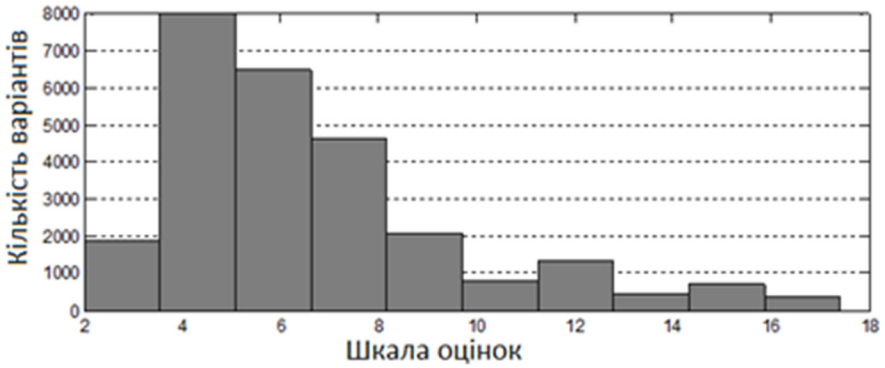


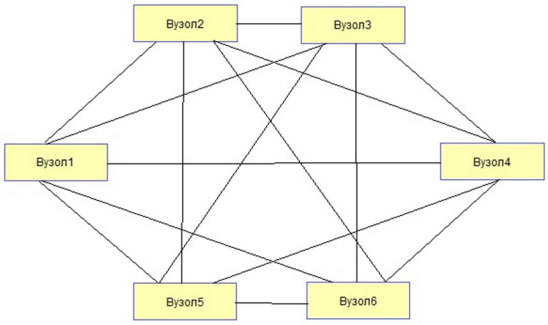
Рис. 3.1. Статистичний розподіл отриманих оцінок

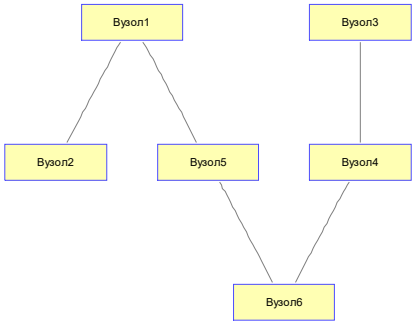
Для зручності аналізу отриманих результатів всі топології було розбито на 11 груп. Кількість топологій у кожній групі є різною, так само як і різними є затримки передавання (Таблиця 3.1).

Таблиця 3.1. Групи топологій

Номер групи	1	2	3	4	5	6	7	8	9	10	11
Кількість ребер	5	6	7	8	9	10	11	12	13	14	15
Загальне число конфігурацій	3003	5005	6435	6435	5005	3003	1365	455	105	15	1
Число зв’язних конфігурацій	1296	3660	5700	6165	4945	2997	1365	455	105	15	1

Одинадцята група містить одну повнозв’язному топологію (Рис. 3.2, а), в якій налічується 15 ребер. Для цієї топології отримана найнижча оцінка. Інтегральна оцінка зростає зі зростання кількості ребер і має найвище значення (17.409) для топології, що складається з 5 ребер, де вузли з’єднані послідовно. До першої групи в загальному входять топології, що складаються з 5 ребер. Найнижча оцінка (8.94) серед топологій першої групи отримала топологія "зірка".





а) б)

Рис. 3.2. Структура мережі з найнижчою (а)
та найвищою (б) інтегральною оцінкою

Розроблена модель дає можливість згенерувати всі можливі конфігурації топологій. Проте, для цього необхідно, щоб всі вузли були еквівалентними, а перестановка вузлів не впливає не топологію. Отже число конфігурацій для кожного варіанту топології дорівнює $A_N^1 = \frac{N!}{(N-1)!} = \frac{6!}{5!} = 6$. Для таких варіантів всі оцінки є рівними. В процесі дослідження зауважено, що однакові оцінки можуть бути у різних конфігурацій. Враховуючи неможливість графічного представлення всіх варіантів, в цій роботі вони не наводяться різні топології з рівною інтегральною оцінкою. В таблиці 3.2 зібрано максимальні та мінімальні оцінки всіх груп топологій та кількість унікальних оцінок в групі.

Таблиця 3.2. Отримані оцінки конфігурацій мережних структур

Номер групи	1	2	3	4	5	6	7	8	9	10	11
Число оцінок, які відрізняються	20	338	1103	1146	483	76	11	4	2	1	1
Максимум оцінки	17.41	13.21	10.08	7.58	5.61	4.36	3.61	2.98	2.53	2.2	2
Мінімум оцінки	8.94	7.66	6.21	4.82	4.13	3.6	3.14	2.78	2.44	2.2	2

Як випливає із аналізу таблиці (Таблиця 3.2), можна зробити висновок, що значення оцінок в групах прочинаючи з 7-ї і вище не перекриваються. Це пояснюється тим, що в цих конфігураціях є достатня кількість ребер та обрані маршрути шляхи мають мінімальну кількість спільних ланок. Всі інші групи мають невеликий перехідний діапазон оцінок, а тому топології з різною

кількістю ребер є еквівалентними, що є результатом упущення впливу функціональних параметрів вузлів.

3.1.3. Передавання інформації на основі аналізу функцій корисності у процесі надання послуг

Розрізняють пакети за класами та вузлами призначення. Допустимо, що є пакети типу К, і кожний тип призначений для вузла $a(N_k)$при $k = 1, \dots, K$.

Позначимо, що $U^k \in C_0$ є інтенсивністю вхідного потоку k -го типу пакетів у джерелах трафіку ІТС, а $deg(U^k)$ – це інтенсивність вихідного потоку пакетів, що виходить з мережного вузла $a(N_k)$, і нехай $T^k \in C_1$ є швидкість потоку у каналі зв'язку ІТС. Стійкість мережі визначається для $k = 1, \dots, K$:

$$\partial(T^k) + U^k - deg(U^k)a(N_k) = 0 \tag{3.4}$$

Це рівняння є наслідком закону збереження.

Вектори визначимо з К елементами C_1 таким чином::

$$C_1^K = \{ \sum_{m=1}^{M} f_m \, b(m) : \; f_m = (f_m^1, \dots, f_m^K), f_m^k \in \boldsymbol{R} \},$$

Представлення швидкості потоків всіх пакетів:

$$T = (T^1, \dots, T^k, \dots, T^k) \in C_1^K,$$

де

$$T^k = \sum_{i<j} T_{ij}^k \left(a(i), a(j)\right).$$

Позначимо, що $F: C_1^K \to R$ – функція вартості потоку пакетів по мережевих каналах зв'язку. Умовно скажемо, що F є додатня, опукла і диференційна функція. Локальна вартість потоку пакетів в каналі досить часто є предметом порівняття із даними про стан мережних потоків, які утримуються локально; в протилежному разі, між вузлами мережі доведеться передавати більше службової інформації [286]. Допускаємо, що вартість каналу залежить тільки від величини потоку і зростає відповідно до величини потоку. Окрім цього, не передбачено передавання потоки пакетів однакового типу у зустрічних напрямках по каналу зв'язку ІТС, адже це створить непотрібні цикли

маршрутизації; але канал зв'язку може мати різні напрямки потоків пакетів різних типів [277].

В алгебраїчно-топологічній структурі описана функція витрат яку розглянемо наступним чином. Для кожного каналу задаємо напрямок, і припускаємо, що напрямок каналу є) з $i < j$. Функцію вартості F визначається у відповідності до величини потоку, прирівнюємо $F(T_{ij}^k)$ і $F(T_{ji}^k)$, тобто

$$F\left(T_{ij}^k\right) = F\left(T_{ji}^k\right) = F(\left|T_{ij}^k\right|).$$

Хоча , потрібно розрізняти похідні від $F(T)$ по T_{ij}^k і T_{ji}^k . При $i < j$ позначимо похідну F по T_{ij}^k як $\frac{\partial F}{\partial T_{ij}^k}$ і похідна F по T_{ji}^k як $\frac{-\partial F}{\partial T_{ij}^k}$ таким чином, що

$$\frac{\partial F}{\partial T_{ij}^k} = -\frac{\partial F}{\partial T_{ji}^k}$$

Отже це означає, що $\frac{\partial F}{\partial T_{ij}^k}$, пов'язана з потоками типу k від вузла $a(i)$ до вузла $a(j)$, рівна $\frac{-\partial F}{\partial T_{ji}^k}$, яка пов'язана з "віртуальним" зворотнім потоком від $a(j)$ до $a(i)$. Відзначимо, що зростання F залежить від збільшення інтенсивності потоку, для прямого потоку типу k по каналу $(a(i), a(j))$, похідна першого порядку $\frac{\partial F}{\partial T_{ij}^k}$ повинна бути додатня, і $\frac{\partial F}{\partial T_{ji}^k}$ повинна бути від'ємна. Для $k = 1, \dots, K$, позначимо градієнт F в C_1 буде

$$\frac{dF}{dT^k} = \sum_{i<j} \frac{dF}{dT_{ij}^k}(a(i), a(j)).$$

В задачі оптимальної маршрутизації [27] для мережі передачі даних, $F(T)$ стандартного вигляду можна записати як

$$\sum_{(ij)} \frac{\sum_{k=1}^K \left|T_{ij}^k\right|}{C_{ij} - \sum_{k=1}^K \left|T_{ij}^k\right|}, \tag{3.5}$$

де C_{ij} є пропускною здатністю каналу (i, j), яка визначається середнім числом пакетів в кожному вузлі, які знаходяться в черзі типу $M/M/1$.

Формування завдання оптимізації робимо в такий спосіб:

$$Min\ F(T), \tag{3.6}$$

враховуючи

$$U^k \partial(T^k) - deg(U^k)\, a(N_k) = 0, \forall k = 1, \dots, K. \tag{3.7}$$

Завдання оптимізації, яке було представлене вище, може бути розв'язане за допомогою розрахунку градієнта, використовуючи $\frac{dF}{dT^k}(T(t))$. В обчисленнях, величина $\frac{dF}{dT^k}(T(t))$ має бути обчислена локально; інакше необхідно передавати значну кількість інформації про $\frac{dF}{dT^k}(T(t))$ між мережними вузлами. Сукупна затримка по каналах зв'язку є однією з найпоширеніших функції витрат. Цей тип функції затримки є підсумовуванням затримок для різних класів трафіку у різних каналах, що означає, в контексті даної структури, що градієнт $\frac{dF}{dT^k}(T(t))$ є незалежним від процесів у сусідніх каналах, однак в межах одного каналу можливі взаємодії потоків між собою.

Завдання передавання інформаційного потоку [28], яке сформульване вище, суттєво відрізняється від класичних. Оптимізуємо вартість з урахуванням інтенсивностей потоків між заданими парами джерело-призначення по наперед визначених шляхах .Матрицею шляхів описуємо топологію мережі, в якій запис (i, j) становить одиницю, коли канал i є елементом шляху j; і нуль – у протилежному разі.

Передавання мультисервісного потоку сформулюємо як завдання у просторі, який визначається множиною каналів зв'язку .Це забезпечує мережним вузлам інформацію лише про агреговану інтенсивність вхідних потоків, у яких спільний кінцевий вузол, та інтенсивність потоків одного класу. Відзначимо, що маршрути передавання і вхідні вузли інформаційних потоків при цьому залишаються невіомимим. З використанням граничного оператора, ∂ формулюється закон збереження пакетів. До переваг цього підходу слід віднести те, що хід оптимізація обмежується значно меншим простором станів у порівнянні із стандартним формулюванням у просторі, що визначається множиною шляхів. Як приклад, проведемо аналіз мережі розмірністю N вузлів,

яка складається з $N\,(N-1)/2$ можливих пар джерело-призначення і кожна з тих пар може бути з'єднана багатьма шляхами. За умови, що жоден шлях попередньо не визначений, розмірність задачі обмежена у $N\,(N-1)/2$. Даний підхід забезпечить підвищення адекватності методики формування інформаційних потоків через мережу для передавання в режимі без встановлення з'єднання, оскільки попереднє визначення шляху у цьому разі не вимагається.

Методи лінійного програмування можуть бути використані для знаходження розв'язку, якщо вибирати лінійні функції вартості для F(T) в (3.6). З лінійної функції вартості випливає розв'язок, аналогічний використанню OSPF протоколу. Але, звернемо увагу на квадратичні функції вартості. Оптимальні розв'язки у випадку їх використання є унікальними і можуть бути отримані у замкнутому режимі, що є беумовною перевагою квадратичних функцій вартості.

Як конструктивний параметр, можна розглядати параметр B (вага каналу), який може підлаштувати інженер-конструктор, щоб досягти оптимальної маршрутизації за різними властивостями. Як правило, проблеми практичного проектування мережі не підлягають зведенню до оптимізації спільної функції вартості. Як приклад, недостатньо просто мінімізувати затримку або імовірність відкидання пакетів. Доволі часто на практиці застосовують методики, в основі яких є інтегральні функції вартості, цільові параметри яких мають власну вагу, а це дає змогу збалансувати багатокритеріальне завдання оптимізації.

Момент на який потрібно звернути увагу, відзначається тим, що до одношляхової маршрутизації зводяться випадки застосування лінійних функцій вартості (OSPF як приклад), а от функції вищих порядків і квадратичні функції зокрема можуть юути використані у випадку багатошляхової маршрутизації, що додатково балансує у ІТС. Це, в кінцевому венсі, означає що вибір найкращої функції вартості здебільшого залежить завантаженості ІТС. Краще використовувати лінійну функцію *F(T)*, коли завантаження мале, але, коли затримки досягають критичної межі при зростi навантаження, краще використовувати квадратичну функцію або функцію більш високого порядку.

Від цілей системи, залежить середовища функціонування та вибір найкращої функції вартості і вагових коефіцієнтів мережних ресурсів.

Оптимальний розв'язок в закритій формі дає квадратична функція вартості каналу, представлена алгебраїчно-топологічним виразом. Однак, оптимальний розв'язок забезпечує багатошляхова маршрутизація, який може бути непридатним в деяких реальних формулюваннях завдання.

Нехай маємо пакети K типів, що розрізняються за класами, а також за цільовими вузлами. Моделлю мережі є зважений граф, у якому кожен канал $(a(i), a(j))$ щодо пакетів K-го типу є зваженим по B_{ij}^k враховуючи що $B_{ij}^k = B_{ji}^k$, і B_{ij}^k є рівне нулю, якщо не існує каналу зв'язку між $a(i)$ і $a(j)$. Вага B_{ij}^k як правило є параметром, який визначає продуктивність, наприклад, це може бути затримка або доступна пропускну здатність.

Завдання оптимізації з квадратичною функцією вартості формулюється таким чином:

$$\text{Min } F(T) = \sum_{k=1}^{K} \sum_{i<j} \left(\frac{T_{ij}^k}{B_{ij}^k}\right)^2 \text{ по T,} \tag{3.8}$$

враховуючи

$$\partial(T^k) + U^k - de\, g(U^k)\, a(N_k) = 0, \forall k = 1, \dots, K.$$

Через $\frac{T^k}{B^k} = \sum_{i<j} \frac{T_{ij}^k}{B_{ij}^k}(a(i), a(j))$, позначимо праву частину.

Завдання оптимізації (3.8) розв’язуємо з урахуванням того, що

$$\left\|\frac{T^k}{B^k}\right\|^2 = \left\|\sum_{i<j} \frac{T_{ij}^k}{B_{ij}^k}(a(i), a(j))\right\|^2 =< \delta_{B^k}(X^k) + z_{B^k}, \delta_{B^k}(X^k) + z_{B^k} >_1$$

$$=< \delta_{B^k}(X^k), \delta_{B^k}(X^k) >_1 +< z_{B^k}, z_{B^k} >_1 \geq \delta_{B^k}(X^k), \delta_{B^k}(X^k) >_1.$$

Бачимо, що мінімум досягається, коли справедливо наступне

$$\frac{T^k}{B^k} = \delta_{B^k}(A_{B^k}(de\, g(U^k)\, a(N_k) - U^k)).$$

Отримуємо оптимальний розв'язок сформульованого завдання

$$\hat{T}^k = B^k .* \delta_{B^k}\left(A_{B^k}(de\, g(U^k)\, a(N_k) - U^k)\right), \forall k = 1, \dots, K, \qquad (3.9)$$

В отриманому розв'язку (3.9) A_{B^k} є єдиним оператором, що має бути доступним глобально, і він є єдиним незворотнім відображенням $\text{А}_\text{В}$: К ↦ К.

Динамічна зміна швидкості інформаційних потоків, що призначена для компенсації коефіцієнта варіації другого порядку вхідного трафіку, може бути реалізована на базі способу керування трафіком по замкнутому контуру. Цей спосіб, який базується на лінійному законі керування, є динамічною інверсією. Отже, слід визначити міру підстроювання інтенсивностей інформаційних потоків. Він має кілька властивостей. Пункт один, скориговані розв'язки маршрутизації теж створюють ациклічні потоки. Пункт два, управління по замкнутому контуру зберігає стабільність станів мережі. Пункт третій, розподілений алгоритм можна застосувати для керування трафіком по замкнутому контуру, тим самим забезпечуючи вузлам можливість оновлення таблиці маршрутизації автономно. Це спряє балансуванню ступеня завантаженості черг між вузлами мережі.

Далі розглянемо коректування інтенсивностей інформаційних потоків $\bar{T}^k$, де стаціонарним станом розв'язку задачі маршрутизації є $\bar{T}^k$, який отримано у попередньому підрозділі. По замкнутому контуру управління обмежується підмножиною мережних каналів $D_1 \subset C_1$, де D_1 формується з $\bar{T}^k$, за вийнятком каналів до вузла-призначення $a(N_k)$, отже формується набір каналів для передавання k-типу потоку пакетів бе урахування каналу до вузла-призначення. Через D_0, позначимо підмножину вузлів, інцедентних D_1 де обслуговуються пакети k-типу, за винятком цільового вузла $a(N_k)$.

Очікувану довжину черги пакетів потоку k - типу у вузлі позначимо $\bar{Q}^k(i)$, $a(i) \in D_0$. Дані позначеннямають місце, коли система перебуває у стаціонарному стані. Ця величина може обмежується або розміром буфера, або рівна середній довжині черги у залежності від обраної моделі системи, і залежить від T^k. Звернемо увагу, що вибір параметру $\bar{Q}^k(i)$ здійснюється на етапі мережного

проектування, наприклад, для мінімізації тривалості обслуговування або ймовірність відмови в обслуговуванні.

Відповідно до (3.14), контролер замкнутого контору буде керуватись станом черги. З точки зору мінімізації завантаження контролера, балансування навантаження важливе для контролю варіації другого порядку трафіку на вході контролера.

З наступного динамічного рівняння інверсії над D_0, розроблена команда управління

$$\frac{dQ^k}{dt}(t) = -\epsilon_k L_B(Q^k(t)/\bar{Q}^k, \tag{3.10}$$

$a(j) \notin D_0$ для деяких $\epsilon_k > 0$, що є постійною, яка вказує на ступінь збіжності алгритму до стаціонарного розв'язку. Приймемо, що у вузлах справедлива тотожність

$$\frac{dQ^k}{dt}(t) \equiv 0$$

Отримана система керування визначена як

$$\frac{dQ^k}{dt} = -\epsilon_k L_B(Q^k/\bar{Q}^k) \in C_0.$$

У цьому випадку динаміка системи в часі

$$\frac{dQ^k}{dt} = U^k - deg(U^k)\, a(N_k) + \partial(T^k) \tag{3.11}$$

та

$$\frac{dQ^k}{dt} = -\epsilon_k L_B(Q^k/\bar{Q}^k), \tag{3.12}$$

де (3.11) випливає із закону збереження. Праві сторони (3.11) і (3.12) рівні між собою, тому динаміка системи може бути отримана на основі такого рівняння

$$\epsilon_k L_B\left(\frac{Q^k}{\bar{Q}^k}\right) + U^k - deg(U^k)\, a(N_k) + \partial(T^k) = 0.$$

Дане вище рівняння може бути розв'язане в явному вигляді для T^k, і його розв'язок

$$T^k = B^k .* \delta_{B^k} A_B (de\, g(U^k)\, a(N_k) - U^k) + \sum_{l=1}^{L} c_l\, z_l + \epsilon_k B .* \delta_B (Q^k / \bar{Q}^k).$$

Коригування інформаційних потоків позначимо як

$$\Delta T^k = \epsilon_k B .* \delta_B (Q^k / \bar{Q}^k).$$

Рівняння управління (3.10) має наступні властивості:

– оскільки $\deg(\epsilon_k L_B (Q^k(t)/\bar{Q}^k)) = 0$ при будь-яких $Q^k(t)$, цільову динаміку може забезпечити ΔT^k, що забезпечує виконання $\partial(\Delta T^k) = -\epsilon_k L_B (Q^k/\bar{Q}^k)$. Зміна інтенсивності сервісних потоків у рівнянні динаміки (3.11) досягається, коли

$$\Delta T^k = \epsilon_k B .* \delta_B A_B L_B (Q^k(t)/\bar{Q}^k)) = \epsilon_k B .* \delta_B (Q^k(t)/\bar{Q}^k)), \qquad (3.13)$$

яке вираховується тільки локальними операторами і потребує лише локальної інформації;

– рівняння динаміки призначене для балансування черг для того, щоб у мережних вузлах досягти рівномірного завантаження буферів, так як власний підпростір власного значення нуля L_B є одновимірним, який є $\sum_{i=1}^{N} fa(i)$ для $f \in R$. По-іншому, буде здійснюватися динамічне керування станом черги у вузлах $\in D_0$

$$\frac{Q(i)}{\bar{Q}(i)} = \frac{Q(j)}{\bar{Q}(j)} \qquad (3.14)$$

у вузлах $a(i) \neq a(j)$. Якщо черги не перебувають у стані балансу, то відповідно до (3.12), алгоритм займається їх балансуванням. Коли черги перебувають у стані балансу, то керування не відбувається і справедливе співвідношення (3.4);

– оскільки новий розв'язок задачі маршрутизації є образом δ_B, він залишається ациклічним;

– отриманий розв'язок сформульованого завдання маршрутизації $\Delta T^k + \bar{T}^k$, продовжує задовольняти рівняння (3.4), що визначає стійкість мережі.

За допомогою розподілених алгоритмів може бути сформована теорія мережі. Алгоритм полягає у обчисленні стійких розв'язків завдання маршрутизації, які забезпечують баланс завантаженості черг між вузлами ІТС.

Стійкий розв'язок завдання маршрутизації буде оптимальним, коли

$$\hat{T}^k = \lim_{k\to\infty} B^k .* \delta_{B^k} \left(\sum_{j=0}^{k} (I - L'_{B^k})^j D^{-1} (de\, g(U^k)\, a(N_k) - U^k)\right) \quad (3.15)$$

Цей розподілений алгоритм пошук оптимального розв’язку ітеративном способом. Умовою здійснення ітерації є те, що $deg(U^k)$ повинен бути відомим його призначеному вузлу $a(N_k)$. Тобто, ця інформація має бути передана у вигляді службового кадру перед початком ітеративного процесу.

Кожен вузол має здійснити такі етапи для отримання стаціонарного розв'язку завдання маршрутизації для пакетів k-типу.

1. В момент початку роботи алгоритму потенціал $P(0) = -D^{-1}(de\, g(U^k)\, a(N_k) - U^k)$.

2. У наступних ітераціях h, потенціал $P(h)$ обчислюється як

$$P(h) = P(0) + \left(I - L'_{B^k}\right)P(h-1).$$

3. Для забезпечення сходимості алгоритму використаємо наступне. Через $P(i,h)$ позначимо значення, яке отримали під час виконання кроку 2 для вузла $a(i)$ в момент часу h. Вузол $a(i)$ здійснює розрахунок потоку для всіх суміжних каналів каналів $(a(i), a(j))$, тобто

$$T_{ij}(h) = (B_{ij}^k)^2 (p(j,*) - P(i,h)),$$

де $P(j,*)$ є останньою величиною, яку отримано на інтервалі часу h для вузла $a(i)$. Отримане вище співвідношення узгоджується з (3.15). Це означає, що алгоритм сходиться для всіх каналів $(a(i), a(j))$.

$$\left|T_{ij}(h+1) - T_{ij}(h)\right| \le \epsilon,$$

при $\epsilon > 0$.

У даному алгоритмі, кожний з мережних вузлів обмінється значеннями потенціалів P із сусідами. Ці отримані значення є підставою для обчислення значення потенціалу даного вузла. Для здійснення цих операцій не вимагається синронності між вузлами мережі.

У випадку квадратичної функції вартості, розв'язок маршрутизації $\hat{T}^k$, часто означає розв'язок багатошляхової маршрутизації. Реалізація цієї багатошляхової маршрутизації описуються наступним чином. Вузол $a(i)$ аналізує інтенсивність потоку до всіх наступних вузлів $a(j) \in N(a(i)$, і визначає, яка частка цього потоку передаватиметься по суміжних до нього каналах. Відношення інтенсивності потоку даного вузла до сумарної інтенсивності потоку- це частка до конкретного наступного вузла, вона передається всім наступним вузлам $a(j) \in N(a(i)$. Після обчислення частки вузол $a(i)$ надсилає пакети різних вузлів пропорційно на основі зваженого циклічного чи імовірнісного методу передавання.

Коли розмір мережі зростає,це може стати проблемою тому що і кількість шляхів зростає. Може бути враховане ієрархічне обмеження на домен мережі так, що підмережі визначатимуть маршрути передавання для потоків одного класу. Це досягається шляхом введення додаткових граничних умов.

Не дивлячись на можливість використання багатошляхової маршрутизації, в алгоритмі немає обов’язкового обмеження на порядок наскрізної доставки потоку пакетів. Щоб забезпечити наскрізну впорядковану доставку потоку пакетів потрібно внести зміни на мережевому рівні. Може бути здійснена наскрізна впорядкована доставка потоку пакетів, з урахуваннння того, що пакети виділяються на маршрутизаторах із використанням деякого ідентифікатора потоку. Іншою обов’язковою умовою є те, що маршрутизатором організовано передавання пакетів одного потоку не більше ніж до одного суміжного маршрутизатора.

В вузлах мережі можуть накопичуватись черги з різних причин. Це визначається як варіацією вхідного трафіку у ІТС, так і обмеженою пропускною здатністю каналів зв'язку. На основі рівняння (3.13), керування по замкнутому контуру регулює регулює розв'язок задачі маршрутизації

$$\Delta T^k = \epsilon_k B .* \delta_B A_B L_B(Q^k(t)/\bar{Q}^k)) = \epsilon_k B .* \delta_B(Q^k(t)/\bar{Q}^k)),$$

Необхідний алгоритм для обчислення коригування розв'язку завдання динамічної маршрутизації можна визначити так:

1. Вузлом періодично або в певні визначені моменти фіксуються довжини черг. Визначається співвідношення фактичної і очікуваної черг $Q/\bar{Q}$.

2 Інформація про співвідношення $Q/\bar{Q}$ має бути передана між вузлами-сусідами.

3. Вузол $a(i)$ розраховує коригування розв’язку задавдання маршрутизації за рівнянням (3.13). Розподіленість алгоритму забезпечується у рівнянні (3.13) тим, що до нього входять тільки локальні оператори, які пересилаються між вузлами-сусідами.

4. Вузли здійснюють зміни у власних таблицях на основі інформації про коригування розв’язку задавдання маршрутизації.

Сформульовано розподілений алгоритм, який не потребує якої-небудь глобальної інформацію. Інформація про стан черг передається між сусідніми вузлами. Даний алгоритм виконується частіше, ніж для стаціонарного розв'язку маршрутизації.

На Рис. 3.3 подано граф мережі, який складається з 10 вузлів. Інтенсивності вхідних потоків у вузлах такі:

- вузол 1 – 2 у. о.;
- вузол 2 – 3 у. о.;
- вузол 4 – 5 у. о.

Інтенсивність потоку на виході із вузлі-призначення 8 становить 10 у. о.

Канали є симетричними, тобто, якщо пропускна здатність каналу між вузлами 1 і 2 складає 10 одиниць, то і пропускна здатність каналу між вузлами 2 і 1 теж становить 10 одиниць. Початкові значення потенціалів такі:

- вузол 1 – 2 у. о.;
- вузол 3 – 3 у. о.;
- вузол 4 – 5 у.о.;
- вузол 8 – 10 у. о.;
- решта вузлів – 0 у.о.

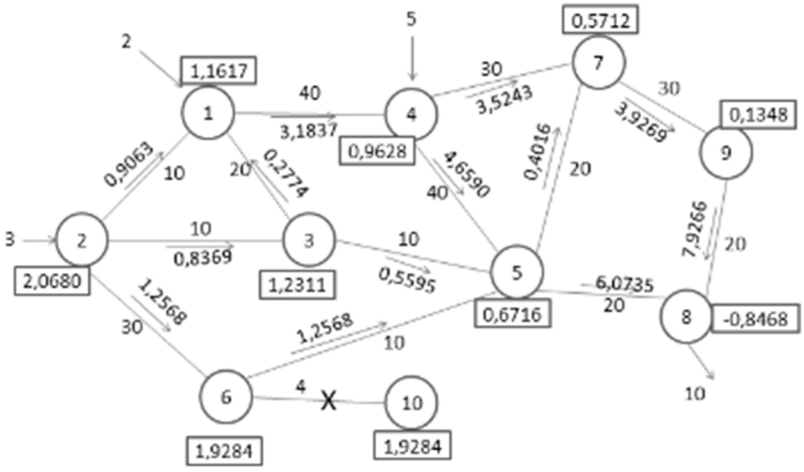


Рис. 3.3. Мережа з 10-ма вузлами з ациклічною маршрутизацією

Як наслідок, на Рис. 3.3 зображено потенціали на кожному вузлі після досягнення збіжності алгоритмом. Виділимо, що за підсумками роботи алгоритму високі потенціали отримали вхідні вузли, на які надходили інформаційні потоки (1, 2 і 4). У свою чергу, найнижчий потенціал отримав вузол-призначення (8).

У представленому методі передавання інформації потік даних передається по каналу зв’язку у напрямку від вузла з високим потенціалом до вузла з низьким потенціалом. Тобто умовна інтенсивність потоку може бути визначена через різницю потенціалів, розділену на ваговий коефіцієнт каналу.

Алгоритм передбачає, що потік між вузлами з однаковим потенціалом не існує (наприклад, канал 10-6). Це пояснюється тим, що у логічній топології мережі не можуть існувати петлі (цикли). Визначено, що алгоритм збіжний із точністю 10^{-5}, а кількість ітерацій, необхідна для досягнення збіжності, становить 187.

3.2. Метод класифікації інформаційно-комунікаційних послуг з урахуванням рівня підписки користувачів та метрик якості сервісу

Випадкові величини позначимо великими літерами. Ті самі малі літери позначатимуть відомі величини.

Об'єм потоку визначаємо з максимального рівня підписки та тривалості потоку. Отже, клієнт i в класі k з максимально можливим рівнем підписки

$S_{i,k} \sim F_{S_k}$ і потоком з тривалістю $D_{i,k} \sim F_{D_k}$ має розмір потоку $V_{i,k} = S_{i,k} D_{i,k}$ [біт], який отриманий унаслідок кодування на верхньому рівні.

Архітектуру каналу зв'язку розглянемо з декількома класами обслуговування, де кожен клас обслуговування відрізняється різним рівнем забезпечення миттєвого нормалізованого рівня підписки для всіх потоків певного класу. Канал надає набір класів обслуговування *K*, які характеризуються числом $(\alpha_k, k = 1, \dots, K)$ при $\alpha_k \in (0,1)$. Нехай $0 < \alpha_1 < \cdots < \alpha_K < 1$. Клієнту i в класі k гарантується, що в момент часу t він отримає миттєвий рівень підписки $S_{i,k}(t)$, таким чином, що $S_{i,k}(t) \geq \alpha_K S_{i,k}$, де $S_{i,k}$ – максимальний рівень підписки, запропонований надавачем послуг. Нормалізований миттєвий рівень $\frac{S_{i,k}(t)}{S_{i,k}}$ повнен гарантовано перевищувати клас k QoS, що гарантує α_k.

Щоб краще забезпечети QoS, використовують класи [289]. Клас обслуговування α_k, призначений для встановлення відповідності між фактичними вимогами користувача до якості обслуговування та технічними параметрами мережі і ресурсами, які слід виділити для надання сервісу. Вектор-стовпець середніх інтенсивностей поступлення пакетів $(\lambda_k, k = 1, \dots, K)$ визначає попит на послуги. Послуги різних класів утворюють потоки з відмінними властивостями. У рамкамках моделі послуги класифіковано за максимальним рівнем підписки і за тривалістю потоку. Зокрема, для потоків класу k, F_{S_k} є функцією розподілу рівня підписки, а F_{D_k} визначає функцією розподілу тривалості потоку. Відзначимо, що оптимальне кодування визначаж максимальний рівень підписки, шо пропонується надавачем послуг. Визначаємо середні значення цих рівнів підписки як $\sigma_k = E[S_k]$ і $\delta_k = E[D_k]$.

Нехай для кожного потоку максимальний рівень підписки та тривалість потоку – незалежні випадкові величини. Розподіл розміру потоку класу k позначимо через F_{V_k}, так, що

$$F_{V_k}(v) = \int_0^\infty F_{S_k}(\frac{v}{d}) dF_{D_k}(d) = \int_0^\infty F_{D_k}(\frac{v}{s} dF_{S_k}(s) \qquad (3.16)$$

Нехай F_X – функція розподілу довільної випадкової величини X, для якої слід визначити випадкову величину $\hat{X}$ з функцією розподілу $F_{\hat{X}}(x) = \frac{1}{E[X]}\int_0^x \omega dF_X(\omega)$. Функція розподілу відповідає визначає розподіл потоків за обсягом і тривалістю, коли система розглядаємо в звичайному масштабі часу. Існує вища імовірність спостерігати потоки більшої тривалості, ніж короткотривалі потоки. Розмір потоку є функцією від тривалості потоку. Нехай $\widehat{D}_k$ і $\widehat{V}_k$ – тривалість і розмір потоку класу k у звичайному масштабі часу, враховуючи, що $\widehat{D}_k \sim F_{\widehat{D}_k}$ і $\widehat{V}_k \sim F_{\widehat{V}_k}$.

Нехай кожен надавач послуг реалізує дискретну множину можливих рівнів підписки S. Нехай S є максимальним рівнем підписки, тоді множина доступних рівнів підписки визначається множиною класів послуг, які визначені каналом передавання, тобто $S = (\alpha_1 S, \dots, \alpha_K S, S)$. Отже, користувач *i* цього потоку, якому надається клас обслуговування k', забезпечується миттєвим рівнем підписки $S_{ik'}(t) \in (\alpha_k S, k = k', \dots, K) \cup S$. Відповідно, користувачу надається обслуговування рівня $\alpha_{k'}$ або краще. Набір значень $\alpha_k S,\ k = k', \dots, K \cup S$ є множиною рівнів обслуговування, які не гірші за необхідну якість сервісу $\alpha_{k'}$.

Нехай процес адаптації каналу π визначає зв'язок інформаційних потоків із миттєвими значенням рівнів підписки з урахуванням таких умов-обмежень: визначений рівень підписки гарантується надавачем послуг; визначений рівень підписки достатній для користувача; загальний рівень підписки не більший за пропускну здатність каналу зв'язку.

Нехай метрикою QoS процесу адаптації є $E^0[Q^\pi]$:

$$E^0[Q^\pi] = E^0[\frac{1}{D}\int_0^D \frac{S^\pi(t)}{S} dt] \tag{3.17}$$

де $E^0[\cdot]$ – середнє значення тривалості очікування користувача

$$E^0[Q] = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n \int_{-\infty}^{\infty} Q_i(t)dt,$$

де $Q_i(t)$ – QoS, надана користувачеві *i* в момент *t*, коли *i*-ий користувач не перебуває в системі, $Q_i(t)$ приймається рівною 0. $S^\pi(t)$ показує, що миттєвий

рівень підписки буде залежати від процесу адаптації. Q^{π} – середній нормалізований рівень підписки. Відзначимо, що користувач i класу послуг k з максимальним рівнем підписки $S_{i,k}$ буде обслужений з QoS $Q_{i,k}^{\pi} \in [\alpha_k, 1]$. Отже, ступінь забезпечення якості обслуговування може бути оцінений як відношення загальної кількості інформації, яку отримав користувач, $\int_0^{D_{i,k}} S_{i,k}^{\pi}(t)$, до загальної кількості інформації, яка може бути надана з максимальним рівнем підписки $S_{i,k}D_{i,k}$.

Адаптація інтенсивності потоку є способом досягнення балансу між високою якістю обслуговування і низькою ймовірністю блокування потоків. Використання адаптивного кодування інформаційних потоків для контролю перевантаження забезпечує зниження імовірності блокування. Зниження рівня підписки таким чином у процесі перевантаження дає змогу знайти ресурси для обслуговування нових інформаційних потоків. Представимо, що канал обслуговує максимальну кількість потоків, одночасно забезпечуючи необхідний рівень якості обслуговування для інформаційних потоків, які вже отримують сервіс. Звідси випливає, що потік класу k' з максимальним рівнем підписки *s,* допущений до передавання у каналі зв'язку, повинен забезпечувати виконання умови

$$\sum_{k=1}^{K} \sum_{i=1}^{n_k(t)} \alpha_k \, s_{i,k} = \alpha_{k'} s \leq c, \tag{3.18}$$

де c – пропускна здатність, $n(t) = (n_k(t), k = 1, \dots, K)$ – кількість активних потоків в кожному класі обслуговування.

Архітектура каналу з множиною класів обслуговування складається з таких елементів: інтенсивність надходження, λ_k; клас обслуговування, α_k; густина розподілу максимального рівня підписки F_{S_k} із середнім значенням $E[S_k] = \sigma_k$, густина розподілу тривалості потоку F_{D_k} із середнім значенням $E[D_k] = \delta_k$. Отже, кожен клас обслуговування пов'язаний з: розподілом розміру потоку F_{V_k}; розподілом обсягу типового розміру потоку $F_{\widehat{V}_k}$, похідного від F_{V_k}; розподілом

тривалості типового потоку $F_{\widehat{D}_k}$, похідного від F_{D_k}. Користувачеві надається змінний в часі миттєвий рівень підписки $S_{i,k}^{\pi}(t)$, а його надання залежить від процесу адаптації π, набору рівнів підписки, що забезпечується надавачем послуг, а також ступінь забезпечення якості класом обслуговування користувача. QoS, надана користувачеві, $Q_{i,k}^{\pi}$, є середнім нормованим рівень підписки, і слід визначити процес адаптації π, який дає змогу максимізувати усереднену по користувачах QoS $E^0[Q^{\pi}]$.

3.2.1. Розподіл ємності інформаційно-телекомунікаційної системи у процесі надання інформаційно-телекомунікаційних послуг

Припустимо, що задано вектор випадкових величин, де $N(t) = (N_k(t), k = 1, \dots, K)$ – кількість потоків в момент часу *t* кожного класу обслуговування. Мінімальні і максимальні навантаження для класу *k* становлять $\underline{p}_k = (\lambda_k \delta_k)(\alpha_k \sigma_k)$ і $\bar{p}_k = (\lambda_k \delta_k)\sigma_k$. Отже, мінімальне навантаження для класу *k* визначається добутком середньої кількості потоків класу *k*, тобто, $E[N_k(t)] = \lambda_k \delta_k$ (в режимі низького блокування), і середнього мінімального рівня підписки $\alpha_k \sigma_k$ класу *k*. Точно таким способом визначимо максимальне навантаження. Загальні мінімальні і максимальні навантаження позначимо як $\underline{\rho} = \sum_{k=1}^{K} \underline{\rho}_k$ і $\bar{\rho} = \sum_{k=1}^{K} \bar{\rho}_k$. Загальну адаптивність визначаємо відношенням загального мінімального навантаження до загального максимального навантаження $\alpha = \underline{\rho}/\bar{\rho}$.

Послідовність вузлів з номерами *m,* для яких застосуємо процес адаптації, буде розглянуто далі. Вектор-стовпець інтенсивностей поступлення на *m*-й вузол визначається як $\lambda(m) = (\lambda_k(m), k = 1, \dots, K)$ і $\lambda_k(m) = m\lambda_k$. Нехай класове і загальне мінімальне і максимальне навантаження на *m*-й вузол позначимо як $\underline{\rho}_k(m)$, $\bar{\rho}_k(m)$, $\underline{\rho}(m)$, $\bar{\rho}(m)$. Відзначимо, що загальна адаптивність α не залежить від *m*. Тому, пропускна здатність *m*-го каналу становить $c(m) = \gamma \bar{\rho}(m)$ для деяких $\gamma > 0$.

Визначимо три режими масштабування пропускної здатності, які залежать від загальної адаптивності α й параметру γ: перевантажений режим, коли $\gamma < \alpha$; режим адаптації, коли $\alpha \leq \gamma \leq 1$; недовантажений режим, коли $\gamma > 1$. Для перевантаженого режиму характерна недостатня пропускна здатність для оброблення загального мінімального навантаження, тобто $c(m) = \gamma\bar{p}(m) \leq \alpha\bar{p}(m) = \underline{p}(m)$. Для недозавантаженого режиму характерно, що пропускна здатність перевищує загальне максимальне навантаження, тобто, $c(m) = \gamma\bar{p}(m) > \bar{p}(m)$. У режимі адаптації надана пропускна здатність знаходиться в межах між загальним мінімальним і максимальним навантаженням.

3.2.2. Максимізація якості надання різнокласових інформаційно-телекомунікаційних послуг

Для деяких класів обслуговування розглянемо процес адаптації каналу [166, 167]. Основа полягає в тому, щоб максимізувати узагальнений усереднений по користувачах і по часу нормалізований рівень підписки $E^0|Q|$. Оптимальний процес адаптації полягає у сортуванні всіх потоків за розміром і наданні максимального рівня підписки максиально можливій кількості потоків малого розміру із дотриманням гарантій на мінімальний рівень підписки для потоків кожного класу із заданим обмеженням величини пропускної здатності каналу. Позначимо, що π_m – оптимальна багатокласова адаптивна політика. Нехай $n(t) = \sum_{k=1}^{K} n_k(t)$ – кількість потоків у вузлі ІКС. Миттєвий рівень підписки для оптимального багатокласового процесу адаптації буде становити $s^{\pi_m}(t) = (s_{i,k}^{\pi_m}(t), i = 1 \dots, n_k(t), k = 1, \dots, K)$. Проте, тривалості всіх потоків мають бути відомими, тобто $D_{i,k} = d_{i,k}$ для $i = 1, \dots, n_k(t)$ і $k = 1, \dots, K$.

Оптимальний процес багатокласової адаптації, π_m, який максимізує нормалізований рівень підписки, $E^0|Q|$, є набором $s^{\pi_m}(t) = \left(s_{i,k}^{\pi_m}(t), i = 1 \dots, n_k(t), k = 1, \dots, K\right)$ в кожен момент *t*, який є результатом розування задачі цілочисельного програмування:

$$\max_{s(t)} q_{agg}(t) = \sum_{k=1}^{K} \sum_{i=1}^{n_k(t)} \frac{s_{i,k}(t)}{s_{i,k} d_{i,k}} \tag{3.19}$$

коли

$$\sum_{k=1}^{K}\sum_{i=1}^{n_k(t)} s_{i,k}(t) \le c,$$

$$s_{i,k}(t) \in (\alpha_l s_{i,k}, l = k, \dots, K) \cup s_{i,k},$$

$$\forall i = 1, \dots, n_k(t), k = 1, \dots, K.$$

Отже, існує процес багатокласової оптимальної адаптації $\tilde{\pi}_m$ з миттєвими значеннями $s^{\tilde{\pi}_m}(t)$ з $s_{i,k}^{\tilde{\pi}_m}(t) \in \{\alpha_k s_{i,k}, s_{i,k}\}$ для $i = 1, \dots n_k(t)$ і $k = 1, \dots, K$. Нехай $q^{\pi_m}(t)$ і $q\tilde{\pi}_m$ є QoS для процесів адаптації π_m і $\tilde{\pi}_m$. Тоді різниця між цими розподілами прямує до нуля, коли $n(t) \to \infty$ і

$$\frac{q^{\pi_m}(t) - q^{\bar{\pi}_m}(t)}{q^{\pi_m}(t)} \le \frac{k_m}{n(t)}, \tag{3.20}$$

при $k_m < \infty$.

Розподілимо потоки *j* за розміром, тоді потік (*i,k*) буде мати індекс *j*, якщо він має *j*-ий найменший розмір зі всіх потоків. Отже, $s_1 d_1 < \dots < s_{n(t)} d_{n(t)}$. Визначимо розподіл в межах процесу квазі-оптимальної багатокласової адаптації

$$s_j^{\bar{\pi}_m}(t) = \begin{cases} s_j, & j = 1, \dots, \bar{n} - 1 \\ a_j s_j, & j = \bar{n}, \dots, n(t) \end{cases}, \tag{3.21}$$

де

$$\bar{n} = \max\{\left|\sum_{j=1}^{m-1} s_j + \sum_{j=m}^{n(t)} a_j s_j \le c\right\}. \tag{3.22}$$

Як наслідок, процес квазі-оптимальної адаптації досягається наданням максимального рівня підписки якомога більшій кількості малорозмірних потоків без урахування асоційованого з ними класу обслуговування із збереженням достатньої пропускної здатності для обслуговування інших потоків більшого розміру з мінімальним рівнем підписки, який відповідає рівню якості обслуговування їхнього класу. Як результат, отримано процес квазі-оптимальної адаптації, який базується на використанні тільки мінімального і максимального рівнів підписки, які визначені надавачем послуг. Адаптація є квазі-оптимальною

тому, що може залишитись невикористаною певна пропускна здатність. Отже, відносна різниця в значенні цільової функції між процесом квазі-оптимальної і оптимальної адаптації прямує до нуля зі зростанням кількості потоків. Тобто, для каналів з великою пропускною здатністю різницею у значеннях цільової функції в результаті здійснення двох згаданих видів адаптації можна знехтувати.

Нехай наближений усереднений нормований рівень підписки *k-го* класу обслуговування потоків відповідно до багатокласового масштабування швидкості передавання є

$$q_k^{\gamma,\pi_m} = \lim_{m\to\infty} E^0\,[Q_k^{m,\pi_m}],$$

де $E^0[Q_k^{m,\pi_m}]$ – це якість обслуговування в каналі m за процесом адаптації π_m для *k-ого* класу обслуговування.

Запишемо рівень підписки з урахуванням коефіцієнту масштабування пропускної здатності

$$q_k^{\gamma,\pi_m} = \begin{cases} \alpha_k, & \gamma < \alpha \\ 1-(1-\alpha_k)\bar{F}_{V_k}(v^*), & \alpha \le \gamma \le 1 \\ 1 & \gamma > 1 \end{cases} \tag{3.23}$$

де $\bar{F}_{V_k}(\cdot)$ є розподілу і v^* є унікальним значенням v

$$\sum_{k=1}^{K}(\bar{p}_k - \underline{p}_k)\,F_{\hat{V}_k}(v) = \gamma\bar{p} - \underline{p}. \tag{3.24}$$

Варто відзначити, що наближена QoS для класу k – це мінімальне значення α_k, коли канал передавання знаходиться в перевантаженому режимі, тобто $\gamma < \alpha$. Отже, припущення щодо незмінності мінімального рівня підписки інформаційних потоки, які вже отримують сервіс, для забезпечення пропускної здатності каналу для нових вхідних потоків, справджується у перевантаженому режимі. Відповідно, наближена QoS для класу k досягає максимуму, коли канал працюює в ненавантаженому режимі, тобто $\gamma > 1$. Це означає, що цей режим забезпечує потокам максимальний рівень підписки. Наближена QoS в режимі адаптації при $\alpha \le \gamma \le 1$ є функцією гарантованої QoS, покласового розподілу пропускної здатності і параметру масштабування. Залежність процесу адаптації

для класу k від розподілу тривалостей потоків F_{D_k} і розподілу максимальних рівнів підписки F_{S_k} виникає через розподіл пропускної здатності F_{V_k}.

Потенційно процес оптимальної адаптації потребує динамічної адаптації миттєвого рівня підписки для кожного потоку в моменти, коли потік надходить або надсилається. Сукупність потоків має більш-менш фіксований розподіл для каналів з великою пропускною здатністю, які обслуговують велику кількість потоків. Отже, миттєвий рівень підписки у процесі оптимальної адаптації, для потоків малого чи великого розміру, навряд чи буде істотно відрізнятися.

У зв'язку з цим потокам, що надійшли, присвоюють фіксований рівень підписки, який гарантується на час всього перебування в системі, тому динамічна адаптація не застосовується. Варто відзначити, що невеликі потоки будуть обслуговуватись із максимальним рівнем підписки, а великі потоки – із мінімальним рівнем підписки, які гарантовані відповідними їм класами обслуговування. У цьому випадку множина всіх політик доступу є підмножиною множини адаптації, так як політика доступу генерує рішення про надання ресурсів з визначеного процесу адаптації тільки під надходження потоку. Отже, якщо політика доступу має таку ж наближену QoS, як і процес оптимальної адаптації, то політика доступу є оптимальною. Покажемо оптимальну межу, яка відрізняє малі і великі потоки, і показує що наближена QoS за оптимальної політики доступу відповідає наближеній QoS у процесі оптимальної адаптації. Це доводить твердження про оптимальність політики доступу.

Політика контролю доступу π_{am} в ІКС є асимптотично-оптимальною та визначає оптимальну межу обсягу великих і малих потоків $v^{\gamma,\pi_{am}}$

$$\boldsymbol{v}^{\gamma,\pi_{am}} = \begin{cases} 0, & \gamma < \alpha \\ v^*, & \alpha \le \gamma \le 1 \\ \infty, & \gamma > 1 \end{cases} \tag{3.25}$$

де v^* – розв'язок v з рівняння

$$\sum_{k=1}^{K} (\bar{p}_k - \underline{p}_k) F_{\hat{V}_k}(v) = \gamma \bar{p} - \underline{p} \tag{3.26}$$

Оптимальна політика контролю доступу *i*-го користувача *k*-го класу полягає у присвоєнні користувачеві рівня підписки

$$s_{i,k}^{\gamma,\pi_m} = \begin{cases} S_{i,k}, & v_{i,k} \le v^{\gamma,\pi_{am}} \\ \alpha_k S_{i,k}, & \text{інші} \end{cases}. \quad (3.27)$$

Очікуваний асимптотичний нормалізований рівень підписки для потоків класу k із застосуванням політики π_{am} відповідає отриманому значенню у процесі оптимальної багатокласової адаптації π_m.

Відзначимо, що оптимальна межа не залежить від класу обслуговування *k,* тобто мережа розрізняє потокиза їх розміром, незалежно від класу надання послуг.

Відповідно, наближено-оптимальна політика контролю доступу відзначається простотою і може бути застосована, коли статистичні властивості випадкового рядку максимального рівня підписки та тривалості потоків відомі для кожного класу надання послуг.

3.2.3. Чисельний розв'язок завдання максимізації якості надання різнокласових послуг

Нехай у каналі ІКС надаються три типи сервісу:

– Однокласове обслуговування. Аудіо та відеопотоки спільно використовують пропускну здатність і спільно адаптовані за процесом оптимальної однокласової адаптації. Вважатимемо, що аудіо та відеопотоки володіють незмінною адаптивністю потоку $\alpha_1 = \frac{1}{4}$. Тобто гарантії рівня обслуговування для кожного класу рівні між собою, і $\alpha_k = \alpha$ для кожного $k = 1, \dots K$.

– Двокласове обслуговування. Надаються два класи сервісу: загальний клас обслуговування з гарантованим нормованим рівнем підписки α_1 і преміальний клас обслуговування з гарантованим нормованим рівнем підписки $\alpha_2 = \frac{3}{4}$. Користувачі аудіо-сервісу, здебільшого, підписані на загальний клас, що здатен розрізняти розміри потоків для надання їм прийнятної QoS, а користувачі відео-сервісу, в основному, підписані на

преміальний клас для того, щоб гарантувати рівень QoS незалежно від їх розміру.

– Окремі канали передавання. У цьому режимі пропускна здатність поділена порівну на два окремі канали, один – для аудіо-, а інший – для відеопотоків. Аудіопотоки сумісно використовують призначений їм канал, а всі відеопотоки – свій канал. Потоки у кожному каналі адаптовані за процесом оптимальної однокласової адаптації. Аудіопотокам надано нормований рівень підписки α_1, а відеопотоки отримують нормований рівень підписки α_2.

Визначені типи обслуговування у графічному вигляді представлено на Рис. 3.4.

Цифрою «1» позначимо загальний клас обслуговування аудіопотоків, а цифрою «2» – преміальний клас обслуговування відеопотоків.

Інтенсивність надходження для користувачів аудіопотоків $\lambda_1 = 1$, середня тривалість потоку $\delta_1 = 180$ с (тобто типова довжина аудіозапису), а середній максимальний рівень підписки $\sigma_1 = 0{,}1$ Мбіт/с; класом обслуговування забезпечено $\alpha_1 = \frac{1}{4}$. Розподіли тривалості аудіопотоків і максимального рівня підписки – експоненційні. Отже, $F_{D_1}(d) = 1 - \exp(-\frac{d}{\delta_1})$, а $F_{S_1}(s) = 1 - \exp(-\frac{s}{\sigma_1})$.

Інтенсивність надходження для користувачів відеопотоків $\lambda_2 = 0{,}01$, яка залежить від функції попиту, яка, у свою чергу, оцінюється як вартість використання преміального класу. Середня тривалість відеопотоку $\delta_2 = 1800$ с, а середній максимальний рівень підписки – $\sigma_2 = 1{,}0$ Мбіт/с. Надавач відеоконтенту забезпечує мінімальне кодування потоку з адаптивністю $\alpha_2 = \frac{1}{4}$, але преміальний клас гарантує QoS $\alpha_2 = \frac{3}{4}$. Тобто, користувачі відеопотоків з низькою платоспроможністю обиратимуть клас 1 нижчої якості з адаптивністю $\frac{1}{4}$, а користувачі з великою платоспоможністю віддаватимуть перевагу другому

класу з адаптивністю $\frac{3}{4}$, що надає потік відеоданих з високою якістю обслуговування. Статистичні розподіли часу існування відеопотоків та максимального рівня підписки – експоненційні. Отже, $F_{D_2}(d) = 1 - \exp(-\frac{d}{\delta_2})$, а $F_{S_2}(s) = 1 - \exp(-\frac{s}{\sigma_2})$.

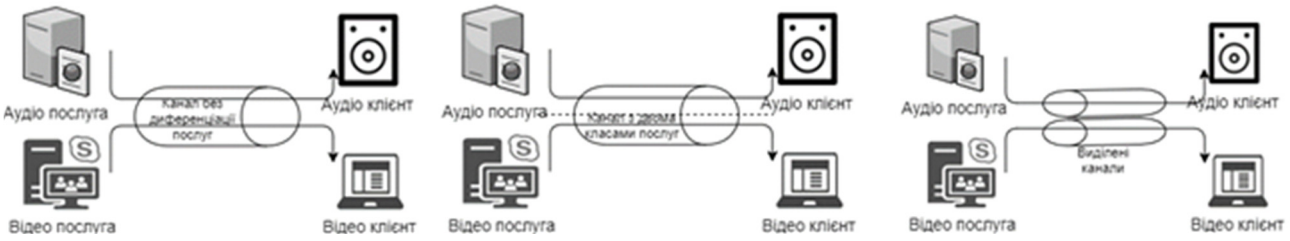


Рис. 3.4. Графічне представлення трьох типів надання двох видів послуг в каналі передавання

Мінімальне навантаження для аудіопотоків $\underline{\rho_1} = (\lambda_1\delta_1)(\alpha_2\sigma_1) = (1 \times 180)\left(\frac{1}{4} \times \frac{1}{10}\right) = 4.5$ Мбіт/с, і максимальне навантаження – $\bar{\rho}_1 = (\lambda_1\delta_1)(\sigma_1) = (1 \times 180)\left(\frac{1}{10}\right) = 18$ Мбіт/с. Мінімальне навантаження відеопотоків без преміального класу обслуговування $\underline{\rho_2} = (\lambda_2\delta_2)(\alpha_2\sigma_2) = \left(\frac{1}{100} \times 1800\right)\left(\frac{1}{4} \times 1\right) = 4.5$ Мбіт/с. Мінімальне навантаження для відеопотоків, коли пропонується преміальний клас обслуговування $\underline{\rho_2} = (\lambda_2\delta_2)(\alpha_2\sigma_2) = \left(\frac{1}{100} \times 1800\right)\left(\frac{3}{4} \times 1\right) = 13.5$ Мбіт. Максимальне навантаження для відео-клієнтів (для всіх трьох типів сервісу) $\bar{\rho}_2 = (\lambda_2\delta_2)(\sigma_2) = \left(\frac{1}{100} \times 1800\right)(1) = 18$ Мбіт/с. Тобто, загальне мінімальне навантаження – 9 Мбіт/с без преміального класу обслуговування і 18 Мбіт/с, коли надається преміальний сервіс. Загальне максимальне навантаження становить 36 Мбіт/с. Відзначимо, що максимальне навантаження становить 18 Мбіт/с для аудіо і відеопотоків. Це обумовлює доцільність поділу пропускної здатності порівну на два канали у випадку третього типу обслуговування.

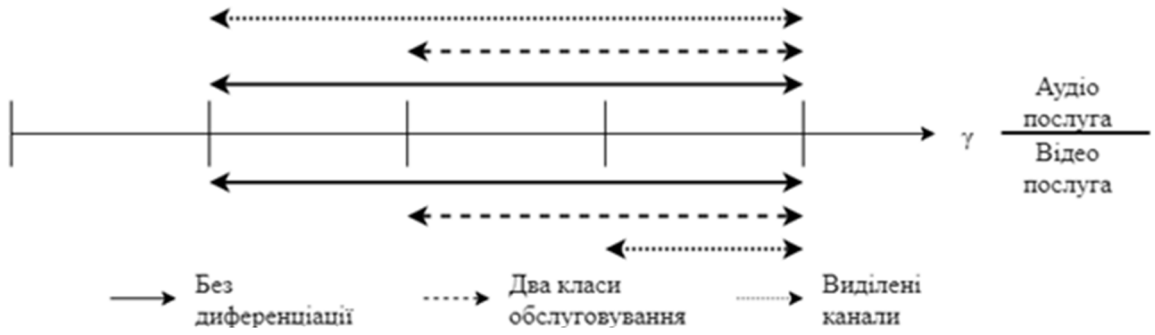


Рис. 3.5. Графічне представлення процесу адаптації аудіо та відеопотоків для кожного типу обслуговування

Обслуговування різних типів інформаційних потоків розрізняється за режимами масштабування виділеної пропускної здатності, що пояснюється диференціацією класів обслуговування та процесів адаптації каналів передавання. Згадані режими подано на Рис. 3.5. Проаналізуємо перший тип обслуговування, у якому аудіо- і відеопотоки використовують нормований рівень підписки $\alpha_1 = \frac{1}{4}$ у процесі оптимальної однокласової адаптації. Загальна адаптивність у цьому випадку $\frac{\rho}{\bar{\rho}} = \frac{9}{36} = \frac{1}{4}$, тобто режим адаптації інтенсивності аудіо- та відеопотоків визначається нерівністю $\frac{1}{4} \leq \gamma \leq 1$. У другому типі сервісу аудіопотоки володіть гарантованим нормованим рівнем підписки $\alpha_1 = \frac{1}{4}$, а відепотоки мають гарантований нормований рівень підписки $\alpha_2 = \frac{3}{4}$. Загальна адаптивність складає $\frac{\rho}{\bar{\rho}} = \frac{18}{36} = \frac{1}{2}$, тобто режим адаптації інтенсивності визначається нерівністю $\frac{1}{2} \leq \gamma \leq 1$. Діапазон адаптації менший для цього типу сервісу, оскільки відеопотокам надається більший гарантований нормований рівень підписки. У третьому типі сервісу аудіо- та відеопотоки передаються окремими канали, таким чином, що аудіопотокам надано нормований рівень підписки $\alpha_1 = \frac{1}{4}$, а відеопотокам – нормований рівень підписки $\alpha_2 = \frac{3}{4}$. Оскільки використовуються різні канали, режим адаптації інтенсивності для аудіопотоків відповідає $\frac{1}{4} \leq \gamma \leq 1$, в той час, як режим адаптації для відеопотоків відповідає $\frac{3}{4} \leq \gamma \leq 1$.

Отже, досягнення балансу між імовірністю блокування потоку і рівнем підписки здійснюється шляхом вибору типу обслуговування. Режим найменшого перевантаження має перший тип обслуговування з відсутнім преміальним обслуговуванням відеопотоків $0 < \gamma < \frac{1}{4}$. Другий тип обслуговування має спільний режим перевантаження $0 \le \gamma \le \frac{1}{2}$. Тобто перевантаження в межах $\frac{1}{4} \le \gamma \le \frac{1}{2}$ є вартістю збільшення гарантованої якості обслуговування відеопотоків з $\frac{1}{4}$ до $\frac{3}{4}$. І, нарешті, третій тип обслуговування має різні режими адаптації для двох каналів. Аудіоканал перевантажений тільки при $0 \le \gamma \le \frac{1}{4}$, а відеоканал – при $0 \le \gamma \le \frac{3}{4}$.

Перший тип сервісу еквівалентний моделі багатьох класів сервісу, де всі класи забезпечують однакову QoS. Запишемо наближений нормований рівень підписки на інформаційні аудіо- та відеопотоки (клас 1 і клас 2 відповідно)

$$q_1^{\gamma,1} = 1 - (1 - \alpha_1)\bar{F}_{V_1}(F_{\hat{V}_1} v_1^*(\gamma)) \tag{3.28}$$

$$q_2^{\gamma,1} = 1 - (1 - \alpha_1)\bar{F}_{V_2}(F_{\hat{V}_2} v_1^*(\gamma)) \tag{3.29}$$

де $v_1^*(\gamma)$ є розв'язком v з рівняння

$$(\bar{\rho}_1 - \underline{\rho}_1)F_{\hat{V}_1}(v) + (\bar{\rho}_2 - \underline{\rho}_2)F_{\hat{V}_2}(v) = \gamma\bar{\rho} - \underline{\rho}. \tag{3.30}$$

У першому типі сервісу $\bar{\rho}_1 = 18,\ \underline{\rho}_1 = 4{,}5,\ \bar{\rho}_2 = 18, \underline{\rho}_2 = 4{,}5,\ \overline{\rho} = 36$ і $\underline{\rho} = 9$. З урахування цього, запишемо (3.30) у такому вигляді

$$F_{\hat{V}_1}(v) + F_{\hat{V}_2}(v) = \frac{8}{3}\gamma - \frac{2}{3}. \tag{3.31}$$

Для другого типу сервісу наближений нормований рівень підписки для аудіо- та відеопотоків становить

$$q_1^{\gamma,2} = 1 - (1 - \alpha_1)\bar{F}_{V_1}(F_{\hat{V}_1} v_2^*(\gamma)) \tag{3.32}$$

$$q_2^{\gamma,2} = 1 - (1 - \alpha_2)\bar{F}_{V_2}(F_{\hat{V}_2} v_2^*(\gamma)) \tag{3.33}$$

де $v_2^*(\gamma)$ є розв'язком (3.30), але з $\underline{\rho}_2 = 13{,}5$ і $\underline{\rho} = 18$. Отже,

$$F_{\widehat{V}_1}(v) + \frac{1}{3}F_{\widehat{V}_2}(v) = \frac{8}{3}\gamma - \frac{4}{3}. \quad (3.34)$$

Для третього типу сервісу наближений нормований рівень підписки для аудіо- та відеопотоків становить

$$q_1^{\gamma,3} = 1 - (1-\alpha_1)\bar{F}_{V_1}(F_{\widehat{V}_1}\frac{\gamma-\alpha_1}{1-\alpha_1}) \quad (3.35)$$

$$q_2^{\gamma,3} = 1 - (1-\alpha_2)\bar{F}_{V_2}(F_{\widehat{V}_2}\frac{\gamma-\alpha_2}{1-\alpha_2}) \quad (3.36)$$

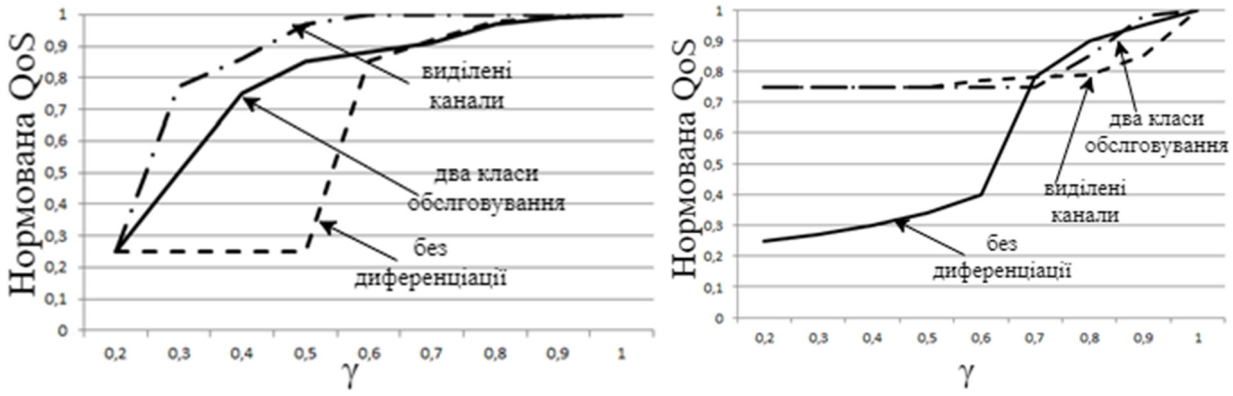
Ці рівняння безпосередньо випливають з аналізу єдиного класу обслуговування.

На рис. 3.3 представлено залежність нормованого рівня підписки (якості обслуговування) для аудіо- та відеопотоків із застосуванням трьох визначених типів сервісу. На рис. 3.3, а представлено графіки наближеної QoS для аудіопотоків для першого, другого і третього типів обслуговування, а на рис. 3.3, б – наближеної QoS за аналогічних умов.

Проаналізуємо перший тип сервісу. Аудіо і відеопотоки обслуговуються з мінімальною гарантованою якістю, тобто $\frac{1}{4}$, а у перевантаженому режимі – $0 < \gamma \leq \frac{1}{4}$. Зі зростанням γ в процесі адаптації, коли $\frac{1}{4} \leq \gamma \leq 1$, аудіопотоки отримують швидке зростання якості до максимального нормованого рівня підписки 1 при $\gamma = 0{,}6$, при цьому зростання якості для відеопотоків майже не спостерігається. Така картина спостерігається тому, що в однокласовому режимі обслуговування зі спільними обмеженнями QoS процес оптимальної адаптації віддає перевагу меншим потокам і обмежує більші потоки. Зі зростанням γ більше 0,6, можна спостерігати, що якість зростає більш стрімко для відеопотоків. Пояснюється це тим, що аудіопотоки на цей момент вже отримують максимальну якість, у зв'язку з цим від доступної пропускної здатності підвищує якість обслуговування відеопотоків.

За другого типу сервісу аудіо- і відеопотокам надаються відповідні мінімальні забезпечення якості ($\frac{1}{4}$ і $\frac{3}{4}$, відповідно) в перевантаженому режимі, тобто $0 < \gamma \leq \frac{1}{2}$. Зі зростанням γ у процесі адаптації, тобто $\frac{1}{2} \leq \gamma \leq 1$, можна

бачити, що якість обслуговування аудіопотоків швидко зростає до значення максимального нормованого рівня підписки 1, коли $\gamma = 0{,}9$. До цього значення параметру масштабування зростання якості для відеопотоків майже не спостерігається, що відбувається через особливості процесу оптимальної адаптації, яка надає перевагу потокам меншого розміру. У випадку зростання γ понад 0,9, бачимо, що якість обслуговуваггя відепотоків зростає більш помітно. Зауважимо, що для $\gamma > 0{,}7$ перший тип сервісу надає вищу усереднену QoS для відеопотоків у порівнянні з другим типом сервісу.



а) б)

Рис. 3.6. Наближена залежність нормованого рівня підписки аудіо- (а) та відеопотоків (б) від параметру масштабування γ у трьох типах сервісу

Звичайно, цей наслідок може видатися нелогічним, адже другий тип сервісу призначений для якіснішого обслуговування відеопотоків у порівнянні з першм типом. Але потрібно пам’ятати, що перший тип сервісу має змогу адаптувати відеопотоки із максимальною інтенсивністю до нормованого рівня підписки, що становить $\frac{1}{4}$, а другий тип сервісу адаптує найбільші відеопотоки до нормованого рівня підписки $\frac{3}{4}$. Ось чому другий тип сервісу має адаптувати більше відео потоків, ніж перший, чим і пояснюється нижча узагальнена середня якість обслуговування.

Для третього типу сервісу можемо спостерігати, що аудіопотоки обслуговуються з мінімальним нормованим рівнем підписки $\frac{1}{4}$ в режимі

перевантаження, коли $0 < \gamma < \frac{1}{4}$. У свою чергу, відеопотоки обслуговуються з мінімальним рівнем підписки $\frac{3}{4}$ в режимі перевантаження, коли $0 < \gamma < \frac{3}{4}$. Зростання γ у процесі адаптації аудіопотоків тобто $\frac{1}{4} \leq \gamma \leq 1$, забезпечує швидке зростання якості обслуговування аудіопотоків, яка у всьому діапазоні адаптації знаходиться у межах якості обслуговування, отриманої за першого і другого типів сервісу. При цьому якість обслуговування аудіопотоків у третьому типі сервісу завжди нижча, ніж у першому типі сервісу, оскільки в спільному класі обслуговування присутні відеопотоки, адаптація яких здійснюється перед адаптацією аудіопотоків. Також, якість обслуговування аудіопотоків за третього типу сервісу більша у порівнянні із другим типом, коли $\frac{1}{4} \leq \gamma \leq \frac{3}{4}$, оскільки в цьому режимі блокуються відеопотоки. У випадку третього типу сервісу, коли $\frac{3}{4} \leq \gamma \leq 1$, якість обслуговування аудіопотоків еквівалентна за другого і третього типів сервісу.

Проаналізуємо якість обслуговування, коли використовується третій тип сервісу. Бачимо, що якість обслуговування відеопотоків вища для другого типу, ніж за третього, коли $\frac{1}{2} < \gamma \leq \frac{3}{4}$, що пояснюється блокування відеопотоків у цьому режимі адаптації для третього типу сервісу, тому всім відеопотокам надається мінімальний рівень підписки. Коли $\frac{3}{4} \leq \gamma \leq 1$, то за третього типу сервісу відеопотоки не блокуються, чим пояснюється вища якість їхнього обслуговування у порівнянні із другогим типом. Така картина спостерігається, бо якістю обслуговування відеопотоків у другому типі сервісу жертвують для більш високої якості обслуговування аудіопотоків в згаданому режимі адаптації через використання процесу оптимальної адаптації інтенсивності.

Сукупність результатів, представлених на Рис. 3.6, може використовуватись у процесі планування каналу з метою досягнення оптимального співвідношення низького блокування інформаційних потоків в ІКС і високих нормованих рівнів підписки. Коли планувальник зможе забезпечити обслуговування за $\gamma > \frac{3}{4}$, то

третій тип сервісу гарантує максимально можливу якість обслуговування для аудіо- та відеопотоків і відсутність блокування. Коли ж надання послуг планувальнком може бути забезпечене тільки для $\frac{1}{2} \leq \gamma \leq \frac{3}{4}$, то варто обрати другий тип сервісу, оскільки тоді буде забезпечено мінімізацію блокування для аудіо- та відеопотоків, а також надано гарантований рівень якості обслуговування для відеопотоків. У випадку, коли мале блокування є важливішим, ніж рівень якості обслуговування, і послуги надаються в режимі $\frac{1}{4} \leq \gamma \leq \frac{1}{2}$, то більш пріоритетним виглядає вибір першого типу обслуговування.

3.3. Метод максимізації інтенсивностей сервісних інформаційних потоків у телекомунікаційній мережі заданої структури

У сучасних телекомунікаційних мережах важливим складним завданням є аналіз потоку даних для надання послуг кінцевим користувачам. Швидкість потоку залежить від кількості користувачів, які підключені до сегменту локальної мережі, типів послуг та даних, які вони передають. Сучасні телекомунікаційні мережі орієнтовані на зміст; отже, важливо, щоб користувачі отримували високоякісні інформаційні та телекомунікаційні послуги. Деяка кількість ресурсів повинна бути розподілена на потоки послуг для надання цих послуг. Отже, управління та розподіл мережевих ресурсів між кількома потоками послуг для забезпечення високоякісного надання послуг є актуальним науковим завданням.

Проблема, розглянута в документі, насправді, може бути сформульована як питання: у випадку декількох потоків послуг, скільки пропускної спроможності ми можемо виділити для кожного потоку послуг у кожній точці мережі?

Робота пов'язана з дослідницькою областю, яка визначає використовувані фракції пропускної здатності, коли користувачі отримують доступ до послуг. Деякі дослідники пропонують методи та методи розподілу мережевих ресурсів, за винятком впливу процесу маршрутизації та характеристик мультисервісного трафіку. Однак ці статті не враховують впливу логічної структури на передачу потоків послуг.

Дуже важливо вказати на абстракцію проблеми, щоб забезпечити її незалежність від будь-яких мережевих технологій та / або протоколів. Вони можуть бути застосовані лише на етапі вивчення конкретного випадку.

Автори даної роботи раніше запропонували спосіб сервіс-орієнтованого планування ресурсів, концепція забезпечення послуг в гетерогенних сервіс-орієнтованих систем і методологічній основі гетерогенного розподілу пропускної здатності мережі між потоків послуг. На підставі згаданих робіт ми провели детальний чисельний аналіз розподілу мережевих ресурсів на потоки послуг, використовуючи представлення проблеми як задачі лінійного програмування. Цей підхід має обмеження у використанні через нелінійний та стохастичний характер мережевих процесів. У будь-якому випадку, це може бути застосовано у випадку, коли потоки послуг є постійними значеннями. Цю ситуацію можна спостерігати лише за дуже короткі часові ряди. Тому рішення проблеми має тимчасовий ефект і повинно бути перераховане кожного разу, коли змінюються потоки сервісу чутливо. Основна перевага запропонованого підходу полягає в простоті операцій і, отже, в тривалості обчислювальної тривалості. Як правило, обчислення вирішення проблеми нелінійної оптимізації набагато більше часу, навіть якщо його можна застосувати в довгостроковій перспективі. Обчислювальна складність також набагато вище. У багатьох випадках дуже важко реалізувати нелінійний розв'язок на практиці. Тому більша частина наукового суспільства намагається перетворити нелінійні проблеми на лінійні, якщо це можливо. Цей процес потребує також спрощення обмежень рішення для лінійних функцій.

3.3.1. Формулювання завдання максимізації інтенсивностей сервісних потоків як завдання лінійного програмування

Мережевий ресурс представлений загальною ємністю каналів, що сполучають мережні пристрої. Обсяг ресурсу, що виділяється потоку, залежить від попередньо визначеної структури мережі (топологія мережі). Швидкість сервісного потоку залежить від пропускної здатності каналу та обмежується іншими потоками між тією ж парою вузлів. Набір всіх маршрутів формує логічну

структуру мережі. Ця логічна структура є обмежуючим фактором, оскільки фізичні ресурси не можуть бути використані повністю після організації логічної структури. Таким чином, процес утворення логічної структури є визначальним фактором максимізації швидкості сервісного потоку. Проблема є загальною для всіх типів мереж, які використовують провідний або безпроводовий зв'язок. Різниця виникає лише тоді, коли ми аналізуємо мережеві протоколи, котрі використовуються для формування логічної структури. У даному випадку ми будемо аналізувати найбільш поширені протоколи мережного рівня, такі як RIP v.2 та OSPF. У будь-якому випадку, охоплена проблема не обмежується такими протоколами. Її можна розширити за допомогою протоколів розподілу радіоресурсів. Загалом, поточна проблема виникає, коли виконується багаторазовий доступ до спільних ресурсів. Її вирішення може бути досягнуте шляхом максимізації ресурсів, що виділяються конкретному замовнику. У нашому випадку попит від замовника – це сервісний потік, який повинен передаватися через мережу.

Відомо, що формування логічної структури є або статичним (що є дуже рідким), або динамічним (за допомогою динамічних протоколів маршрутизації); або поєднання обох підходів. Динамічна маршрутизація може бути застосована шляхом вибору різних критеріїв маршруту: метрика найнижчої кількості переходів по маршруту (RIP) або стану каналу (OSPF). Логічні структури, утворені за допомогою кожного з протоколів, зазвичай не однакові, і тому частка виділених фізичних ресурсів може бути різною через різне навантаження мережі, тому протокол маршрутизації безпосередньо впливає на бажану швидкість кожного сервісного потоку.

Під потоком ми маємо на увазі трафік інформаційних пакетів між парою вузлів, який передається за допомогою одного маршруту.

Визначаючи потік за маршрутом, проблема розподілу мережевих ресурсів може бути сформульована з точки зору задачі лінійного програмування таким чином: максимізувати швидкість кожного потоку послуг на вході деякого мережевого вузла, що забезпечує одночасне існування та справедливу

конкуренцію потоків і дає змогу оцінити максимальну кількість клієнтів, що користуються послугою, з різними рівнями якості обслуговування.

Телекомунікаційна мережа представлена графом $G = (V, E)$, з логічною структурою

$$L = \{\mu(1,2); \mu(1,3); \ldots; \mu(i,j)\}, \tag{3.37}$$

де $i, j \in V$ утвореною за допомогою алгоритму динамічної маршрутизації. Логічна структура обмежує використання фізичних ресурсів, представлених вектором $\vec{x}$. Розрахунок елементів цього вектора може бути виконаний шляхом розв'язання завдання лінійного програмування, представленого наступною системою нерівностей [11, 23]:

$$\min_x f^T x, \quad \begin{matrix} A \times x \le b \\ Aeq \times x = beq \\ eb \le x \le ub \end{matrix} \tag{3.38}$$

де $f(x)$ – цільова функція; A, A_{eq} – коефіцієнти лінійних рівнянь; x – шукана змінна; eb, ub – верхня і нижня межі шуканої змінної.

Запишемо задачу оптимізації у такому вигляді:

$$\min_x f(x) \tag{3.39}$$

де $f(x)$ – лінійна цільова функція, що розраховується як

$$f(x) = -\sum_{(i,j)} x_{i,j} \tag{3.40}$$

де i, j – номери вузлів, $x_{i,j}$ – змінна, яка визначає швидкість сервісного потоку між парою вузлів. Розв'язок завдання має задовольняти системі обмежувальних умов:

– обмеження продуктивності вузла:

$$\sum_{j \in \mu(i,j)} x_{i,j} + 2 \cdot \sum_{k \in \mu(i,j)} x_{k,j} + \sum_{i \in \mu(i,j)} x_{i,j} \le \sum_{j \in V} C_{i,j} \tag{3.41}$$

де $C_{i,j}$ – пропускна здатність каналу (i, j) (в умовних одиницях), $\mu(i, j)$ – маршрут між парою вузлів i, j ;

– обмеження пропускної здатності каналу:

$$\sum_{i,j \in \mu(i,j)} x_{i,j} \le C_{i,j} \tag{3.42}$$

– обмеження конкуренції інформаційних потоків:

$$0,5 \cdot \frac{C_{i,j}}{N^{\max}_{\mu(i.j)}} \le x_{i,j} \le C_{i,j} \tag{3.43}$$

де $N^{\max}_{\mu(i.j)}$ – максимальна кількість потоків, що передають у каналі (i, j), який належить до маршруту $\mu(i, j)$.

Ці умови були встановлені для забезпечення рівної конкуренції між мережними потоками через особливості методів лінійного програмування. Ми зробили висновок, що якщо два або більше потоків передаються в мережний канал, то потік через найкоротший шлях отримує всі ресурси, подавлюючи тим самим інші потоки.

3.3.2. Вибір методу розв’язання завдання лінійного програмування

Розв’язок сформульованого завдання може бути досягнуто за допомогою системи MatLab, яка було обрана, оскільки включає функції для лінійної оптимізації та дозволяє працювати з графами. З цією метою ми використовували дві бібліотеки - Bioinformatics Toolbox та Optimization Toolbox. Bioinformatics Toolbox містить методи створення, аналізу, обробки та візуалізації графів. Optimization Toolbox реалізує численні алгоритми лінійної та нелінійної оптимізації. Використовуючи ці методи, ми розробили програмну модель телекомунікаційної мережі та розв’язали завдання розподілу мережних ресурсів серед непріоритетних службових потоків, забезпечуючи максимізацію їх інтенсивностей.

Використовуючи функцію оптимізації biograph(), граф представлений як графовий об'єкт. Використовуючи функцію shortestpath(), ми змогли розрахувати

найкоротші шляхи між кожною парою вузлів у представленому графі. Отримані маршрути разом із вагами ребер графа є вхідними даними для завдання оптимізації. Оскільки завдання, яке потрібно розв'язати, належить до класу лінійного програмування, ми використовуємо функцію linprog() на основі алгоритму Interior point.

Вирішення завдання розподілу ємності повинно здійснюватися для конкретної реалізації телекомунікаційної мережі. У нашому випадку її представлено зваженим графом з 9 вершинами і 16 ребрами (Рис. 3.7). На підставі цього графа ми розрахували маршрути між кожною парою вузлів у двох випадках: використання протоколу маршрутизації RIP та використання протоколу маршрутизації OSPF. Ми припускаємо, що в мережі використовується тільки одношляхова маршрутизація.

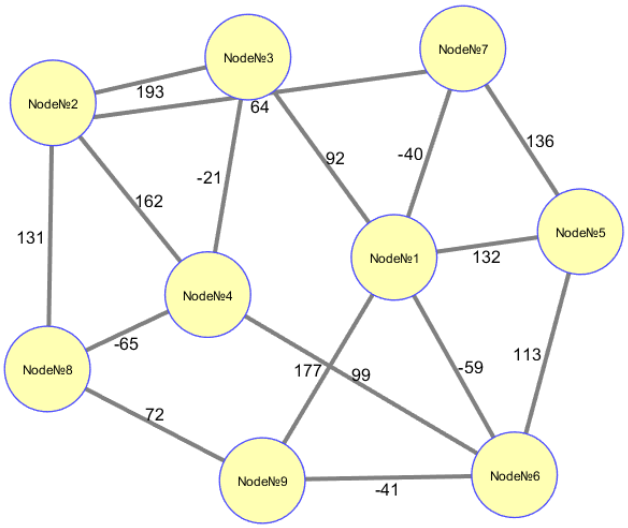


Рис. 3.7. Граф досліджуваної телекомунікаційної мережі

3.3.3. Чисельний розвязок завдання максимізації інтенсивностей сервісних інформаційних потоків

Відповідно до рівнянь (3.40) – (3.43) для даного графа формуються цільова функція та лінійні обмеження. Виходячи з них, ми розв'язали завдання розподілу ємності мережі між потоками послуг з заданими інтенсивностями, які підлягають максимізації. Ці показники представлені вектором.

Розв’язки завдання оптимізації із застосуванням методу Interior point розміщені в таблицях (Таблиця 3.3 і Таблиця 3.4).

Таблиця 3.3. Розв’язок завдання максимізації інтенсивності сервісних потоків у випадку маршрутизації OSPF

Кроки алгоритму	Первинна задача $A \cdot x - b$	Дуальна задача $A' \cdot y + z - f$	Розрив дуальності $x' \cdot z$	Відносна похибка
Ітерація 0	$9.86 \cdot 10^{3}$	52.2	$5.91 \cdot 10^{4}$	$7.20 \cdot 10^{3}$
Ітерація 1	$9.69 \cdot 10^{2}$	2.62	$7.60 \cdot 10^{3}$	$8.93 \cdot 10^{-1}$
Ітерація 2	90.00	$1.03 \cdot 10^{-1}$	$1.38 \cdot 10^{3}$	$5.35 \cdot 10^{-1}$
Ітерація 3	8.94	$7.51 \cdot 10^{-3}$	$1.78 \cdot 10^{2}$	$1.61 \cdot 10^{-1}$
Ітерація 4	$6.90 \cdot 10^{-1}$	$1.90 \cdot 10^{-3}$	44.30	$5.25 \cdot 10^{-2}$
Ітерація 5	$2.56 \cdot 10^{-1}$	$3.32 \cdot 10^{-4}$	16.80	$2.01 \cdot 10^{-2}$
Ітерація 6	$5.26 \cdot 10^{-2}$	$2.62 \cdot 10^{-13}$	3.66	$4.36 \cdot 10^{-3}$
Ітерація 7	$2.98 \cdot 10^{-4}$	$1.52 \cdot 10^{-13}$	$3.15 \cdot 10^{-2}$	$3.87 \cdot 10^{-5}$
Ітерація 8	$1.50 \cdot 10^{-8}$	$2.25 \cdot 10^{-14}$	$1.59 \cdot 10^{-6}$	$1.96 \cdot 10^{-9}$
Ітерація 9	$2.19 \cdot 10^{-12}$	$1.64 \cdot 10^{-14}$	$1.59 \cdot 10^{-14}$	$2.01 \cdot 10^{-15}$
Ітерація 10	$1.27 \cdot 10^{-13}$	$9.42 \cdot 10^{-16}$	$3.98 \cdot 10^{-17}$	$1.17 \cdot 10^{-16}$

Оптимальний розв’язок сформульованого завдання лінійного програмування у випадку маршрутизації OSPF досягається при 10 ітераціях.

Таблиця 3.4. Розв’язок завдання максимізації інтенсивності сервісних потоків у випадку маршрутизації RIP

Кроки алгоритму	Первинна задача $A \cdot x - b$	Дуальна задача $A' \cdot y + z - f$	Розрив дуальності $x' \cdot z$	Відносна похибка
Ітерація 0	$7.90 \cdot 10^{3}$	52.80	$6.09 \cdot 10^{4}$	$7.20 \cdot 10^{3}$
Ітерація 1	$1.30 \cdot 10^{3}$	$1.18 \cdot 10^{-14}$	$1.24 \cdot 10^{4}$	1.17
Ітерація 2	$1.43 \cdot 10^{2}$	$5.61 \cdot 10^{-14}$	$2.55 \cdot 10^{3}$	$5.84 \cdot 10^{-1}$
Ітерація 3	26.60	$3.30 \cdot 10^{-14}$	$4.19 \cdot 10^{2}$	$2.04 \cdot 10^{-1}$
Ітерація 4	1.61	$6.64 \cdot 10^{-14}$	37.70	$2.54 \cdot 10^{-2}$
Ітерація 5	$3.49 \cdot 10^{-1}$	$1.15 \cdot 10^{-13}$	8.93	$5.93 \cdot 10^{-3}$
Ітерація 6	$2.77 \cdot 10^{-4}$	$1.17 \cdot 10^{-14}$	$4.39 \cdot 10^{-2}$	$3.62 \cdot 10^{-5}$
Ітерація 7	$1.51 \cdot 10^{-8}$	$1.21 \cdot 10^{-14}$	$2.26 \cdot 10^{-6}$	$1.86 \cdot 10^{-9}$
Ітерація 8	$3.71 \cdot 10^{-13}$	$1.11 \cdot 10^{-14}$	$1.25 \cdot 10^{-13}$	$1.31 \cdot 10^{-15}$
Ітерація 9	$1.18 \cdot 10^{-13}$	$3.21 \cdot 10^{-15}$	$3.13 \cdot 10^{-16}$	$3.79 \cdot 10^{-16}$

Оптимальний розв’язок представленого завдання лінійного програмування у випадку маршрутизації RIP досягається при 9 ітераціях.

Розв’язок для протоколу RIP зображено на Рис. 3.8 (двонаправлені потоки, синій відповідає швидкості потоку з вихідної вершини до місця призначення, жовтий – навпаки), а також для протоколу OSPF – на Рис. 3.9.

Представлені результати відображають максимальну швидкість потоку, яка може бути досягнута в мережі, представленій графом на Рис. 3.7 у випадку використання протоколу RIP або OSPF. Ми помічаємо, що показники в кожному конкретному випадку відрізняються.

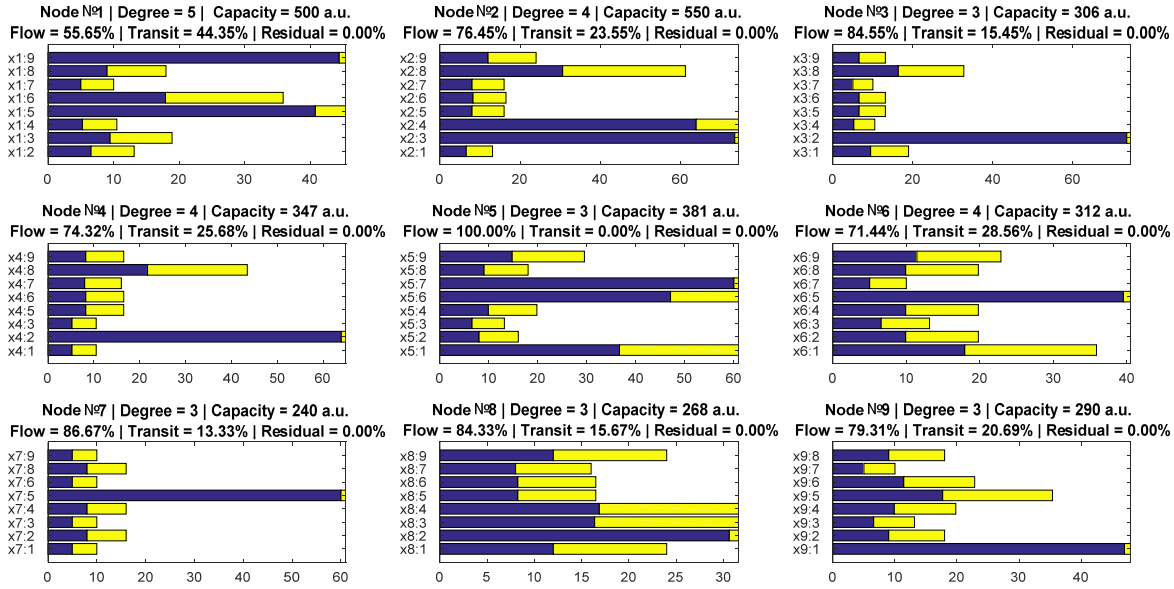


Рис. 3.8. Інтенсивності сервісних потоків між кожною парою вузлів у логічній структурі, утвореній протоколом RIP

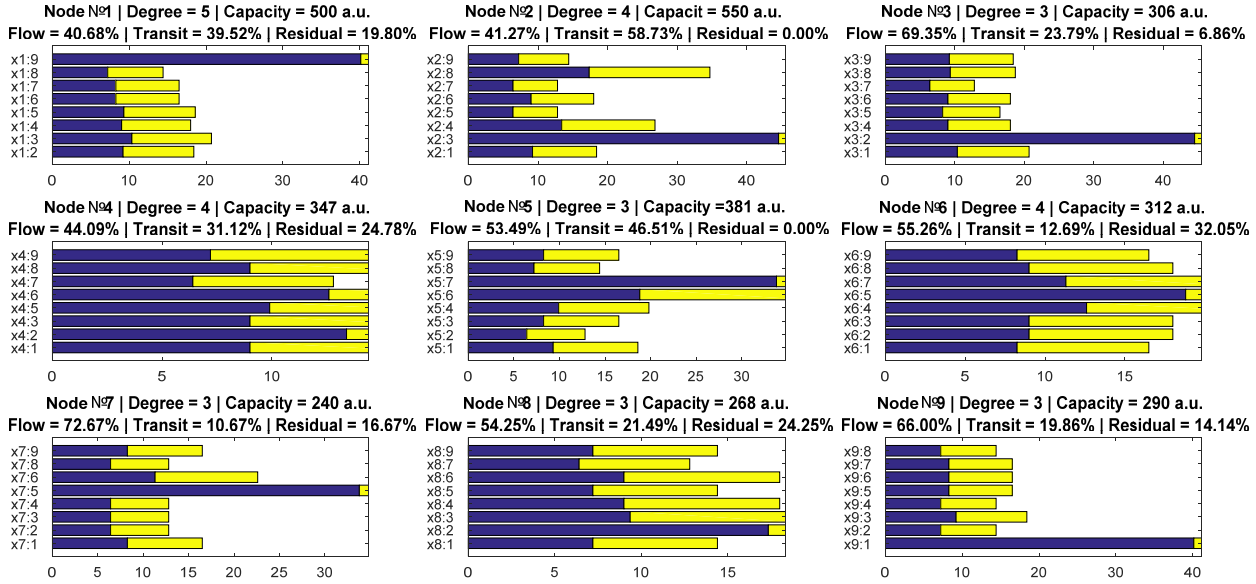


Рис. 3.9. Інтенсивності сервісних потоків між кожною парою вузлів у логічній структурі, утвореній протоколом OSPF

Це означає, що вивчені протоколи формують різні обмеження на використання мережевих ресурсів, як було зазначено вище. Числові результати дають нам краще розуміння процесів трафіку в даній мережі.

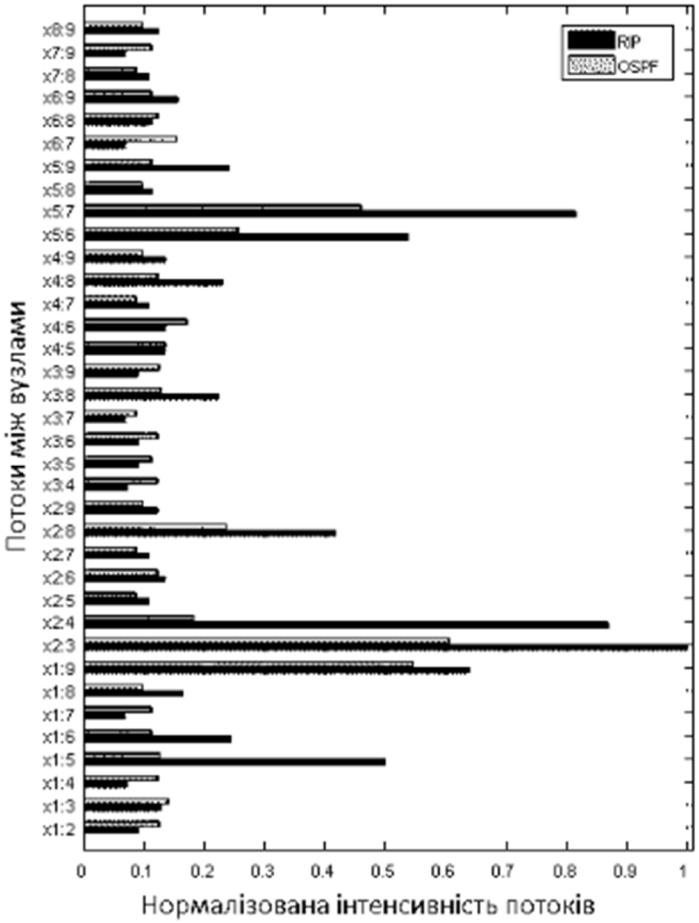
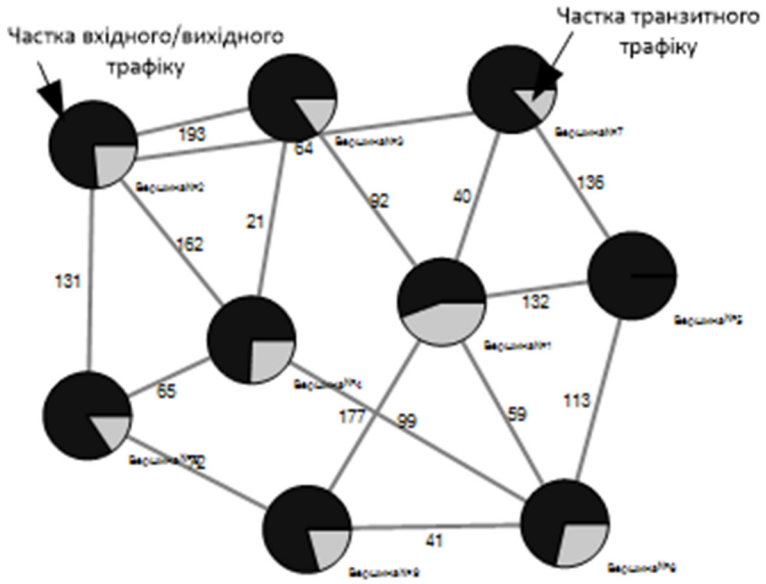


Рис. 3.10. Порівняння результатів для протоколів RIP and OSPF

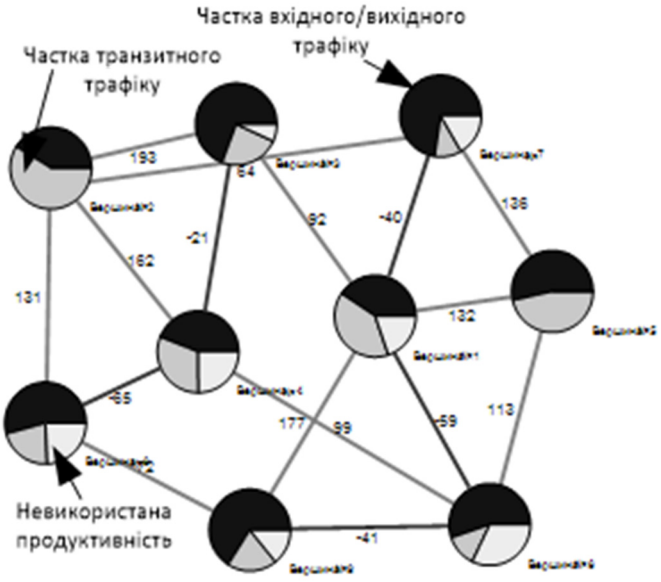
Метою визначення максимальних рівнів сервісного потоку, що передаються в конкретній мережі, є можливість керувати сервісами у верхньому шарі моделі мережі, щоб відповідати вимогам користувача. Наприклад, якщо загальний попит користувачів набагато вищий, ніж досягнута максимальна швидкість сервісного потоку, ми можемо застосувати стратегію визначення пріоритетів, щоб надавати послуги користувачам з високим пріоритетом або адаптувати якість, щоб переконатися, що ми можемо обслуговувати необхідну кількість сеансів у мережі.

На Рис. 3.10 зображено порівняльний аналіз результатів. Логічним подальшим кроком є визначення причин зазначених відмінностей. Наскільки ми

знаємо про маршрутизацію у мережі, це може бути викликано показниками як RIP, так і OSPF для розрахунку найкоротшого шляху. У випадку з OSPF ми використовуємо показники, які враховують пропускну спроможність каналу, а у випадку RIP ми працюємо з числом переходів.



а)



б)

Рис. 3.11. Розподіл мережних ресурсів у логічній структурі, сформованій а) відповідно до протоколу RIP; б) відповідно до протоколу OSPF (чорний – сервісний потік, сірий – транзитний трафік з інших сервісних потоків)

Тому канали (ребра в термінах теорії графів) з низькою пропускною здатністю можуть не використовуватися в разі логічної структури, сформованої OSPF. Давайте перевіримо це припущення.

Співвіднесемо отримані розв'язки завдання оптимізації із структурою мережі, щоб проаналізувати вплив розрахованих сервісних потоків на використання мережних ресурсів.

Передаючи потоки з розрахованими значеннями по досліджуваному фрагменту мережі, в кожному вузлі спостерігається наступне використання ресурсів:

- ресурси, що використовуються вихідним потоком;
- ресурси, що використовуються вхідним потоком;
- ресурси, що використовуються транзитним потоком;
- незадіяні ресурси.

У процесі передавання потоків через мережу з маршрутизацією RIP, розподіл отриманого ресурсу представлено на Рис. 3.11, а.

Сервісний потік з розрахованими значеннями інтенсивності в логічній структурі, сформованій за протоколом RIP, використовує всі доступні мережні ресурси (з точки зору пропускної спроможності), оскільки вузли не містять невикористаних ресурсів. Як показано на Рис. 3.11, вузол №5 бере участь тільки в передаванні трафіку вхідного і вихідного потоків, інші вузли також передають транзитний трафік. Оскільки кожен вузол є ініціатором потоку, наявність транзитного потоку в вузлі обмежує кількість ресурсів, які можна виділити для сервісного потоку у цьому вузлі. Ми помітили, що при збільшенні ступеню вершини графа частка транзитного трафіку в цьому вузлі також зростає (№1, №4, №6). Це відбувається через особливості роботи протоколу RIP. Він розраховує маршрути за критерієм мінімальної кількості переходів, з максимальним використанням вершин з найвищим ступенем. У такому разі ці вершини (центри тяготіння) можуть стати вузькими місцями; отже, використання отриманого розв'язку сформульованого завдання може визначити слабкі сторони при розробці фізичної структури телекомунікаційної мережі. Рекомендовано в

фізичних структурах, що містять вершини з різними ступенями, визначити ті, для яких ступінь максимальний, і забезпечити доступну пропускну здатність сусідніх каналів вищою, ніж інші. Це дозволить забезпечити резервну пропускну здатність для транзитного трафіку та зменшить його вплив на інтенсивність сервісного потоку, яка формується в цьому вузлі.

Сформульоване завдання також розв'язане у випадку формування логічної структури за протоколом OSPF. Отримані результати відрізняються від описаних вище для протоколу RIP. Як видно на Рис. 3.11, б, майже кожен вузол містить невикористані ресурси.

Детальний аналіз показав, що невикористані ресурси присутні в вузлах, суміжні канали яких мають мінімальні пропускні здатності. Вони відхиляються протоколом OSPF під час відбору за станом каналу, як і очікується (ребра, позначені від'ємними вагами на Рис. 3.11, б). Поясненням є те, що протокол OSPF обирає маршрути без урахування інших потоків, які передаються на окремих каналах. Крім того, як показано на Рис. 3.11, б, значна частина транзиту проходить через вузли №2 та №5, оскільки їх сусідні канали мають найбільшу пропускну здатність. Протокол OSPF виявляє зайві канали в фізичній структурі мережі, які можна видалити. З іншого боку, результат показує невідповідність фізичної та логічної структури. Ми рекомендуємо скористатися зваженою багатошляховою маршрутизацією для максимального використання фізичних ресурсів у цьому випадку.

Таким чином, ми спостерігаємо логічну структуру, сформовану протоколом маршрутизації, що накладає значні обмеження на наявність фізичних ресурсів для передавання потоків сервісів.

3.4. Висновки до третього розділу

1. Проведено дослідження конфігурацій мережних топологій на основі теорії графів із їх інтегральним оцінюванням за методом аналізу ієрархій. Розроблено модель, що дає змогу згенерувати всі варіанти конфігурації топологій із заданою кількістю вузлів та здійснити їх порівняльне дослідження за такими метриками, як пропускна здатність та затримка передавання

інформації. Отримані результати дають можливість визначити очікувану ефективність вибору маршрутів передавання даних у деякій мережній структурі та порівняти її з еквівалентними конфігураціями. У процесі вибору конфігурації топології телекомунікаційної мережі врахування інтегральної оцінки забезпечує підвищення ефективності функціонування протоколів динамічної маршрутизації під час її функціонування.

2. Завдання маршрутизації для сучасних телекомунікаційних мереж може бути сформульоване і розв’язане із використанням методів алгебраїчної топології. Запропонований метод має декілька основних властивостей:

– По-перше, скориговані розв'язки завдання маршрутизації створюють також ациклічні потоки.

– По-друге, стаціонарність станів ІТС забезпечено керуванням потоками по замкнутому контуру.

– По-третє, керування потоками по замкнутому контуру може здійснюватися із використанням розподіленого алгоритму. Тобто, кожен вузол має здійснювати оновлення внутрішньої таблиці маршрутизації незалежно від інших вузлів. Цим досягається баланс черг у мережних вузлах.

3. Проведено дослідження архітектури каналу передавання інформаційних потоків із диференціацією за класами обслуговування, кожен із яких має високі вимоги до значень метрик QoS. Визначено процес оптимальної адаптації каналу для підтримки декількох класів обслуговування і отримано залежності якості обслуговування кожного класу потоків у процесі оптимальної адаптації їх інтенсивності. Також, якість сервісу може бути гарантована політикою контролю доступу, оптимільність якої показано у розділі. Це нівелює потребу в динамічній адаптації. Процес обслуговування двох класів інформаційних потоків у трьох дисциплінах досліджено для визначення оптимальних умов їх використання.

4. Запропоновано розв’язок завдання розподілу ресурсів між мережними сервісними потоками за умови рівноправної конкуренції за доступні фізичні ресурси для оцінки максимальної кількості клієнтів, які можуть використовувати

послугу з різними рівнями якості обслуговування. Завдання сформульовано в термінах теорії графів і розв’язано методом лінійного програмування для заданого графа мережі. Досліджено вплив логічної структури на наявність мережевих ресурсів. Результати свідчать, що у випадку формування логічної структури за протоколами RIP потоки в рівній конкуренції досягають своєї максимальної інтенсивності і використовують всі доступні мережеві ресурси. У випадку OSPF, використовуються не всі доступні фізичні ресурси, він оминає канали з мінімальною пропускною здатністю. Цей дає змогу виявити надмірність у фізичній структурі та зменшити вартість мережі або змінити політику маршрутизації шляхом введення k-шляхової маршрутизації через невикористані канали, і як наслідок – підвищивши продуктивність мережі.

РОЗДІЛ 4. МЕТОДИ КЕРУВАННЯ КОНФІГУРАЦІЄЮ ТА ЗАБЕЗПЕЧЕННЯ КАТАСТРОФОСТІЙКОСТІ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМ

4.1. Модель корпоративного клієнта системи надання інформаційно-комунікаційних послуг на базі cloud послуги «програмне забезпечення як сервіс»

Сучасні системи надання інформаційно-телекомунікаційних послуг характеризуються суттєвим зміщенням своїх функцій в бік програмної реалізації, що зумовлено потребою клієнтів отримати адаптивну систему. Ця система має забезпечувати можливість оновлення як функціонального, так і презентаційного. Це означає, що програмна реалізація дає змогу досягати розширення функціоналу системи шляхом зміни програмного коду. Такі зміни можуть стосуватися як аспектів логіки, яка закладена в системі, взаємодії зі сховищами даних, базами даних, так і зміни тих аспектів, які стосуються способу подання системи кінцевому користувачеві.

Враховуючи вищевикладене, легко прийти до висновку, що завдання моделювання поведінки такої системи вкрай складне, а подекуди розв’язку цього завдання не існує взагалі. Це пов’язано із тим, що тривалість розробки адекватної моделі зазвичай перевищує тривалість існування конкретної версії системи, а будь-яка зміна у програмному коді може призвести до змін у поведінці системи, які розроблювана модель просто не здатна врахувати [262].

Розуміння поведінки гетерогенної розподіленої інформаційно-телекомунікаційної системи на прикладному рівні є надзвичайно важливим, оскільки цей рівень забезпечує безпосередній контакт користувача із системою. Якщо розробити модель системи вкрай складно або й узагалі неможливо, то єдиним способом оцінки її поведінки залишається експериментальне дослідження. Планування експерименту для дослідження поведінки складної інформаційно-телекомунікаційної системи є складним та нетривіальним завданням. Попри сам метод дослідження поведінки, окремої уваги потребують

також компоненти системи, які підлягають дослідженню, із урахуванням того, що сама система повинна бути неперервно доступною для клієнтів та кінцевих користувачів.

Для проведення експериментального дослідження поведінки таких систем, у роботі здійснено розроблення моделі корпоративного клієнта, структурна схема якої представлена на Рис. 4.1.

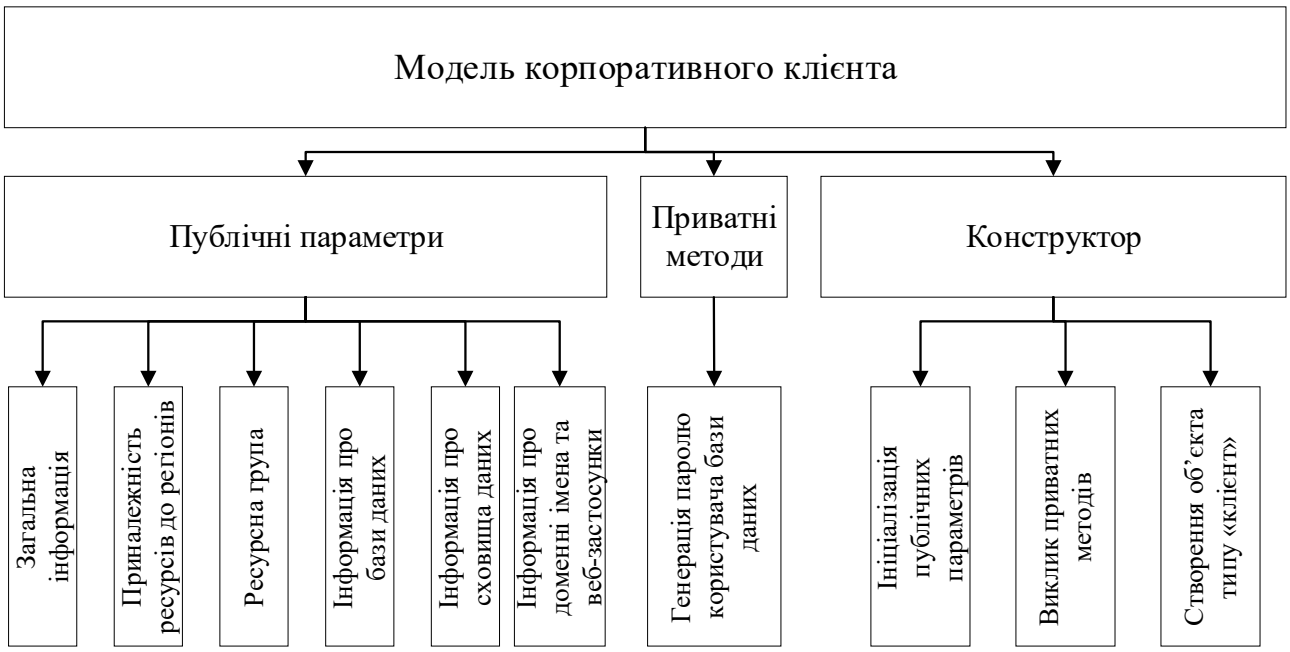


Рис. 4.1. Структурна схема моделі корпоративного клієнта розподіленої гетерогенної інформаційно-комунікаційної системи надання послуг

Модель являє собою набір властивостей та приватних методів, які ініціалізуються викликом конструктора. Сама модель реалізована у вигляді програмного коду із використанням мови програмування C#. Вона дає змогу відтворити структурно та функціонально поведінку реального корпоративного клієнта розподіленої інформаційно-комунікаційної системи. Оскільки модель є адаптивною, то вона забезпечує можливість змінної конфігурації ресурсів клієнтів, що, у свою чергу, дає можливість моделювати різних характер впливів різного роду користувачів на поведінку досліджуваної системи.

Назвемо цю модель програмною, оскільки вона дає змогу із використанням засобів мови програмування створювати віртуальні структурні образи реальних клієнтів інформаційно-комунікаційної системи. Модель базується на принципі

агрегування функцій корисності, який сформульовано у другому розділі дисертації. Ця модель ляже в основу експериментального процесу навантажувального тестування, який описано та обґрунтовано у шостому розділі роботи. Модель використовує технологію імітаційного моделювання систем.

У результаті ініціалізації моделі набором вхідних параметрів формується програмний об’єкт, який володіє всіма властивостями програмної моделі із конкретною конфігурацією, яка відповідає типу корпоративного клієнта системи. Зокрема, ідеться про розміщення cloud ресурсів клієнта, наявність чи відсутність геореплікації даних клієнта, параметри бази даних та сховища даних клієнта, розміщення приватної інформації клієнта, яка забезпечує можливість решті компонентів розподіленої інформаційно-телекомунікаційної системи зі спільною інфраструктурою взаємодіяти із ресурсами клієнта. Важливою особливістю цієї моделі є те, що її використання забезпечує не лише ініціалізацію об’єкта, який характеризує клієнта, але і з його використанням забезпечити розгортання cloud ресурсів клієнта на базі cloud провайдера.

Таким чином досягається можливість досліджувати поведінку складної розподіленої гетерогенної інформаційно-телекомунікаційної інфраструктури без використання ресурсів реальних клієнтів системи для створення навантаження на її компоненти. Це дає змогу уникнути небажаного завантаження ресурсів клієнта, а також убезпечує систему від загроз безпеці даних кінцевих користувачів.

Тестові клієнти, створені на підставі запропонованої моделі, використані у процесі навантажувального тестування з метою дослідження поведінки інформаційно-телекомунікаційної системи, результати якого наведено у шостому розділі роботи.

У результаті застосування об’єкта, що представляє клієнтські ресурси, до cloud інфраструктури, утворюється набір таких ресурсів: сервери бах даних відповідно до кількості регіонів, у яких розміщено дані користувача, відповідні репліки баз даних на цих серверах, відмовостійка група серверів баз даних, що забезпечує безперебійну роботу системи у разі недоступності ресурсів клієнта у

первинному регіоні, сховище даних користувача, де зберігаються пов’язані із записами у базі даних артефакти (здебільшого це медіаконтент або ж аналітичні дані, які збирає система для подальшої оптимізації), набір пов’язаних із корпоративним клієнтом доменних імен спільних cloud ресурсів, здійснюється конфігурування приватних даних, які використовуються іншими спільних ресурсами для комунікації із призначеними ресурсами клієнта. Варто відзначити, що у випадку розміщення ресурсів клієнта тільки у одному регіоні, група відмовостійкості не ініціалізується, що означає відсутність реплік бази даних та, відповідно, зменшення навантаження від такого клієнта на інформаційну та телекомунікаційну складові загальної системи.

Модель розроблена вперше, а її застосування дає змогу досягти підвищення ефективності методу експериментального дослідження поведінки складної інформаційно-телекомунікаційної інфраструктури шляхом навантажувального тестування за рахунок врахування впливу на систему користувачів із різною конфігурацією ресурсів.

4.2. Модель спільної Cloud інфраструктури системи надання інформаційно-комунікаційних послуг

Для того, щоб врахувати у моделі корпоративного клієнта структурно-функціональні параметри спільної cloud інфраструктури, у роботи вперше розроблено її програмну модель. Під програмною моделлю спільної cloud інфраструктури будемо розуміти програмний код, написаний на мові програмування C#, який дає змогу генерувати екземпляри спільних cloud ресурсів, інформація про які необхідна у процесі розгортання клієнтських ресурсів у розподілених гетерогенних інформаційно-телекомунікаційних системах. Оскільки інфраструктура базується на принципі мультиклієнтського використання (орендування), то деякі її елементи підлягають спільному використанню множиною корпоративних клієнтів. Кожен корпоративний клієнт, в свою чергу, включає набір кінцевих користувачів, тим самим реалізуючи сформульований раніше принцип агрегації функцій корисності. Тобто сумарна корисність послуги для всіх користувачів є корисністю послуги для клієнта.

Спільна cloud інфраструктура (Рис. 4.2) являє собою набір таких компонентів: веб-застосунки, спільні системні бази даних, сховища приватних даних клієнтів, спільна зона сервісу доменних імен. Оскільки спільна інфраструктура реалізує принцип географічної надлишковості, то в моделі реалізовано також балансувальники трафіку, які відповідають за спрямування запитів користувачів до застосунків у відповідному регіоні за критерієм максимальної продуктивності (мінімального часу очікування відповіді від сервера застосунку).

Подібно до програмної моделі корпоративного клієнта, програмна модель спільної інфраструктури ініціалізується набором вхідних параметрів. Результатом її ініціалізації є створення програмного об’єкту спільної інфраструктури, властивості якого використовуються у процесі розгортання інфраструктури корпоративного клієнта у розподіленій гетерогенній інформаційно-телекомунікаційній системі. Запропонована програмна модель базується на технології імітаційного моделювання. Вона відрізняється від відомих здатністю адаптуватися під інфраструктурні зміни шляхом інтелектуального формування властивостей об’єкта, що представляє ресурси спільної cloud інфраструктури. Ця здатність дає змогу враховувати інфраструктурні зміни спільних ресурсів, як, наприклад, зміну кількості спільних веб-застосунків або перехід на іншу зону сервісу доменних імен у процесі створення ресурсів корпоративного клієнта, що дає змогу урізноманітнити та додатково деталізувати метод дослідження поведінки таких складних систем на основі навантажувального тестування, результати якого представлено у шостому розділі роботи [262].

Представлена модель спільної cloud інфраструктури базується на принципі функцій вартості, який сформульовано у другому розділі роботи. Йдеться про те, що спільна інфраструктура зазнає впливу множини клієнтів у процесів користування їх сервісами, що надає досліджувана розподілена інформаційно-комунікаційна система. Функція вартості дає змогу оцінити ступінь впливу

кожного окремого клієнта на зниження продуктивності спільної інфраструктури для нових клієнтів.

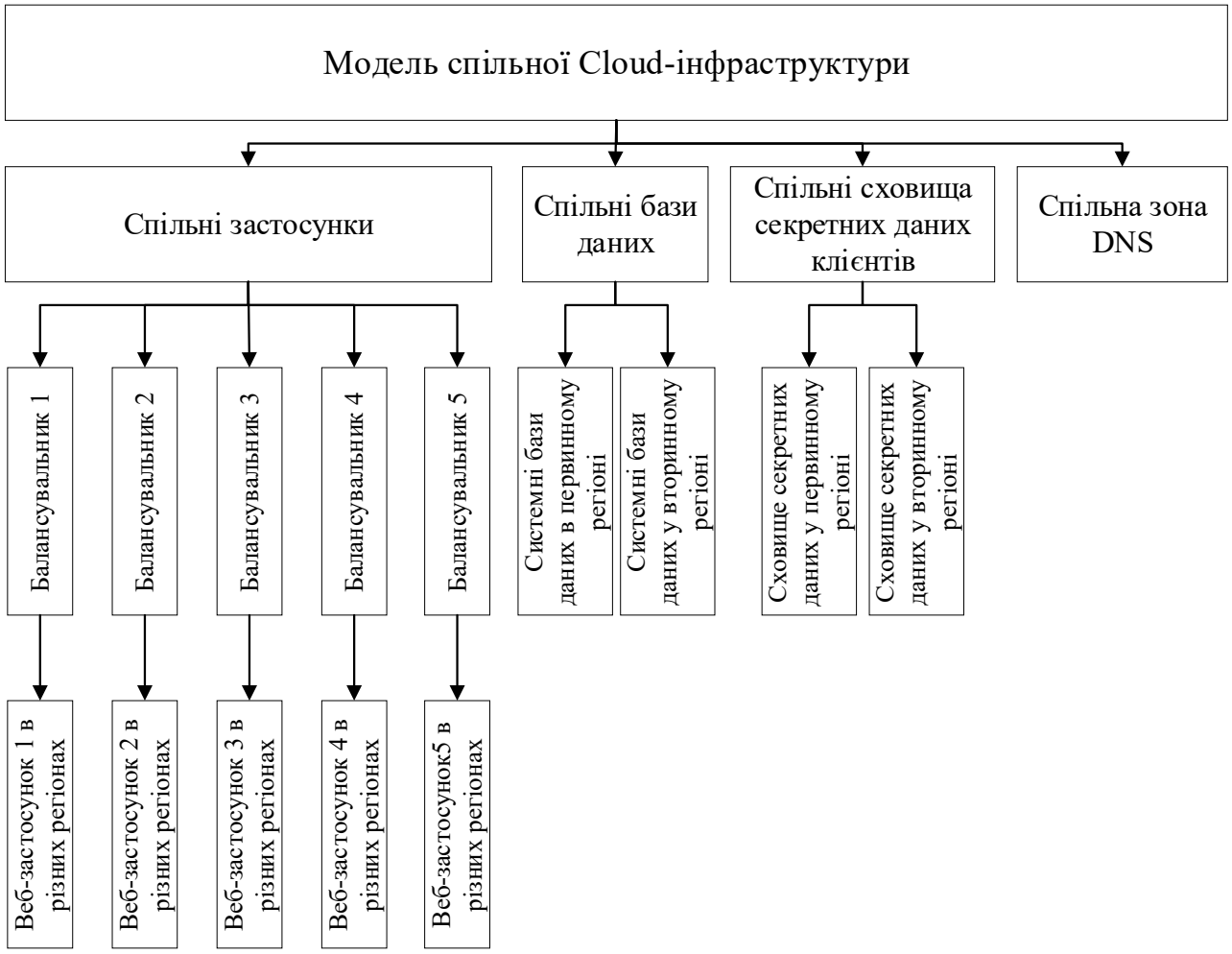


Рис. 4.2. Структурна схема моделі спільної інфраструктури розподіленої гетерогенної інформаційно-комунікаційної системи надання послуг

У сукупності описані вище дві моделі дають змогу узагальнити та описати процес комунікації між множиною користувачів та cloud ресурсів на рівні надання послуг. Не слід забувати, що при цьому особливої уваги заслуговують питання безпеки даних користувачів та катастрофостійкості розподілених інформаційно-телекомунікаційних систем надання послуг.

4.3. Дослідження ефективності функціонування системи інформаційної безпеки в cloud системах типу IaaS

Нові розподілені ІКС і розвиток глобальних систем зберігання та обробки даних ставлять високі вимоги до захисту інформації і системи інформаційної

безпеки в цілому. Під захистом інформації будемо розуміти сукупність заходів, як правових, адміністративних, організаційних, так і технічних, які спрямовані на збереження і дотримання цілісності інформації, а також встановлюють відповідний порядок доступу до інформації [14, 205].

Атаки можуть виступати у якості загроз інформації безпеки ІКС, і можуть мати за ціль апаратні засоби (атаки типу «відмова в обслуговуванні»), а також і інформаційні ресурси системи (атака типу «людина посередині»). Особливість полягає у тому, що природа атаки є випадковою. Отже, необхідним елементом системи захисту є наявність експериментальних даних щодо впливу атак на ІКС. Ці дані можна отримати шляхом проведення імітаційного моделювання.

Через велику кількість існуючих загроз [34] розроблено чимало засобів їх передбачення та виявлення. Описано способи реаліазії моделей поведінки зловмисника та моделей загроз на основі аномального трафіку. Ці моделі мають в основі формалізоване представлення суб'єкта атаки та рівнів впливу атак, які ним ініційовані. Завдяки такому апарату є можливість автоматично формувати перелік факторів зловмисного впливу та системних вразливостей, які спричиняють саму можливіть атаки. Відповідно до цієї інформації розробляється і застосовується політика безпеки.

Відомий метод та алгоритм побудови моделі атаки із застосуванням теорії графів для складної ІКС, який дає змогу отримати представлені у вигляді графа впливи атак (ребра графа) на вузли мережі (вершини графа). Найкоротший шлях між парою вузлів, де вузол-витік являє собою зловмисника, несе найбільшу загрозу інформаційній безпеці [66, 67].

Безумовні переваги згаданих методів полягають у їх теоретичній обґрунтованості, однак, в основному, відсутні результати практичних досліджень, що базуються на експериментах. Теоретичні методи, що сприяють підвищенню інформаційної безпеки, доволі часто не можуть бути повною мірою застосовані на практиці, здебільшого через складність їх адаптації до наявних прогармно-апаратних засобів IaaS. З урахуванням проведеного аналізу, у даному розділі запропоновано експериментальне дослідження та надано практичні

рекомендації підвищення ступеню інформаційної безпеки, які базуються на результатах експерименту із дослідженням впливу атак типу «відмова в обслуговуванні» та «людина посередині» на модель фрагменту ІКС із застосуванням cloud-сервісу типу IaaS.

Запропоновано модель фрагменту ІКС в системі GNS3, компонентами якої є мережний екран Cisco ASA, маршрутизатор Cisco 3640, комутатор Cisco Catalyst 2960 та кінцевих споживачі, які реалізовані як віртуальні машини з використанням програмного забезпечення VMware 8 на базі операційних систем Windows XP, Windows 7 та Kali Linux.

Проведено моделювання DoS-атаки на веб-сервер хмарної ІКС для досягнення стану недоступності обчислювальних ресурсів, а також змодельовано атаку типу «людина посередині» з метою перехоплення облікових даних для автентифікації в соціальній мережі. Такі атаки реалізовано на основі технологій SYN-flood і ARP-spoofing, відповідно. За результатами дослідження процесу та впливу атак запропоновано комплекс заходів щодо обмеження наслідків атак для ІКС та показано технічний ефект від впродження цих рекомендацій для cloud систем IaaS.

Отже, на Рис. 4.3 подано модель фрагменту ІКС, на основі якої виконано дослідження впливу атаки типу «відмова в обслуговуванні» [19].

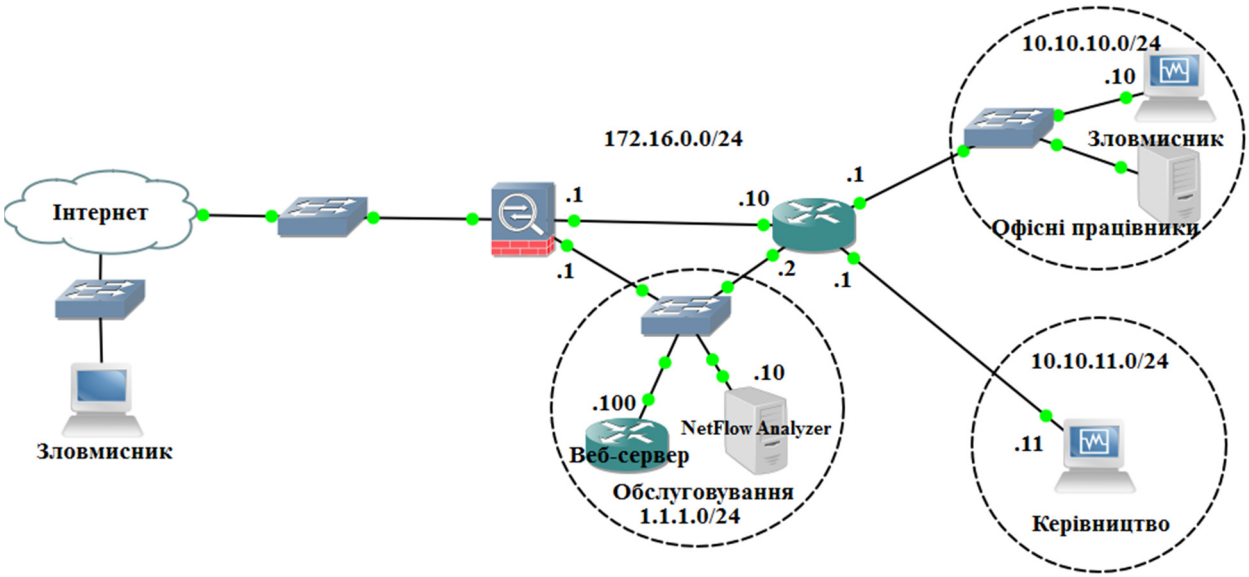


Рис. 4.3. Модель фрагменту ІКС, яка досліджується

У запропонованій моделі ІКС розділено на зовнішній (від постачальника послуг до мережного екрану), внутрішній (фрагмент системи всередині закритої спільноти) та демілітаризований (де мають розташовуватись компонети, доступні віддалено) сегменти. Відповідна конфігурація у синтаксисі мови Cisco IOS наведена нижче:

```
ASA(config)# interface gi0
ASA(config-if)# ip address dhcp
ASA(config-if)# nameif outside
ASA(config-if)# security-level 0
ASA(config)# interface gi1
ASA(config-if)# ip address 172.16.0.1 255.255.255.0
ASA(config-if)# nameif inside
ASA(config-if)# security-level 100
ASA(config)# interface gi2
ASA(config-if)# ip address 1.1.1.1 255.255.255.0
ASA(config-if)# nameif dmz
ASA(config-if)# security-level 50
```

Модель ІКС типу IaaS, представлена на Рис. 4.3, реалізована із використанням таких апаратних засобів:

- ядро системи релізоване із використанням маршрутизатора Cisco 3640 з чотирьма відділами NM-1FE-TX для портів FastEthernet. Маршрутизатор оснащено операційною системою IOS версії 12.4(13);
- у якості основного елемента інформаційної безпеки використано мережний екран Cisco ASA 5520, оснащений операційною системою Cisco Adaptive Security Appliance Software;
- під’єднання внутрішнього сегменту користувачів до системи здійснюється через комутатор Cisco Catalyst 2960;
- системає надає інформаційні послуги через маршрутизатор Cisco c7200 із ввімкненим http сервером;
- обладнання користувача реалізоване на віртуальних машинах. Сервер аналізу інформаційних потоків оснащених програмним забезпеченням NetFlow Analyzer розміщено на віртуальній машині під управлінням операційної системи Windows Server 2008 R2;

– обладнання зловмисника реалізоване на фізичній робочій станції під управлінням операційної системи Kali Linux та оснащене програмним забезпеченням Net Tools. Комунікація з основною ІКС здійснюється через системний інтерфейс-петлю.

В реалізованій ІКС забезпечено підключення до Інтернету, оскільки це додатковий фактор адекватності здійсненого моделювання. Спосіб підключення доступу до зовнішньої мережі для системи моделювання GNS3 полягає у такому:

– слід утворити системний інтерфейс-петлю на фізичній машині, де встановлено систему GNS3;
– слід надати спільний доступ до мережного інтерфейсу, який забезпечує доступ фізичної машини до Інтернету і ввімнути створений системний інтерфейс-петлю;
– у налаштування Cloud елементу системи моделювання GNS3 у якості інтерфейсу слід обрати новостворений інтерфейс-петлю;
– після цього Cloud елемент стає пунктом доступу до Інтернету для довільного елемента моделі в GNS3.

Моніторинг процесів у запропонованій моделі IaaS ІКС є однією з основних вимог інформаційної безпеки. У зв’язку з цим, збір та обробка статистики мережних процесів здійснюється на окремому сервері із програмним забезпеченням NetFlow Analyzer, який в своїй основі має протокол NetFlow, який широкі можливості аналізу мережних процесів. Приклад на Рис. 4.4 демонструє аналіз мережних процесів на мережному екрані.

Атаку типу «відмова в обслуговуванні» організовано на сегмент обслуговування у демілітаризованій зоні шляхом надсилання SYN-пакетів із зовнішньої мережі. Для реалізації SYN-flood процесу використано програмне забезпечення «Net tools», яким оснащено обладнання зловмисника.

Атака в приватному середовищі типу «людина посередині» організована для перехоплення приватної інформації, яка необхідна для автентифікації в соціальній мережі. У цьому експеримента на обладнанні зловмисника встановлено операційну систему Kali Linux.

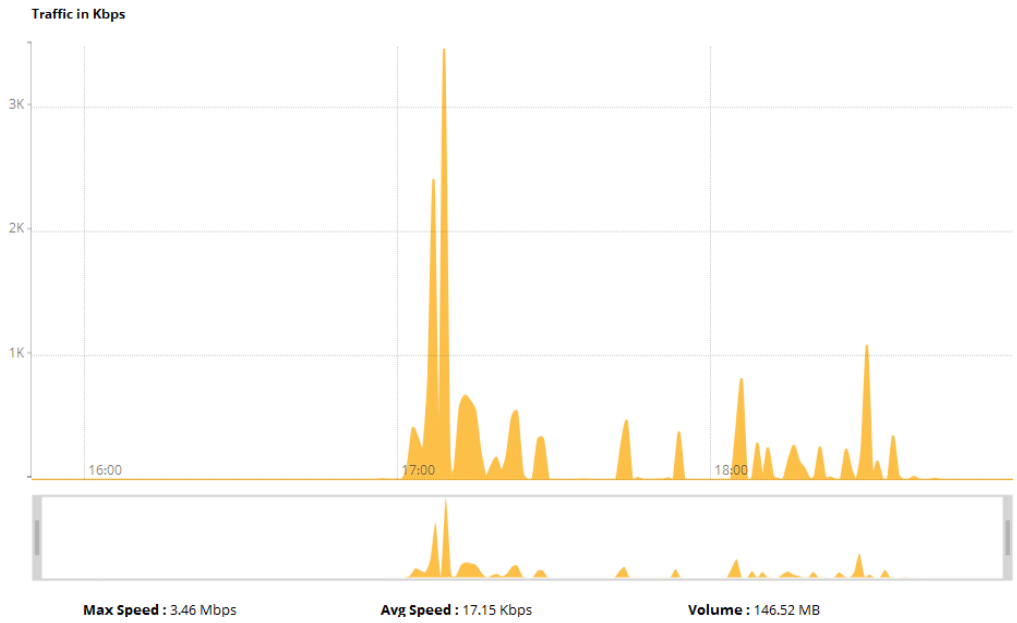


Рис. 4.4. Аналіз процесу передавання трафіку через мережний екран

Прослуховування даних реалізовано на основі ARP-spoofing. Це процес дає змогу представити пристрій зловмисника як мережний шлюз. Таким чином, інформація передається від пристрою користувача до реального шлюза через пристрій зловмисника. У зворотному напрямку передавання відбувається аналогічним чином. Оскільки для передавання використовується захищений протокол HTTPs, то використаємо інструмент, який забезпечує перехоплення автентифікаційних даних звичайних користувачів у відкритому вигляді. Також подано спосіб під'єднання до активної комунікаційної сесії черз перехоплення файлів coockie. Відзначимо, що coockie містять параметр remixsid, який ідентифікує права користувача в межах активної сесії, і завдяки нявності цього параметра є можливість під'єднання до такої сесії. Важливо врахувати, що доступ завдяки відомому параметру remixsid є тільки із множини IP-адрес, які звиайний користувач раніше використовував для доступу до досліджуваного інформаційного ресурсу. В досліджуваній ІТС доступ до публічної мережі здійснюється шляхом перетворення IP-адрес із перевантаженням (PAT) на мережному екрані, тому, фактично, звичайний користувач і зловмисник використовують одну множину публічних IP-адрес.

На основі проведеного дослідження запропонуємо заходи щодо послаблення ступеню впливу атак на ІТС.

Нижче запропоновано алгоритм реалізації атаки типу «відмова в обслуговуванні»:

Етап 1. Підготовка інструментарію атаки. Налаштування інструменту «Net tools» на пристрої зловмисника, перевірка доступності веб-сервера з цього інструменту за його зовнішньою адресою (192.168.137.100), що перетворюється у внутрішню адресу (1.1.1.100). Далі реалізуємо перехоплення трафіку з використанням програми-аналізатора Wireshark на мережному екрані (його інтерфейсах):

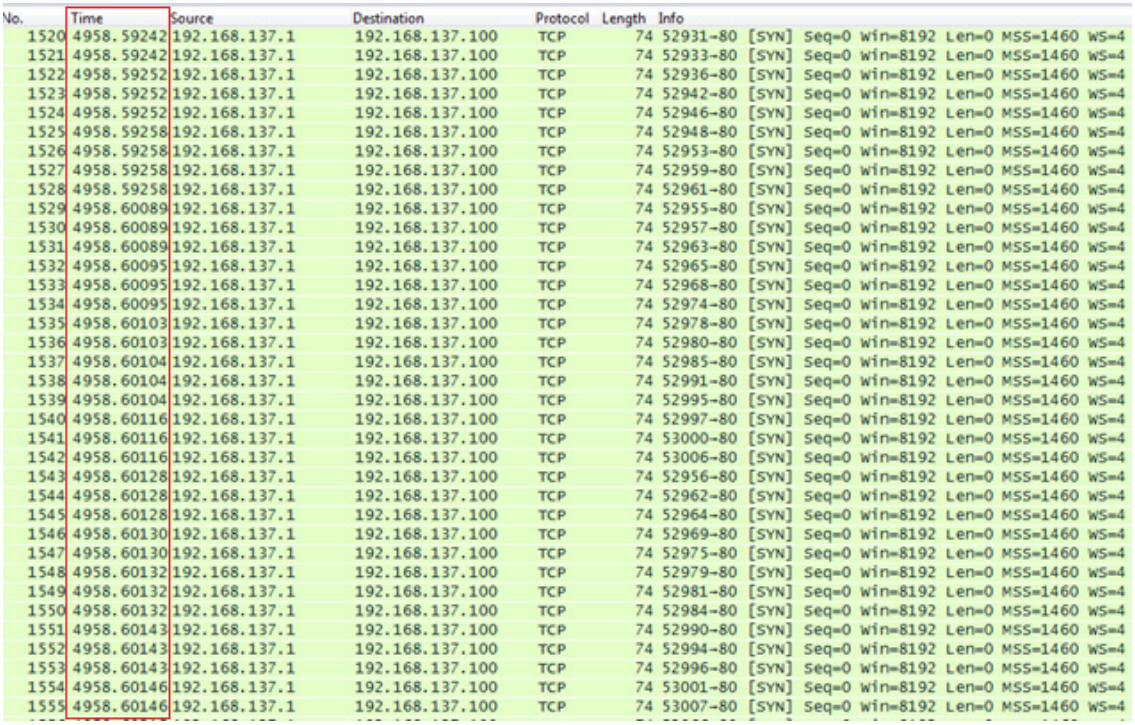
- GigabitEthernet 0, який є інтерфейсом комунікації із зовнішньою мережею;
- GigabitEthernet 2, який є інтерфейсом комунікції із сегментом обслуговування.

Етап 2. Ініціація та процес атаки. Для реалізації атаки використано мережний інструмент Http flooder (DOS) і програмне забезпечення «Net tools», у якому вказано адресу сервера (192.168.137.100), порт сервісу, на який відбувається атака (80) і число SYN-пакетів (300), які використано для завантаження запитами сервера. Результати аналізу трафіку після початку атаки інструментом Wireshark на інтерфейсі зовнішньої мережі та на інтерфейсі сегменту обслуговування представлено на Рис. 4.5, Рис. 4.6.

Далі подано алгоритм реалізації атаки типу «людина посередині»:

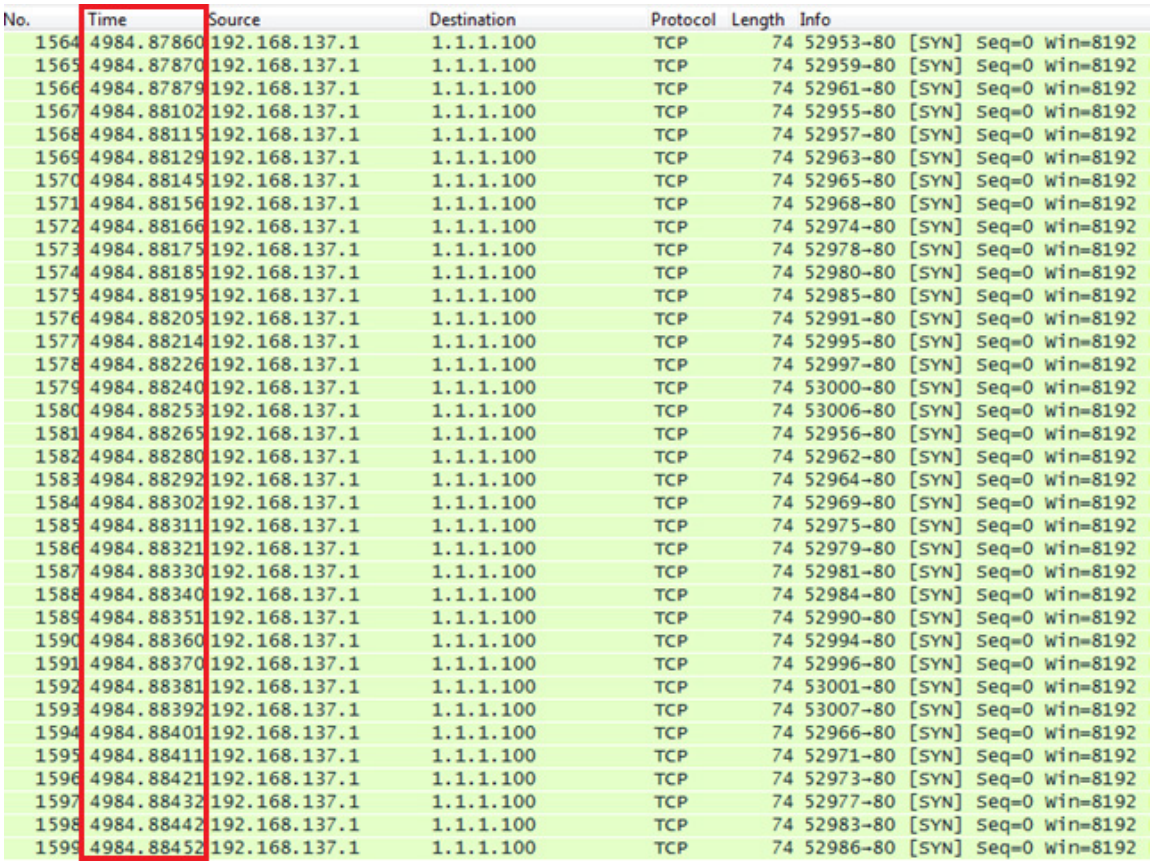
Етап 1. Підготовка атаки. Нехай пристрій зловмисника належить до підмережі 10.10.10.0/24, і уже підключений до мережі та перебуває в активному стані. Вмикаємо на пристрої зловмисника можливість переспрямування трафіку, щоб забезпечити йому комунікацію як із шлюзом мережі 10.10.10.1/24, так і пристроєм користувача 10.10.10.10/24.

Етап 2. Підготовка пристрою звичайного користувача. Для того, щоб реалізувати процеси ARP-spoofing і sniffing, використаємо програмне забезпечення Ettercap, попередньо запущене і відповідно налаштоване. Його завданням є створення помилкової при’язки IP-адреси шлюзу та MAC-адреси пристрою зловмисника у ARP-таблиці на пристрої звичайного користувача. Відповідно, ці ж дії слід виконати і у ARP-таблиці шлюза.



No.	Time	Source	Destination	Protocol	Length	Info
1520	4958.59242	192.168.137.1	192.168.137.100	TCP	74	52931→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1521	4958.59242	192.168.137.1	192.168.137.100	TCP	74	52933→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1522	4958.59252	192.168.137.1	192.168.137.100	TCP	74	52936→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1523	4958.59252	192.168.137.1	192.168.137.100	TCP	74	52942→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1524	4958.59252	192.168.137.1	192.168.137.100	TCP	74	52946→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1525	4958.59258	192.168.137.1	192.168.137.100	TCP	74	52948→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1526	4958.59258	192.168.137.1	192.168.137.100	TCP	74	52953→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1527	4958.59258	192.168.137.1	192.168.137.100	TCP	74	52959→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1528	4958.59258	192.168.137.1	192.168.137.100	TCP	74	52961→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1529	4958.60089	192.168.137.1	192.168.137.100	TCP	74	52955→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1530	4958.60089	192.168.137.1	192.168.137.100	TCP	74	52957→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1531	4958.60089	192.168.137.1	192.168.137.100	TCP	74	52963→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1532	4958.60095	192.168.137.1	192.168.137.100	TCP	74	52965→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1533	4958.60095	192.168.137.1	192.168.137.100	TCP	74	52968→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1534	4958.60095	192.168.137.1	192.168.137.100	TCP	74	52974→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1535	4958.60103	192.168.137.1	192.168.137.100	TCP	74	52978→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1536	4958.60103	192.168.137.1	192.168.137.100	TCP	74	52980→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1537	4958.60104	192.168.137.1	192.168.137.100	TCP	74	52985→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1538	4958.60104	192.168.137.1	192.168.137.100	TCP	74	52991→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1539	4958.60104	192.168.137.1	192.168.137.100	TCP	74	52995→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1540	4958.60116	192.168.137.1	192.168.137.100	TCP	74	52997→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1541	4958.60116	192.168.137.1	192.168.137.100	TCP	74	53000→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1542	4958.60116	192.168.137.1	192.168.137.100	TCP	74	53006→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1543	4958.60128	192.168.137.1	192.168.137.100	TCP	74	52956→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1544	4958.60128	192.168.137.1	192.168.137.100	TCP	74	52962→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1545	4958.60128	192.168.137.1	192.168.137.100	TCP	74	52964→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1546	4958.60130	192.168.137.1	192.168.137.100	TCP	74	52969→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1547	4958.60130	192.168.137.1	192.168.137.100	TCP	74	52975→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1548	4958.60132	192.168.137.1	192.168.137.100	TCP	74	52979→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1549	4958.60132	192.168.137.1	192.168.137.100	TCP	74	52981→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1550	4958.60132	192.168.137.1	192.168.137.100	TCP	74	52984→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1551	4958.60143	192.168.137.1	192.168.137.100	TCP	74	52990→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1552	4958.60143	192.168.137.1	192.168.137.100	TCP	74	52994→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1553	4958.60143	192.168.137.1	192.168.137.100	TCP	74	52996→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1554	4958.60146	192.168.137.1	192.168.137.100	TCP	74	53001→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4
1555	4958.60146	192.168.137.1	192.168.137.100	TCP	74	53007→80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4

Рис. 4.5. Відображення зловмисних запитів під час здійснення DoS-атаки на інтерфейсі зовнішньої мережі в системі IaaS



No.	Time	Source	Destination	Protocol	Length	Info
1564	4984.87860	192.168.137.1	1.1.1.100	TCP	74	52953→80 [SYN] Seq=0 Win=8192
1565	4984.87870	192.168.137.1	1.1.1.100	TCP	74	52959→80 [SYN] Seq=0 Win=8192
1566	4984.87879	192.168.137.1	1.1.1.100	TCP	74	52961→80 [SYN] Seq=0 Win=8192
1567	4984.88102	192.168.137.1	1.1.1.100	TCP	74	52955→80 [SYN] Seq=0 Win=8192
1568	4984.88115	192.168.137.1	1.1.1.100	TCP	74	52957→80 [SYN] Seq=0 Win=8192
1569	4984.88129	192.168.137.1	1.1.1.100	TCP	74	52963→80 [SYN] Seq=0 Win=8192
1570	4984.88145	192.168.137.1	1.1.1.100	TCP	74	52965→80 [SYN] Seq=0 Win=8192
1571	4984.88156	192.168.137.1	1.1.1.100	TCP	74	52968→80 [SYN] Seq=0 Win=8192
1572	4984.88166	192.168.137.1	1.1.1.100	TCP	74	52974→80 [SYN] Seq=0 Win=8192
1573	4984.88175	192.168.137.1	1.1.1.100	TCP	74	52978→80 [SYN] Seq=0 Win=8192
1574	4984.88185	192.168.137.1	1.1.1.100	TCP	74	52980→80 [SYN] Seq=0 Win=8192
1575	4984.88195	192.168.137.1	1.1.1.100	TCP	74	52985→80 [SYN] Seq=0 Win=8192
1576	4984.88205	192.168.137.1	1.1.1.100	TCP	74	52991→80 [SYN] Seq=0 Win=8192
1577	4984.88214	192.168.137.1	1.1.1.100	TCP	74	52995→80 [SYN] Seq=0 Win=8192
1578	4984.88226	192.168.137.1	1.1.1.100	TCP	74	52997→80 [SYN] Seq=0 Win=8192
1579	4984.88240	192.168.137.1	1.1.1.100	TCP	74	53000→80 [SYN] Seq=0 Win=8192
1580	4984.88253	192.168.137.1	1.1.1.100	TCP	74	53006→80 [SYN] Seq=0 Win=8192
1581	4984.88265	192.168.137.1	1.1.1.100	TCP	74	52956→80 [SYN] Seq=0 Win=8192
1582	4984.88280	192.168.137.1	1.1.1.100	TCP	74	52962→80 [SYN] Seq=0 Win=8192
1583	4984.88292	192.168.137.1	1.1.1.100	TCP	74	52964→80 [SYN] Seq=0 Win=8192
1584	4984.88302	192.168.137.1	1.1.1.100	TCP	74	52969→80 [SYN] Seq=0 Win=8192
1585	4984.88311	192.168.137.1	1.1.1.100	TCP	74	52975→80 [SYN] Seq=0 Win=8192
1586	4984.88321	192.168.137.1	1.1.1.100	TCP	74	52979→80 [SYN] Seq=0 Win=8192
1587	4984.88330	192.168.137.1	1.1.1.100	TCP	74	52981→80 [SYN] Seq=0 Win=8192
1588	4984.88340	192.168.137.1	1.1.1.100	TCP	74	52984→80 [SYN] Seq=0 Win=8192
1589	4984.88351	192.168.137.1	1.1.1.100	TCP	74	52990→80 [SYN] Seq=0 Win=8192
1590	4984.88360	192.168.137.1	1.1.1.100	TCP	74	52994→80 [SYN] Seq=0 Win=8192
1591	4984.88370	192.168.137.1	1.1.1.100	TCP	74	52996→80 [SYN] Seq=0 Win=8192
1592	4984.88381	192.168.137.1	1.1.1.100	TCP	74	53001→80 [SYN] Seq=0 Win=8192
1593	4984.88392	192.168.137.1	1.1.1.100	TCP	74	53007→80 [SYN] Seq=0 Win=8192
1594	4984.88401	192.168.137.1	1.1.1.100	TCP	74	52966→80 [SYN] Seq=0 Win=8192
1595	4984.88411	192.168.137.1	1.1.1.100	TCP	74	52971→80 [SYN] Seq=0 Win=8192
1596	4984.88421	192.168.137.1	1.1.1.100	TCP	74	52973→80 [SYN] Seq=0 Win=8192
1597	4984.88432	192.168.137.1	1.1.1.100	TCP	74	52977→80 [SYN] Seq=0 Win=8192
1598	4984.88442	192.168.137.1	1.1.1.100	TCP	74	52983→80 [SYN] Seq=0 Win=8192
1599	4984.88452	192.168.137.1	1.1.1.100	TCP	74	52986→80 [SYN] Seq=0 Win=8192

Рис. 4.6. Відображення зловмисних запитів під час здійснення DoS-атаки на інтерфейсі сегменту обслуговування в системі IaaS

Етап 3. Ініціація атаки та перехоплення автентифікаційних даних звичайного користувача. Використано програмне забезпечення Sslstrip з метою захоплення конфіденційних даних, переданих по протоколу https. За результатами успішної авторизації звичайного користувача в соціальній мережі зловмисник отримав авфентифікаційну інформацію (Рис. 4.7).

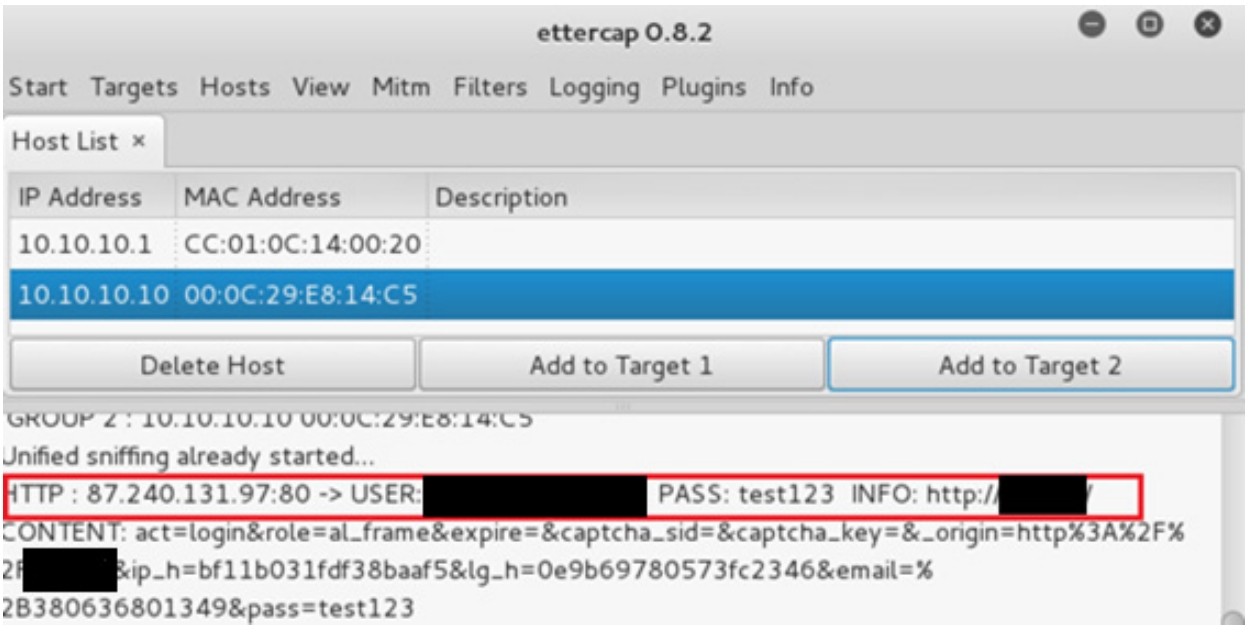


Рис. 4.7. Результати отримання зловмисником автентифікаційних даних звичайного користувача у результаті успішної атаки «людина посередині»

На Рис. 4.7 представлено результат перехоплення cookie та підключення до активної сесії звичайного користувача у ІТС, що представлена як соціальна мережа.

Запропонований алгоритм потребує постійного перехоплення даних в ІТС, оскільки момент активізації сесії звичайного користувача невідомий наперед, а наявність великої кількості ARP-запитів у системі може спровокувати тривогу у системі моніторингу. Навіть вибір вдалого моменту не гарнтує успішності атаки, оскільки використане програмне забезпечення sslstrip підмінює протокол https на http, і веб-переглядач користувача може попередити його про непідтверджений сертифікат веб-сторінки, що повинно викликати підозру і припущення про активний процес атаки. Тому запропоновано альтернативний алгоритм реалізації атаки типу «людина посередині», який відрізняється від попереднього

відсутністю потреби дізнаватися автентифікаційні дані користувача. Цей альтернативний алгоритм дає змогу напряму підключитися до активної сесії звичайного користувача. Кроки алгоритму такі:

Етап 1. Налаштування інструменту Ettercap аналогічно до Етапу 2 попереднього алгоритму.

Етап 2. Фільтрація перехопленого аналізатором Wireshark трафіку за параметром:

```
http.cookie contains remixsid
```

Етап 3. Отримання значення цього параметру із перехопленого cookie.

Етап 4. Зловмисник авторизується у цільовій соціальній мережі з довільної сторінки. У веб-переглядачі використовується плагін Edit cookie для підміни значення перехопленого remixsid у активній сесії зловмисника. Після оновлення веб-сторінки зловмисник отримує доступ до особистої сторінки звичайного користувача.

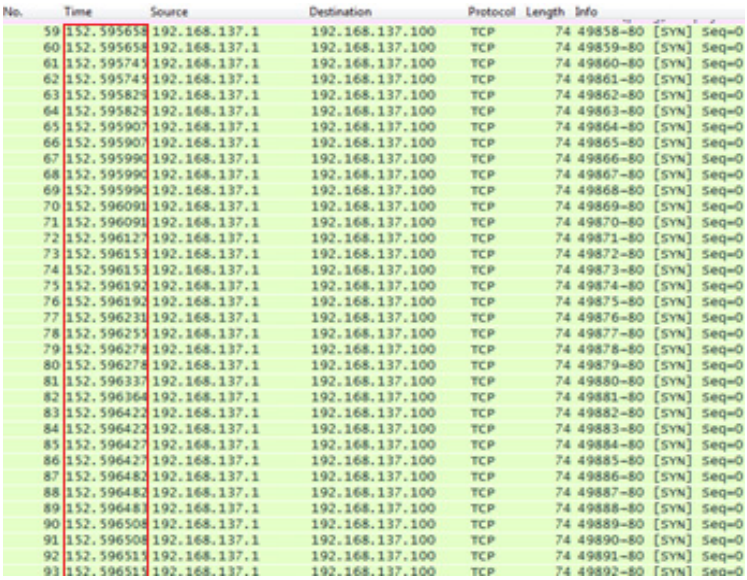
У роботі запропоновано метод зменшення ступеня впливу DoS-атаки на ІТС, який полягає в підвищенні рівня захищеності мережного екрану шляхом обмеження кількості напіввідкритих з'єднань в трьохетапному процесі встановлення TCP-з'єднання. Ідея полягає у тому, що коли число напіввідкритих з'єднань досягне заданого порогу, то мережний екран сам розпочне надсилати відповіді на SYN-запити SYN/ACK-пакетами, уникаючи потрапляння у внутрішній сегмент ІТС цих пакетів, які за вказаних умов вважатимуться зловмисними. Якщо зовнішній пристрій, з якого надійшов запит, відповість ACK-пакетом, тоді мережний екран знатиме, що початковий запит не є елементом можливої SYN-атаки. Для реалізації вищезгаданого підходу використаємо політику Modular Policy Frame, який дає змогу обмежити число напіввідкритих з'єднань із сервером.

Нижче подано конфігурацію мережного екрану, яка дає змогу зменшити ступінь впливу DoS-атаки на ІТС шляхом обмеження відкритих (активних) з'єднань числом 100, напіввідкритих з'єднань – 200 і напіввідкритих з'єднань від конкретного користувача (за його адресою) – 10, аж до повної її зупинки. Також

зменшенню ступеня впливу атаки сприяє встановлення максимальної тривалості з'єднань: для відкритого (активного) з'єднання – на 2 год, для напіввідкритого з'єднання – на 45 секунд і для напівзакритого з'єднання – 25 хв.

```
ASA(config)# class-map tcp_syn – створення спеціального класу tcp_syn.
ASA(config-cmap)# match port tcp eq 80 – віднесення до нього трафіку, що надходить
                                    на порт 80 по протоколу TCP.
ASA(config-cmap)# exit
ASA(config)# policy-map global_policy – відкриття створеної при базових
                                    налаштуваннях політики global_policy.
ASA(config-pmap)# class tcp_syn – віднесення до політики попередньо створеного
                                  класу.
ASA(config-pmap-c)# set connection conn-max 100 – встановлення максимальної
                                              кількості з'єднань.
ASA(config-pmap-c)# set connection embryonic-conn-max 200 – встановлення
                                                  максимальної кількісті
                                                  напіввідкритих з'єднань.
ASA(config-pmap-c)# set connection per-client-embryonic-max 10 – встановлення
                                                      максимальної кількість
                                                      напіввідкритих з'єднань
                                                      для IP-адреси.
ASA(config-pmap-c)# set connection per-client-max 5 – встановлення максимальної
                                                кількості з'єднань для конкретної
                                                IP-адреси.
ASA(config-pmap-c)# set connection random-sequence-number enable
ASA(config-pmap-c)# set connection timeout embryonic 0:0:45 – встановлення таймеру
                                                    для напіввідкритих з'єднань
                                                    (після 45 секунд з'єднання
                                                    блокуватиметься).
ASA(config-pmap-c)# set connection timeout half-closed 0:25:0 – встановлення
                                                    таймеру для напівзакритих
                                                    (half closed) з'єднань.
ASA(config-pmap-c)# set connection timeout idle 2:0:0 – встановлення таймеру для
                                                    тривалості сесії.
```

Для підтвердження ефективності запропонованих заходів здійснимо повторну DoS-атаку і аналіз мережного трафіку аналізатором Wireshark на тих же інтерфейсах, що і до впровадження розроблених заходів інформаційної безпеки (Рис. 4.8).



No.	Time	Source	Destination	Protocol	Length	Info
59	152.595658	192.168.137.1	192.168.137.100	TCP	74	49858→80 [SYN] Seq=0
60	152.595658	192.168.137.1	192.168.137.100	TCP	74	49859→80 [SYN] Seq=0
61	152.595741	192.168.137.1	192.168.137.100	TCP	74	49860→80 [SYN] Seq=0
62	152.595741	192.168.137.1	192.168.137.100	TCP	74	49861→80 [SYN] Seq=0
63	152.595829	192.168.137.1	192.168.137.100	TCP	74	49862→80 [SYN] Seq=0
64	152.595829	192.168.137.1	192.168.137.100	TCP	74	49863→80 [SYN] Seq=0
65	152.595907	192.168.137.1	192.168.137.100	TCP	74	49864→80 [SYN] Seq=0
66	152.595907	192.168.137.1	192.168.137.100	TCP	74	49865→80 [SYN] Seq=0
67	152.595990	192.168.137.1	192.168.137.100	TCP	74	49866→80 [SYN] Seq=0
68	152.595990	192.168.137.1	192.168.137.100	TCP	74	49867→80 [SYN] Seq=0
69	152.595990	192.168.137.1	192.168.137.100	TCP	74	49868→80 [SYN] Seq=0
70	152.596091	192.168.137.1	192.168.137.100	TCP	74	49869→80 [SYN] Seq=0
71	152.596091	192.168.137.1	192.168.137.100	TCP	74	49870→80 [SYN] Seq=0
72	152.596127	192.168.137.1	192.168.137.100	TCP	74	49871→80 [SYN] Seq=0
73	152.596153	192.168.137.1	192.168.137.100	TCP	74	49872→80 [SYN] Seq=0
74	152.596153	192.168.137.1	192.168.137.100	TCP	74	49873→80 [SYN] Seq=0
75	152.596192	192.168.137.1	192.168.137.100	TCP	74	49874→80 [SYN] Seq=0
76	152.596192	192.168.137.1	192.168.137.100	TCP	74	49875→80 [SYN] Seq=0
77	152.596231	192.168.137.1	192.168.137.100	TCP	74	49876→80 [SYN] Seq=0
78	152.596255	192.168.137.1	192.168.137.100	TCP	74	49877→80 [SYN] Seq=0
79	152.596278	192.168.137.1	192.168.137.100	TCP	74	49878→80 [SYN] Seq=0
80	152.596278	192.168.137.1	192.168.137.100	TCP	74	49879→80 [SYN] Seq=0
81	152.596337	192.168.137.1	192.168.137.100	TCP	74	49880→80 [SYN] Seq=0
82	152.596364	192.168.137.1	192.168.137.100	TCP	74	49881→80 [SYN] Seq=0
83	152.596422	192.168.137.1	192.168.137.100	TCP	74	49882→80 [SYN] Seq=0
84	152.596422	192.168.137.1	192.168.137.100	TCP	74	49883→80 [SYN] Seq=0
85	152.596427	192.168.137.1	192.168.137.100	TCP	74	49884→80 [SYN] Seq=0
86	152.596427	192.168.137.1	192.168.137.100	TCP	74	49885→80 [SYN] Seq=0
87	152.596482	192.168.137.1	192.168.137.100	TCP	74	49886→80 [SYN] Seq=0
88	152.596482	192.168.137.1	192.168.137.100	TCP	74	49887→80 [SYN] Seq=0
89	152.596481	192.168.137.1	192.168.137.100	TCP	74	49888→80 [SYN] Seq=0
90	152.596508	192.168.137.1	192.168.137.100	TCP	74	49889→80 [SYN] Seq=0
91	152.596508	192.168.137.1	192.168.137.100	TCP	74	49890→80 [SYN] Seq=0
92	152.596511	192.168.137.1	192.168.137.100	TCP	74	49891→80 [SYN] Seq=0
93	152.596511	192.168.137.1	192.168.137.100	TCP	74	49892→80 [SYN] Seq=0

Рис. 4.8. Статистика надходження SYN-пакетів у процесі DoS-атаки на зовнішній інтерфейс після застосування заходів інформаційної бепеки

Рис. 4.8 свідчить, що на зовнішній інтерфейс мережного екрану у процесі атаки надходять SYN-пакети, джерелом яких є пристрій із адресою 192.168.137.1, як у випадку експерименту без застосування заходів інформаційної безпеки. Однак, аналізуючи стовпець Time (Рис. 4.9) в каналі зв'язку між мережним екраном і веб-сервером, бачимо, ознаки наявності пакетів, які є елементами атаки, відсутні, що і є свідченням ефективності розроблених заходів.

No.	Time	Source	Destination	Protocol	Length	Info
28	145.536870	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
29	150.536668	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
30	155.536595	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
31	160.536275	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
32	165.536215	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
33	169.593927	1.1.1.100	192.168.137.1	ICMP	114	Echo (ping) request id=0x0002, seq=0/0, ttl=255 (no
34	169.594430	vmware_c0:00:02	Broadcast	ARP	42	Who has 192.168.137.100? Tell 192.168.137.75
35	170.252507	vmware_c0:00:02	Broadcast	ARP	42	Who has 192.168.137.100? Tell 192.168.137.75
36	170.535877	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
37	171.098906	vmware_c0:00:02	Broadcast	ARP	42	Who has 192.168.137.100? Tell 192.168.137.75
38	171.753799	1.1.1.100	192.168.137.1	ICMP	114	Echo (ping) request id=0x0002, seq=1/256, ttl=255 (
39	173.690502	1.1.1.100	192.168.137.1	ICMP	114	Echo (ping) request id=0x0002, seq=2/512, ttl=255 (
40	173.692746	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.100? Tell 1.1.1.1
41	173.699501	ca:00:16:70:00:00	00:ab:cd:92:52:02	ARP	60	1.1.1.100 is at ca:00:16:70:00:00
42	173.699981	192.168.137.1	1.1.1.100	ICMP	114	Echo (ping) reply id=0x0002, seq=2/512, ttl=128 (
43	173.759899	1.1.1.100	192.168.137.1	ICMP	114	Echo (ping) request id=0x0002, seq=3/768, ttl=255 (
44	173.761654	192.168.137.1	1.1.1.100	ICMP	114	Echo (ping) reply id=0x0002, seq=3/768, ttl=128 (
45	173.773490	1.1.1.100	192.168.137.1	ICMP	114	Echo (ping) request id=0x0002, seq=4/1024, ttl=255
46	173.773856	192.168.137.1	1.1.1.100	ICMP	114	Echo (ping) reply id=0x0002, seq=4/1024, ttl=128
47	175.535823	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
48	180.535639	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
49	185.535459	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
50	190.535207	00:ab:cd:92:52:02	Broadcast	ARP	42	Who has 1.1.1.10? Tell 1.1.1.1
51	190.721570	vmware_c0:00:02	Broadcast	ARP	42	Who has 192.168.137.42? Tell 192.168.137.75
52	193.412015	ca:00:16:70:00:00	CDP/VTP/DTP/PAgP/UD	CDP	364	Device ID: R6 Port ID: FastEthernet0/0

Рис. 4.9. Статистика надходження SYN-пакетів у процесі DoS-атаки на внутрішній інтерфейс мережного екрану після застосування заходів інформаційної бепеки

Для зменшення ступеня успішності атаки типу «людина посередині» запропоновано такі заходи інформаційної безпеки:

– виділення окремої віртуальної локальної мережі для пристроїв звичайних користувачів, які виконують у роботу у соціальних мережах, або інформаційних ресурсах, аналогічним їм за принципом функціонування. Це дасть змогу обмежити широкомовний домен, а, отже, захистити пристрій користувача від фальшивих ARP-відповідей, які надсилає пристрій зловмисника, видаючи себе за мережний шлюз. Відповідна конфігурація мережного комутатора подана нижче.

```
Sw(config)# vlan 10 – створення VLAN 10, в якій знаходиться кінцева станція для
                    роботи в соціальній мережі.
Sw(config)# interface fa0/2 – інтерфейс, до якого підключена кінцева станція для
                           роботи в соціальній мережі.
Sw(config-if)# switchport mode access – зміна режиму роботи інтерфейсу на режим
                                    доступу.
Sw(config-if)# switchport access vlan 10 – переміщення утвореного інтерфейсу в VLAN
                                         10.
Sw(config)# interface fa0/1 – інтерфейс, що підключений до маршрутизатора Router.
Sw(config-if)# switchport mode trunk – необхідно перевести інтерфейс в транковий
                                   режим для можливості комунікації різних VLAN.
Sw(config-if)# switchport trunk allowed vlan 1,10 – інтерфейс пропускатиме пакети,
                                              що відносяться до VLAN 1 (по
                                              замовчуванню) та VLAN 10.
```

– маршрутизація трафіку з віртуальної локальної мережі VLAN 10 на інші інтерфейси маршрутизатора релізована завдяки створенню сабінтерфейсу, який виконує роль шлюзом для вузлів із VLAN 10.

```
Router(config)# interface fa2/0.10 – здійснюємо підключення до комутатора.
Router(config-subif)# encapsulation dot1q 10 - інкапсуляція пакету за протоколом
                                          802.11q для VLAN 10. Це дозволить
                                          призначати пакетам мітки приналежності
                                          до VLAN 10.
Router(config-subif)# ip address 192.168.10.1 255.255.255.240 – призначення IP-
                                                        адреси сабінтерфейсу, що є
                                                        шлюзом для кінцевих
                                                        пристроїв у VLAN 10.
```

– слід виконати зміну мережних налаштувань пристроїв користувачів, які належать до новоствореної віртуальної локальної мережі. Наприклад, замість адреси 10.10.10.10 вказуємо 192.168.10.2, замість маски підмережі 255.255.255.0 – 255.255.255.240, а шлюз 10.10.10.1 замінюємо на 192.168.10.1.

– ефективним засобом зниження впливу атаки «людина посередині» є статична прив'язка MAC-адреси шлюзу в ARP-таблиці пристрою звичайного користувача. Для цього в терміналі пристрою користувача слід виконати команду:

```
arp -s 10.10.10.1 cc-01-0c-14-00-20
```

де '10.10.10.1' – адреса мережного шлюзу, 'cc-01-0c-14-00-20' – MAC-адреса.

Варто відзначити, що у випадку зміни інтерфейсу шлюза треба щоразу змінювати значення MAC-адреси в ARP-таблиці користувача, навіть якщо IP-адреса шлюза залишилася незмінною.

– використання функції комутатора Dynamic ARP Inspection.

У ході дослідження застосовано дві форми атак: коли зловмисник розташовується за межами мережі, або ж всередині мережі. Перша форма атаки здійснена засобами DoS-атаки на веб-сервер в сегменті обслуговування досліджуваної ІТС для того, щоб нівелювати можливість сервера відповідати на запити користувачів, друга – на прикладі атаки «людина посередині» для заволодіння конфіденційними даними звичайних користувачів. DoS-атаку реалізовано методом SYN-flooding, а атака «людина посередині» – ARP-spoofing.

За результатами вивчення впливу згаданих атак на ІТС запропоновано та розроблено заходи із їх виявлення та нейтралізації. Успішне подолання DoS-атаки забезпечено розробленими налаштуваннями мережного екрану, що дало змогу уникнути потрапляння в захищений сегмент системи 100% пакетів, які виступали елементами SYN-flooding потоку. Для нейтралізації можливості організації атаки типу «людина посередині» запропоновано комплекс заходів, які унеможливлюють застоування методу ARP-spoofing. Запропоновані методи та засоби застосовано у досліджуваній розподіленій гетерогенній ІТС на рівні надання cloud послуг типу IaaS.

Шляхом імітаційного моделювання оцінимо застосування апаратного мережного екрану для захисту від атак типу DDoS, а також розрахуємо обсяг необхідних апаратних ресурсів для виконання імітаційного моделювання, яке відповідає реальним умовам атаки типу «відмова в обслуговуванні».

Розглянуто конкретний сценарій DDoS атаки на внутрішню мережу Естонії. Завдяки добре задокументованим фактам у роботі проведено спланову DDoS атаку із використанням описаних вище способів на ІТС із відтвореню топологією. Параметри і наслідки атаки точно відповідали реальним.

Відзначимо, що типова атака зазвичай містить три етапи:

- підготовчий,
- імплементаційний;
- період впливу наслідків.

Перші дві фази детакльно описані вище. Відзначимо, що етап підготовки визначає успішність майбутньої атаки та потребує залучення великого обсягу ресурсів. Ці ресурси затрачають на організацію мережі зловмисників (ботнет). Важливо, щоб ця мережа була повністю підконтрольною організатору атаки для її одночасного початку. Це забезпечує зловмиснику можливість створити шкідливий трафік такої інтенсивності, яка призведе до перевантаження цільвої ІТС. [22].

Моделювання атаки здійснено у середовищі NESSI. Вона є зручною для цілей дослідження з точки зору простоти використанням та наочності відображення процесів. Враховуючи, що необхідно створити мережу зловмисника, то можливість платформи автоматично генерувати підмережі із заданими параметрами (напр. за кількістю маршрутизаторів, середньою пропускною здатністю ребер тощо) добре підходить для цілей дослідження [22].

а) б)

Рис. 4.10. Конфігурації ІТС: без засобів інформаційної безпеки (а) та із застосуванням мережного екрану (б)

Моделювання проведено на середньостатистичній робочій станції, типовій для домашнього вжитку. Це дає змогу підтвердити, що зловмисна мережа може бути організована із великої кількості малопотужних компонентів.

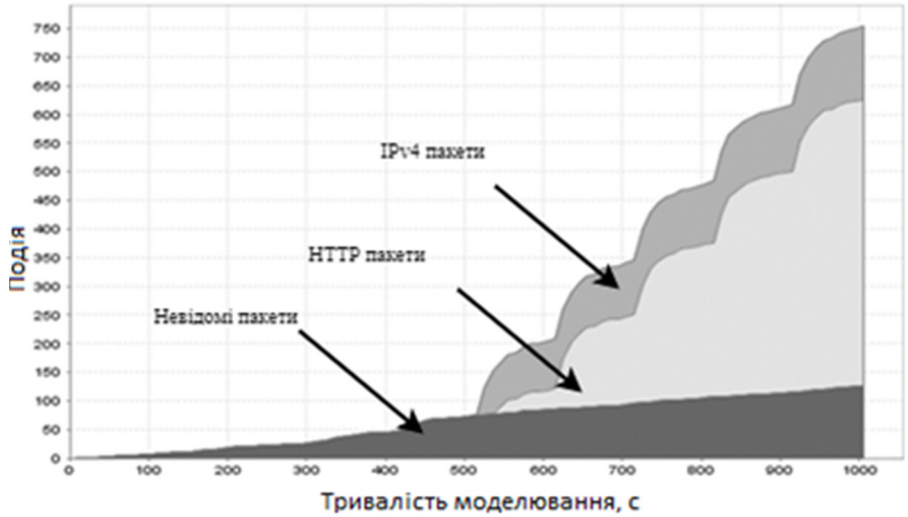


Рис. 4.11. Атака на веб-сервер без мережевого екрану

Рис. 4.12. Атака на веб-сервер з мережевим екраном

Експериментальним шляхом встановлено, що організація масштабної DDoS-атаки на незахищену мережу не потребує залучення надвеликих обчислювальних ресурсів. Тому розроблені у розділі заходи інформаційної безпеки дають змогу суттєво підвищити вартість організації атаки, а, отже, знизити імовірність її організації.

У дослідженні встановдено, що введення мережного екрану [25] із рекомендованими у роботі налаштуванянми у структуру ІТС дало могу знизити частку зловмисного трафіку у вутрішній мережі на 86% у реальних умовах [22].

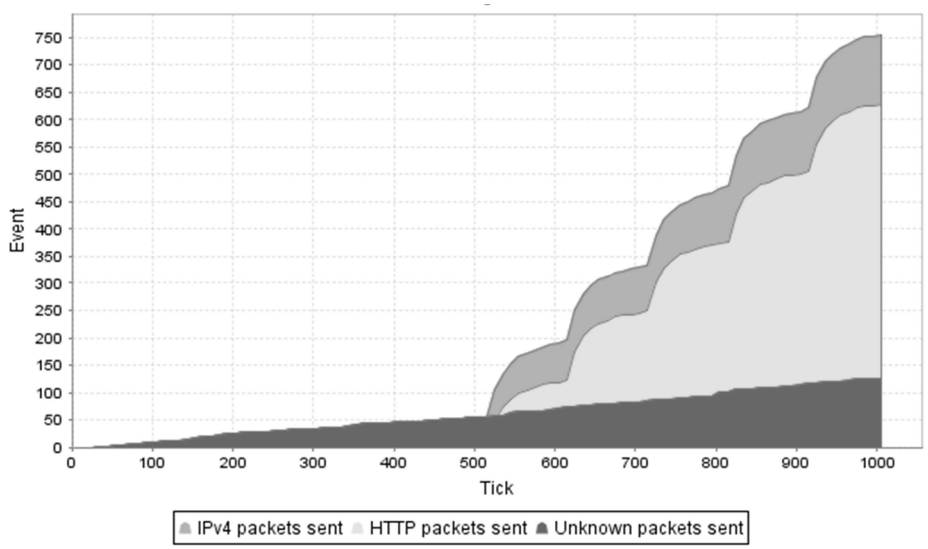


Рис. 4.13. Трафік на внутрішньому інтерфейсі мережного екрану

4.4. Метод забезпечення катастрофостійкості інформаційно-комунікаційних систем

4.4.1. Принципи забезпечення відмовостійкості розподілених систем

Відмовостійкість тісно пов'язана з поняттям надійності. У розподілених системах вона характеризується низкою понять [9, 59, 83, 202]:

- Доступність – система готова до негайного використання.
- Надійність – система може безперервно працювати без збоїв.
- Безпечність – якщо система не працює, це не призводить до виникнення катастрофічних наслідків.
- Підтримуваність – коли система не працює, її можна легко і швидко налагодити (часом непомітно для кінцевих користувачів).

Коли справа доходить до відмов, більшість з них може бути віднесена до однієї з двох категорій: апаратні або програмні відмови [110].

Апаратні відмови більш поширені, але з усіма останніми новинками в галузі проектування та виробництва апаратного забезпечення вони мають тенденцію до зменшення, а більшість з цих відмов пов’язані з мережею або диском.

З іншого боку, збій програмного забезпечення переходить у багато інших різновидів. І помилки програмного забезпечення в розподілених системах можуть бути важко відтворювані і, отже, їх складно виправити.

У малих, автономних системах набагато простіше моделювати умови, необхідні для реплікації та налагодження систем після відмов, при цьому більшість з цих питань класифікуються помилка, яка постійно проявляється під чітко визначенею (але можливо невідомою) сукупністю умов. Проте в більш складних системах або виробничих середовищах, що мають багато серверів, може бути надзвичайно важко знайти та діагностувати нерегулярні помилки, які зникають або змінюють свої характеристики під час процесу їх відтворення.

З більшою кількістю апаратного забезпечення зростає ймовірність виникнення відмови. Додавання додаткового програмного забезпечення та складних взаємодій між різними його компонентами підвищує імовірність виникнення відмов, включаючи нерегулярні. У результаті будь-яку розподілену гетерогенну систему складно діагностувати на відмови [63].

При проектуванні розподілених систем кажуть, що наступні припущення слід вважати помилковими [111]:

- Мережа є надійною.
- Затримка рівна нулю.
- Пропускна здатність необмежена.
- Мережа повністю захищена.
- Мережна топологія не піддається змінам.
- У мережі працює один адміністартор.
- Вартість передавання інформації рівна нулю.
- Мережа є гомогенною.

Досліджуючи кожне з цих припущень та розглядаючи побудову системи в цьому контексті, можна виявити потенційні ділянки виникнення відмов. Системи, які демонструють основні принципи, такі як надійність та доступність, мають в своїй основі проекти, які враховують всі вищеперераховані помилкові припущення. Відновлювальні роботи в цих сферах можуть виконуватися автоматично, в іншому разі при проектуванні мережі слід передбачити план відновлення роботи системи при виникненні катастроф різного характеру. Існує

багато різних прийомів, які допомагають захищатись від відмов: надлишковість, проектування з урахуванням вимог надійності та моніторинг.

Ще однією важливою частиною архітектури, що базується на сервісах, є те, що кожен сервіс повинен бути відмостійким, тобто таким, що здатен зберігати працездатність в тому випадку, якщо один з його компонентів недоступний або працює з помилками. Відмовостійкий сервіс здатен обробляти ці ситуа8ції, при цьому не має спостерігатися значне погіршення його працездатності для користувача. Існує ряд способів досягнення відмовостійкості в розподіленій системі: надлишковість, очікування, флаг-функції та асинхронність.

В розподілених системах існує набір типів помилок, які призводять до відмов:

- Однаразова відмова – з'являється один раз, потім зникає.
- Нерегулярна відмова – виникає, зникає, з'являється знову; але не відповідає жодній відомій моделі (найгірший випадок).
- Постійна відмова – як тільки це відбудеться, лише заміна / ремонт несправного компонента дозволить розподіленій системі працювати нормально.

Standby – це режим очікування, тобто резервний набір функціональних можливостей або дані, що знаходяться в режимі очікування, які можуть бути використані, щоб замінити інший несправний екземпляр. Реплікація може бути використана для підтримки копій основної бази даних в режимі реального часу, щоб дані могли бути замінені без втрат чи збоїв.

Флаги функцій використовуються для ввімкнення або вимкнення функціональності у системі. У випадку виходу з ладу певної системи, функції, які залежать від цієї системи, можуть бути вимкнені і недоступні, доки ця система не повернеться в глобальний доступ.

Асинхронність – одна з найважливіших проектних властивостей будь-якої розподіленої гетерогенної системи. Це, по суті, означає, що кожен сервіс або функціональна частина системи взаємодіє з будь-якою із зовнішніх сутностей асинхронно, отже, повільні або недоступні сервіси безпосередньо не впливають

на функціонування системи в цілому. Це також зазвичай означає, що операції не тісно пов'язані, що не вимагає успішності однієї операції для іншої. Наприклад, запис образу може бути успішним, навіть якщо всі копії файлу не були створені та записані в файлове сховище. Деякі фрагменти файлу записуються як асинхронні – будуть завантажені файли, перетворення файлу у потрібний формат / розмір, записування перетвореного файлу на диск, а потім реплікація його в резервну копію. Хоча всі елементи файлу необхідні для того, щоб запис файлу вважався успішним, кожний з них, зрештою, повинен мати місце, що означає, що клієнт може отримати "успіх" після завантаження файлу, а решта операцій може відбутися асинхронно після цього. Це означає, що якщо одна з поточних послуг недоступна або перевантажена, інші частини образу можуть продовжувати працювати, як очікувалося (у цьому випадку, приймаючи файли від клієнтів).

Моніторинг. Широкий моніторинг та ведення журналів є важливим для будь-якої складної розподіленої системи. Маючи багато різних постійно взаємодіючих послуг, є ризик виникнення надзвичайно нетипових помилок. Важко сказати, що викликало цю ситуацію, і як її вирішити. Одним із найкращих способів зниження цього ризику та швидкого діагностування проблем є гарантія, що всі інтерфейси системи та API інтерфейси контрольовані.

Однак моніторинг у масштабних розподілених веб-системах може бути складним завданням.

Як зазначалося раніше, ключовою частиною процесу масштабування є розбиття частин системи на сервіси, але оскільки кожен з них є незалежним один від одного, це може ускладнити діагностику у випадку виникнення проблем. Є багато різних контрольних точок, вони не обов'язково працюють синхронно, роблячи традиційний послідовний моніторинг або відстеження шаблону виконання набагато складнішим.

Більше того, для більшості цих систем комунікація між ними може бути ускладненп через такі функціональні механізми, таких як спроби повторення (що відбувається, коли один сервіс подає запит, наприклад, отримує зображення, а інший реагує на помилку, наприклад, надто зайнятий щоб обслуговувати запит,

а ініціатор сервісу спробує повторити запит), що лише ускладнює проблему відстеження послідовних подій.

Ще однією проблемою для цих великомасштабних систем є додавання моніторингу до системи. Занадто багато моніторингу може спричинити затримки, займати місце і потенційно перешкоджати нормальній роботі. Це є однією з причин того, чому у багатьох програмних систем є налагоджувальний режим і виробничий режим з меншою реєстрацією подій та моніторингом. Моніторинг може тривати на тому ж фізичному обладнанні, а у випадку, якщо це не так, існує додаткова комунікація, необхідна для функціонування систем моніторингу і для відстеження та реєстрації метрик. Цей додатковий крок є ще одним рівнем взаємодії, що додатково ускладнює розуміння поведінки цих систем.

Незважаючи на всі склднощі, ключові метрики KPI виступають основою розуміння та діагностики проблем. Наведемо декілька прикладів передових методів моніторингу розподіленої системи:

1. Наскрізний моніторинг. Це вимагає визначення наскрізного шляху від всередині ITC. Наприклад, для прикладу хостингу зображень це буде завантаження зображення та забезпечення того, щоб зображення було записано так, як очікувалося, і його можна вилучити із місця його зберігання клієнтом. Це повне використання для веб-системи. Часто існує більше одного з цих випадків, і кожен з них слід контролювати окремо.

2. Монітор кожного сервісу незалежно один від одного. Оскільки кожен сервіс має власний контроль, то моніторинг кожного з них є ключовим елементом для розпізнавання або діагностики проблем в межах сервісу. Моніторинг кожного сервісу може бути різним, але часто існує певне перекривання у метриках, визначення яких є ціллю моніторингу (використання процесора, оперативної пам’яті, диску, мережевих інтерфейсів). Наприклад, хостинг зображень може бути частиною певного моніторингу послуг: швидкість читання, кількість одночасних звернень до системи, стан чергу запису, кількість з'єднань.

2. Моніторинг "до ідеальних" показників – погляд на речі з точки зору кінцевих користувачів. Важливою частиною будь-якої великої веб-системи є розуміння сприйняття кінцевого користувача. Недостатньо зрозуміти внутрішні процеси, але важливо також відстежувати загальний досвід клієнта. Як правило, це робиться за допомогою зовнішнього сервісу, який в найпростішій формі перевірить доступність ІТС, щоб переконатися, що вона відповідає на запити, а у більш складних випадках такий зовнішній сервіс фактично перевіряє наскрізні випадки її використання. Отримані таким чином значення метрик можуть допомогти діагностувати проблеми, які виникають між клієнтом і ІТС, і можуть стати одним із найшвидших способів виявлення мережних проблем. Для глобальних користувацьких баз цей вид моніторингу може бути розподілений географічно таким чином, щоб він міг враховувати гетерогенність мережі.

Необхідно вибрати частоту моніторингу, щоб виявляти проблеми, перш ніж вони впливатимуть на клієнтів (це також означає, що інженери мають достатньо часу для вирішення проблеми, а також для того, щоб зрозуміти інтенсивність необхідних змін та наступні кроки для подалання проблеми). Іншою ключовою частиною будь-якого моніторингу є забезпечення достатньої частоти вибірки даних для виявлення проблем та підвищення рівня попереджень, перш ніж це стане проблемою. Наприклад, якщо одночасних забагато запитів на завантаження зображень, але пропускна здатність фіксується лише через кожні кілька хвилин, можливо, вона не може бути достатньо частою, щоб виявляти ці пікові періоди використання, які викликають проблеми.

Варто встановити прийнятний рівень на підставі попередніх результатів моніторингу. Для того, щоб дійсно отримати максимальну користь від моніторингу, треба розуміти, що для системи "нормально". Без базової або чіткої історії дуже важко визначити, чи є щось не так, і слід це досліджувати. Наприклад, проблеми, викликані надмірним навантаженням або трафіком у системі, було б важко діагностувати, якщо б не було легкого способу зрозуміти, який тип навантаження або пропускна спроможність була типовою. Тому не лише слід відстежувати метрики, але й розглядати їх тренди.

Моніторинг систем моніторингу. Окрім наявності систем моніторингу відстеження подій всередині системи, це також є ключовим елементом для забезпечення моніторингу та звітування. У більшості випадків достатньо мати зовнішнє підключення, щоб забезпечити реагування на програмне забезпечення моніторингу і більшість з відомих систем мають веб-інтерфейс.

Незважаючи на те, що всеохоплюючий моніторинг не завжди необхідний, він є важливим компонентом будь-якої великомасштабної ІТС, і чим складніша архітектура, тим важливіше мати моніторинг на всіх частинах. Коли мова йде про розробку системи, вибір того, який тип моніторингу використовувати, є дуже важливим рішенням, оскільки наявність великої кількості інформації, що виявляється, є ключовою для швидкого вирішення та діагностики проблем. Більшість типів показників, що підлягають моніторингу, підпадають під три категорії: мережа, система та застосунок. Загалом, специфічні для системи метрики є найважливішими, але вони також можуть бути найважчими для вимірювання, оскільки вони вимагатимуть певної унікальної конфігурації.

На більшості розподілених ІТС існує окрема система (яка може бути реалізована однією або кількома машинами), на яку покладаються завдання зі збирання та агрегування даних. Це можна зробити, встановлюючи агенти або інструменти для кожної з машин, які підлягають моніторингу. Це допомагає відстежувати і один з особливих аспектів розподілених систем (особливо тих, що знаходяться у cloud), де машини часто переміщуються (вмикаються чи вимикаються або стають недоступними), забезпечувати налаштування процесу завантаження віртуальних машин, який включає автоматичне додавання моніторингу новим віртуальним машинам.

Ще однією проблемою є вибір того, які дані збирати, і з якою дискретністю має сенс для конкретної архітектури, тому що дуже легко можна збирати забагато даних, що може призвести до перевантаження сервера). Деякі великі ІТС, такі як Wikipedia та Twitter, використовують таке програмне забезпечення, як Ganglia, яке автоматично збирає дані, зберігаючи їх у розрідженому вигляді

для давніх подій; для останніх подій буде більше даних з більшою деталізацією, ніж для подій, які трапилися у минулому.

Існують ефективні системи моніторингу подій у ІТС (з відкритим кодом, що дає змогу на етапі експлуатації покращувати та за необхідності адаптувати їх функціонування): Nagios, Zabbix, Flume і Munin.

Розподілені ІТС – це системи, які компоненти яких не мають спільної пам'яті або синхронізації. В розподілених системах вузли комунікують, обмінюючись інформацією через телекомунікаційне середовище. Окремий компонент у розподіленій системі на фізичному рівні має власну пам'ять та операційну систему, локальні ресурси належать вузлу, що використовує ці ресурси. Ресурси, доступ до яких здійснюється через мережу або, відомі як віддалені ресурси. На Рис. 4.14 показана мережа зв'язку між системами в розподіленому середовищі. У розподіленій системі виконується набір правил для синхронізації дій різних або відокремлених процесів в мережі зв'язку, що формує певний набір пов'язаних завдань. Незалежні підсистеми з дистанційним або локальним доступом до ресурсів засобами розподілених ІТС об'єднуються, щоб сформувати єдину зрозумілу кінцевому користувачеві систему. Користувач у розподіленому середовищі не знає про існування декількох взаємопов'язаних систем, що забезпечує точне виконання завдання.

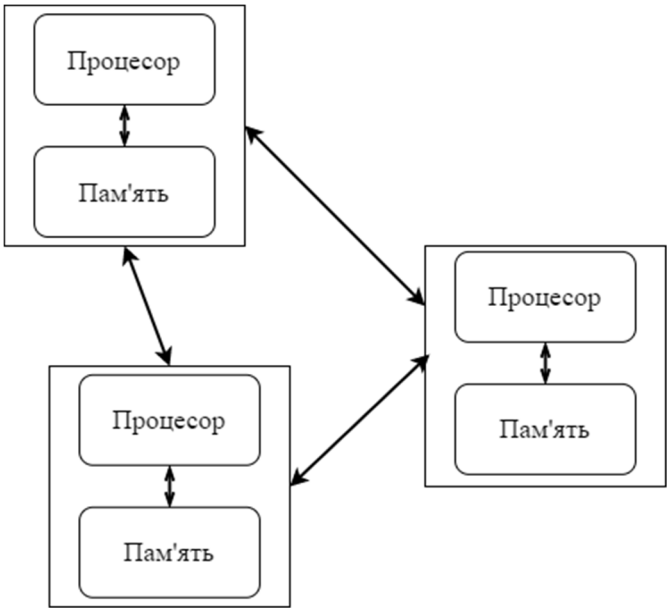


Рис. 4.14. Приклад розподіленої системи

4.4.2. Надлишковість інформаційної системи на рівні процесів

Цей метод забезпечення відмовостійкості часто використовується для відмов, які зникають, коли нічого не було зроблено для виправлення ситуації, а така відмова називається виипадковою відмовою. Випадкові відмови виникають, коли у будь-якому компоненті системи виникає випадкова відмова, або, іноді, через перешкоди навколишнього середовища. Проблема з випадковими відмовами полягає в тому, що їх важко діагностувати, але вони менш критичні. При обробці випадкових відмов застосовується технологія відмовостійкості програмного забезпечення, яку називають резервуванням на рівні процесу (PLR), тому що обладнанняз фізичним резервуванням є більш дорогим для розгортання. Як показано на Рис. 4.15, PLR порівнює процеси для забезпечення правильного виконання, а також створює набір надлишкових процесів для кожного процесу ITC. Резервування на рівні процесу дозволяє ОС планувати процеси на всіх доступних апаратних ресурсах [82].

Перевірка станів і відновлення – це популярна техніка, яка в на першому етапі зберігає поточний стан системи, і це робиться періодично. Інформація про контрольну точку зберігається в стабільному пристрої зберігання даних для легкого відновлення, коли виникає помилка у активному вузлі. Інформація, яка зберігається або перевіряється, включає в себе середовище, стан процесів, значення регістрів тощо. Ця інформація є дуже корисною, якщо потрібно повне відновлення. На Рис. 4.16 показані методи перевірки. Два найбільш відомих – це метод контрольної точки та метод відновлення з журналу подій.

Кожен тип технології відновлення використовує різні механізми. У методі контрольних точок (Рис. 4.17) використовуються контрольні точки, які зберігаються на стабільному пристрої зберігання даних, тоді як методи відновлення на основі журналу базуються на контролі та реєстрації подій [243].

У способі відновлення форм системної аварії існує два типи методу контрольної точки: координовані та некоординовані технології контрольних точок. Ці методи пов'язані з веденням протоколу повідомлень.

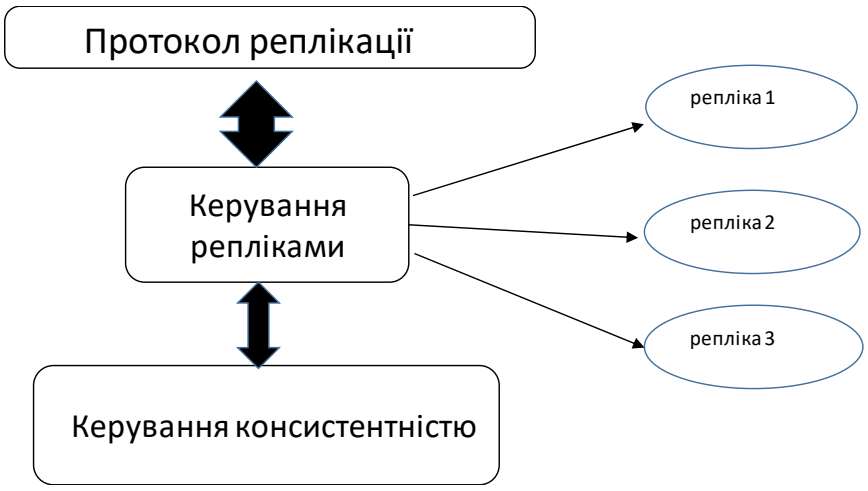


Рис. 4.15. Метод реплікації у розподілених системах

Координована контрольна точка: у цій техніці здійснюється збереження стабільного стану, оскільки координована контрольна точка є послідовним набором контрольних точок. Якщо контрольні точки не є послідовними, повне відновлення системи не може бути виконане.

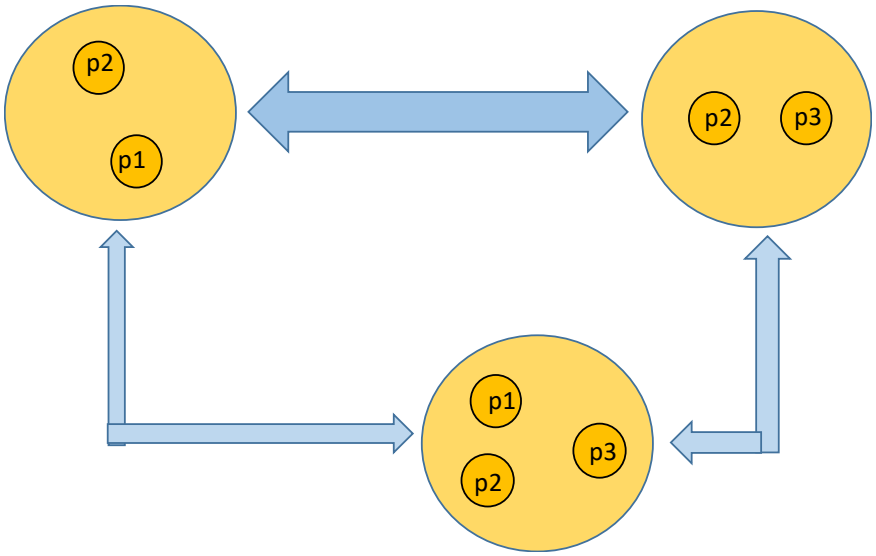


Рис. 4.16. Надлишковість на рівні процесів

У ситуації, коли часто виникає відмова, не можна використовувати координований метод контрольної точки. Час відновлення може бути встановлено на більш високе значення або нижчий обсяг відновлення. Коли відновлення відбувається шляхом опрацювання меншої кількості інформації, це покращує продуктивніст, оскільки відновлення відбувається лише з останнього правильного стану системи замість першого стану.

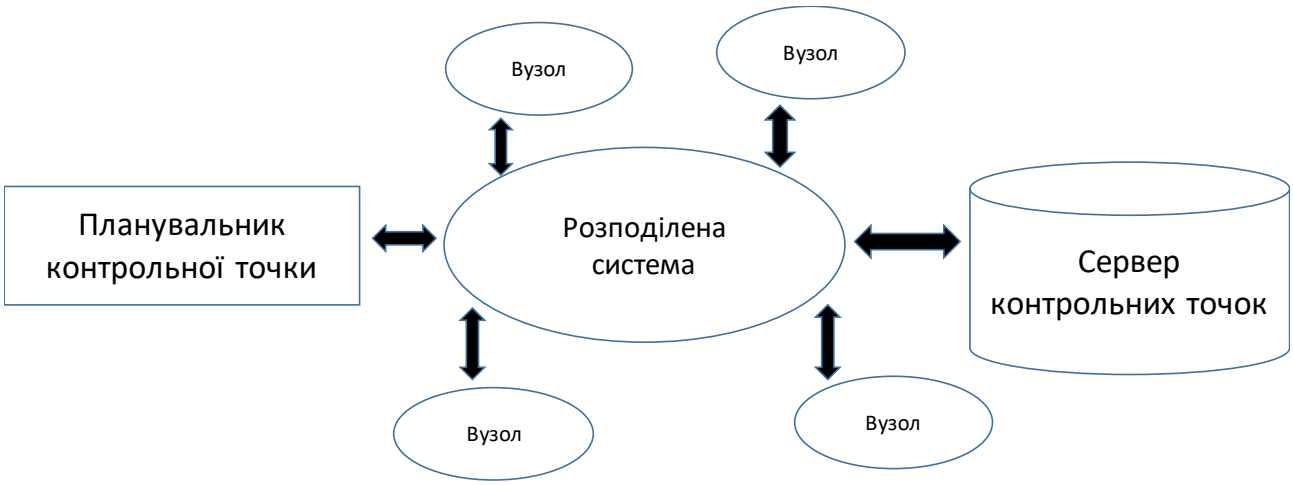


Рис. 4.17. Метод контрольних точок

Некоордована контрольна точка: ця методика поєднує в собі протоколювання повідомлень, щоб забезпечити правильність відновлення. Техніка некоординованої контрольної точки використовує незалежні контрольні точки для здійснення відновлення. Існує три типи протоколів реєстрації повідомлень: оптимістичні, песимістичні та випадкові. У оптимістичному протоколі забезпечується вхід усіх повідомлень. Песимістичний протокол гарантує, що всі повідомлення, отримані процесом, реєструються належним чином та зберігаються на стабільному та надійному носію даних, перш ніж вони передається в систему.

Технологія на основі інтеграції. Реплікація є найбільш широко використовуваним методом або технікою у відмовостійкості. Основним недоліком є множина резервних копій, які вона створює. Оскільки резервні копії зростають у міру збільшення відмов, а витрати на керування є дуже великими, техніка Fusion вирішує цю проблему (Рис. 4.18). Технологія на базі Fusion є альтернативою, оскільки вона вимагає менше машин для резервного копіювання порівняно з технікою, що базується на реплікації. Як показано на Рис. 4.18, резервні машини зливаються відповідно до заданого набору машин. Технологія на основі Fusion має дуже високі накладні витрати в процесі відновлення, і це прийнятно при низькій ймовірності виникнення відмови в системі.

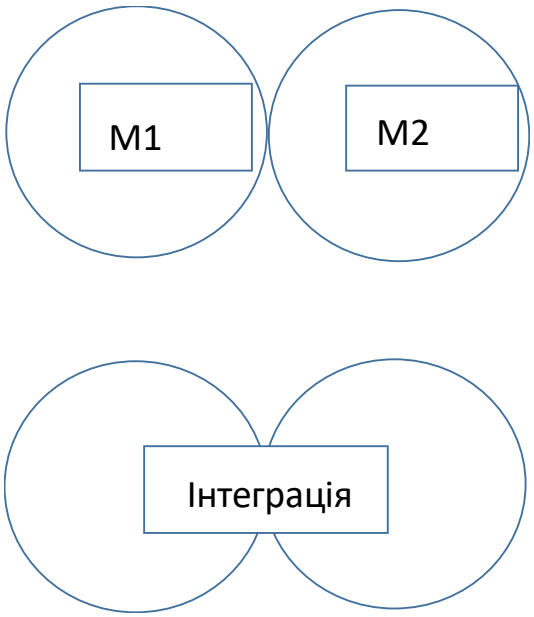


Рис. 4.18. Метод на основі інтеграції

Таблиця 4.1 показує, що всі проаналіовані методи здатні забезпечувати відновленння після множинних відмов. У всіх методах продуктивність залежить від складності процесу та кількості його учасників. З огляду на представлені результати, існує значна потреба в надійному, точному та даптивному механізмі виявлення множини відмов.

Таблиця 4.1. Порівняння методів забезпечення відмовостійкості розподілених систем

Характеристика	Метод на основі реплікації	Метод контрольних точок / відновлення	Метод злиття	Метод надлишковості на рівні процесів
Принцип дії	Переспрямування даних на репліки	Збереження стабільного стану системи і використання його для відновлення	Наявність резервної копії системи	Множина надлишкових процесів
Консистентність	За певним критерієм	Уникнення повідомлень без відповіді	Між машинами, на яких знаходяться резервні копії	Не критичний фактор
Здатність виправляти множинні відмови	Залежить від кількості реплік	Залежить від розкладу збереження контрольних точок	Залежить від кількості резервних машин	Залежить від розмірності множини надлишкових процесів

Характеристика	Метод на основі реплікації	Метод контрольних точок / відновлення	Метод злиття	Метод надлишковості на рівні процесів
Продуктивність	Зменшується зі збільшенням кількості реплік	Зменшується зі збільшенням розміру та частоти збору контрольних точок	Зменшується у випадку збоїв, оскільки вартість відновлення висока	Зменшується зі зменшенням кількості збоїв
Розмірність	N реплік забезпечують контроль N-1 збоїв	N рівневі диски забезпечують контроль N-1 збоїв	Для контролю N збоїв потрібно N резервних машин	Масштабування кількості процесів на основі принципу мажоритарної системи
Виявлення множинних відмов	Надійне, точне, адаптивне	Надійне, точне, адаптивне	Надійне, точне, адаптивне	Надійне, точне, адаптивне

Відмовостійкість – це одна з основних характеристик розподіленої системи, оскільки вона забезпечує безперервність і функціональність системи в точці, де виникають відмови. Проведене дослідження показало властивості та характеристики технологій забезпечення відмовостійкості в розподілених ІТС,. Кожна з технологій та методів, які лежать в їх основі, має переваги та недоліки, які проявляються залежно від архітектури ІТС.

4.4.3. Маскування відмов в розподілених інформаційно-комунікаційних системах

Резервування – це надання функціональних можливостей, які були б непотрібними у бездоганному середовищі. ІТС може складатися з резервних компонентів, які автоматично починають функціонувати, якщо один з компонентів системи втратив працездатність. Ідея застосування надлишковості для підвищення надійності системи відома ще з 50-х років 20 століття.

У ІТС можливі два види резервування: надлишковість у просторі та надлишковість у часі. Просторове резервування забезпечує додаткові компоненти, функції або елементи даних, які не потрібні для безвідмовної роботи. Пропускна здатність відображається в апаратну, програмну та

інформаційну надлишковість залежно від типу резервних ресурсів, доданих до системи.

Введення надлишковості в ІТС передбачає, що опрацювання або передавання даних повторюється, і результат порівнюється зі збереженою копією попереднього результату. Цей процес може відбуватись і паралелізовано.

Стратегія функціонування ІТС у цьому випадку зводиться до приховування випадків відмови від інших процесів із використанням надлишковості. Три основних типи надлишковості, що використовуються в ІТС:

- Інформаційне резервування – додавання надлишкових бітів, щоб забезпечити виявлення помилок / відновлення (наприклад, коди Хеммінга тощо).
- Часове резервування – операція виконується та за потреби виконується знову.
- Фізичне резервування – додавання додаткового обладнання та / або програмного забезпечення до ІТС.

4.4.4. Методи відновлення працездатності інформаційно-комунікаційних систем після катастрофічних явищ

Якщо продуктивніть ІТС зменшується, то це зменшення пропорційно ступеню відмови. У нерезервованих системах навіть незначна відмова може призвести до повної втрати працездатності. Безвідмовна робота особливо бажана у високодоступних системах або в критичних системах. Здатність підтримувати функціональність, коли частини системи відомовляють, називають живучістю.

Надійнісне проектування гарантує системі можливість збереження працездатності із зниженою продуктивністю, коли якась частина системи не працює. Цей термін найчастіше використовується для опису ІТС, призначених для продовження більш-менш повноцінної роботи, можливо, зменшення пропускної здатності або збільшення часу реагування у разі часткової відмови їх компонентів. Тобто система в цілому не зупиняється через проблеми як в апаратному, так і в програмному забезпеченні.

В рамках індивідуальної системи відмовостійкість може бути досягнута завдяки передбаченню виняткових умов і побудові системи для їх подолання та, загалом, для самостійної стабілізації, щоб система сходилася до безпомилкового стану. Однак, якщо наслідки відмови системи є катастрофічними, або витрати на її достатньо надійне функціонування дуже високі, найкращим рішенням може бути використання деякої форми дублювання. У будь-якому випадку, якщо наслідки відмови ІТС є катастрофічні, система повинна мати можливість використовувати реверсію, щоб повернутися до працездатного режиму. Це схоже на відновлення, але може бути діяльністю людини, якщо люди присутні в циклі функціонування системи [65, 79].

У ІТС стійкість – це здатність забезпечувати і підтримувати прийнятний рівень обслуговування, всупереч наявним причинам порушення нормального функціонування. Ці причина для сервісів можуть варіюватися від простої неправильної настройки до великомасштабних стихійних лих і цілеспрямованих атак. Таким чином, еластичність ІТС торкається дуже широкого кола завдань. Щоб підвищити стійкість певної ІТС, необхідно визначити можливі виклики та ризики та визначити відповідні значення метрик стійкості для сервісу, який буде захищено.

Ці послуги включають:

- підтримку розподіленої обробки;
- підтримку мережного зберігання;
- підтримка обслуговування сервісів, таких як
  - відеоконференція;
  - миттєві повідомлення;
- онлайнове співробітництво;
- доступ до застосунків та даних у міру необхідності.

Надійні клієнт-серверні комунікації. У багатьох випадках відмовостійкість в розподіленій ІТС концентрується на несправному процесі. Однак ми також повинні розглянути проблеми з повідомленнями про помилки. Зокрема, канал зв'язку може виявляти відмову, мемент відмови та випадкові помилки.

Комунікація пункт-пункт. У багатьох розподілених системах надійний зв'язок пункт-пункт встановлюється за допомогою надійного транспортного протоколу, такого як TCP. TCP контролює втрачені повідомлення. TCP маскує викикнення відмов.

Семантика RPC при наявності помилок визначає те, як система має реагувати на виникнення відмов у будь-якому з її елементів у довільний момент функціонування системи.

Нижче подано варіанти відмов в розподіленій ІТС на основі архітектурного шаблону клієнт-сервер:

- Клієнт не може знайти сервер.
- Запит від клієнта на сервер втрачено.
- Сервер відмовляє після отримання запиту.
- Відповідне повідомлення від сервера до клієнта втрачається.
- Клієнт зникає після надсилання запиту.
- Комунікація пункт-пункт.

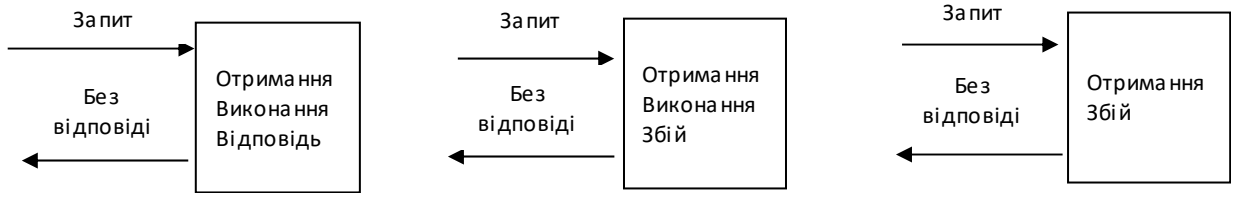


Рис. 4.19. Процеси виникнення відмов у процесі клієнт-серверної комунікації

Дуже важливою є гнучкість процесу шляхом реплікації. Не дивно, що важливі також надійні сервіси багатоадресної розсилки. Такі послуги гарантують доставку повідомлень усім членам групи процесів.

Надійна багатоадресна розсилка при втраті повідомлень гарантує, що вони будуть повторно передані.

Можливості:

- Схеми на основі підтвердження.
- Відправник може стати вузьким місцем.
- Схеми на основі відмови.

Процес, як і канал зв'язку, можуть вийти з ладу в розподіленій системі. Мета RPC полягає в тому, щоб приховати комунікацію. Виклики віддаленої процедури виглядають так само, як локальні. Дійсно, поки клієнт і сервер працюють відмінно, RPC виконує свою роботу надзвичайно добре. Проблема виникає при виникненні помилок. Саме тоді відмінності між локальними та віддаленими викликами не завжди легко замаскувати. У цьому розділі ми розглянемо деякі можливі відмови і запропонуємо способи їх маскування.

Почнемо з того, що клієнт не може знайти потрібний сервер. Наприклад, сервер може бути вимкнений. Крім того, припустимо, що клієнт певну неоновлену версію клієнтського компоненту системи. Тим часом, сервер розвивається, встановлюється нова версія інтерфейсу та створюються нові компоненти. Коли клієнт, нарешті, буде запущений, конектор не зможе зіставити його з сервером і повідомить про відмову.

У локальній системі бази даних для здійснення транзакції, менеджер транзакцій повинен лише передати рішення про відправлення для менеджера відновлення. Проте в розподіленій системі менеджер транзакцій повинен передати рішення всім серверам на різних фізичних розташуваннях, де здійснюється транзакція, і забезпечити одностайне виконання рішення. Коли обробка завершується на кожному фізичному сервері, транзакція переходить у стан частково завершеної та очікує на те, що всі інші транзакції досягають своїх станів часткової завершеності. Коли менеджер транзакцій отримує повідомлення про те, що всі фізичні сервери перебувають в стані часткової завершеності транзакції, він починає процес фіксування транзакції. У розподіленій ІТС транзакції мають виконатись на всіх фізичних серверах, або ж не виконатись на жодному. Це характеризує два можливі стабільні стани ІТС.

Процес фіксування транзакції може бути здійснено із використанням одного з таких протоколів:

- Однофазне фіксування
- Двофазне фіксування
- Трифазне фіксування

Розподілене однофазне фіксування – це найпростіший протокол фіксування. Уявимо, що існує контрольний сервер та декілька підлеглих серверів, де здійснюється транзакція. Кроки однофазного фіксування такі:

- Після того, як кожен з серверів локально завершив свою транзакцію, він надсилає повідомлення "Виконано" на контрольний сервер.
- Виконавці чекають повідомлення "Прийняти" або "Припинити" з контрольного сайту. Цей час очікування називають вікном вразливості.
- Коли контрольний сервер отримує повідомлення " Виконано " від кожного виконавця, він приймає рішення про фіксування чи переривання. Потім він надсилає це повідомлення всім підлеглим серверам.
- Після отримання цього повідомлення підлеглий сервер або фіксує транзакцію, або припиняє її, а потім надсилає повідомлення підтвердження на контрольний сервер.

Розподілене двофазне фіксування знижує вразливість однофазних протоколів фіксування. Дії, виконані у двох фазах, є такими:

Фаза 1: Підготовча фаза.

- Після того, як кожен з виконавців локально завершив свою транзакцію, він надсилає повідомлення "Виконано" на контрольний сервер. Коли контрольний сервер отримав повідомлення "Виконано" від усіх виконавців, він надсилає повідомлення «Підготовка» до виконавців.
- Виконавці голосують про те, чи вони все одно хочуть здійснити транзакцію. Якщо виконавець хоче здійснити її, він надсилає повідомлення "Готово".
- Координатор, який не бажає обміну повідомлення, надсилає повідомлення "Не готовий". Це може статися, коли виконавець має суперечливі одночасні транзакції або таймаут.

Фаза 2: Фаза фіксування / скасування

- Після того, як координатор отримав повідомлення «Готовий» від усіх виконавців, від надсилає їм повідомлення "Глобальне фіксування".

– Виконавці фіксують транзакцію та надсилають повідомлення "Підтвердження фіксування" на координатор.
– Коли координатор отримує повідомлення "Підтвердження фіксування" від усіх виконавців, він вважає транзакцію здійсненою.
– Після того, як координатор отримав перше повідомлення "Не готовий" від будь-якого виконавця, він надсилає повідомлення "Глобальне скасування" до всіх виконавців.
– Виконавці скасовують транзакцію та надсилають повідомлення "Підтвердження скасування" на координатор.
– Коли координатор отримує повідомлення "Підтвердження скасування" від усіх виконавців, він вважає транзакцію перерваною.

Кроки розподіленого трифазного фіксування полягають у наступному

Фаза 1: Підготовча фаза

Ці кроки такі ж, як у розподіленого двофазного фіксування.

Фаза 2: Підготовка до фази фіксування

Координатор видає повідомлення про передавання "Вхід в підготовлений стан". Виконавці голосують "Згода" у відповідь.

Фаза 3: Фаза Прийняття / скасування

Ці кроки такі ж, як і для двофазного доручення, крім того, що повідомлення "Підтвердження фіксування" / "Підтвердження скасування" не є обов'язковими.

Недолік протоколу двофазного фіксування є те, що протокол передбачає наявність стабільного сховища на кожному вузлі з журналом попереднього запису, щоб ні один вузол не відмовив назавжди, що дані в журналі запису ніколи не втрачаються та що будь-які два вузли можуть комунікувати один з одним. Останнє припущення не є надто обмеженим, оскільки мережеве з'єднання зазвичай можна переорієнтувати. Перші два припущення набагато оптимістичніші: якщо вузол повністю знищений, дані можуть бути втрачені.

Найбільшим недоліком двофазного протоколу є те, що це протокол з блокуванням. Якщо координатор повністю втрачає працездатність, деякі виконавці ніколи не завершать свої транзакції: після того, як група надіслала

координатору повідомлення про згоду, вона буде блокуватись, доки не буде отримано підтвердження чи скасування транзакції.

Основною відповідальністю адміністратора бази даних є підготовка до можливості відмови апаратного, програмного забезпечення, мережі, процесу або системної помилки. Якщо така відмова впливає на роботу системи бази даних, як правило, слід відновити базу даних і повернутися до нормальної роботи якомога швидше. Відновлення повинне захищати базу даних та пов'язаних користувачів від непотрібних проблем і уникати або зменшувати можливість дублювання ручної роботи.

Процес відновлення залежить від типу помилки, що виникла, структур, що зазнали впливу, та методу відновлення, який застосовується. Якщо жодні файли не втрачені або не пошкоджені, відновлення може становити лише перезапуск компонента. Якщо дані було втрачено, для відновлення потрібні додаткові дії.

Коли система використовує такі локальні мережі або телефонні лінії для підключення робочих станцій клієнтів до серверів баз даних або для підключення декількох серверів баз даних для формування розподіленої бази даних, збої в мережі, такі як припинені телефонні з'єднання або відмови програмного забезпечення мережі, можуть призвести до переривання нормального функціонування системи бази даних.

- Помилка мережі може перешкодити нормальному виконанню клієнтського застосунка та призвести до неефективності процесу. У цьому випадку фоновий системний процес виявляє та завершує процесу користувача.
- Помилка мережі може призвести до переривання двофазного фіксування розподіленої транзакції. Після того, як мережна проблема виправлена, фоновий процес СУБД автоматично виправляє будь-які розподілені транзакції, які ще не завершені на всіх вузлах розподіленої бази даних системи.

Резервування бази даних складається з резервних копій фізичних файлів (всіх файлів даних і контрольного файлу), які складають базу даних Oracle. Щоб

розпочати відновлення мультимедіа після відмови в середовищі, Oracle використовує резервні копії файлів для відновлення пошкоджених файлів даних або керованих файлів. Заміна поточного, можливо пошкодженого, копію файлу даних, табличного простору або бази даних за допомогою резервної копії називається відновленням цієї частини бази даних.

Пропонується декілька варіантів виконання резервних копій баз даних, зокрема:

- Менеджер відновлення
- Утиліти операційної системи
- Утиліти експорту

При виникненні помилки виникають дві потенційні проблеми:

- Блоки даних, модифіковані транзакцією, можуть бути не записані до файлів даних у час виконання, і можуть відображатися тільки в журналі повторних дій. Таким чином, журнал повторних дій містить зміни, які необхідно повторно застосувати до бази даних під час відновлення.
- Після фази накатки файли даних можуть містити зміни, які не були здійснені під час відмови. Ці нездійснені зміни повинні бути відновлені, щоб забезпечити узгодженість транзакцій. Ці зміни були або збережені до файлів даних перед відмовою, або введені під час фази накатки.

Зазвичай використовується два окремих кроки для успішного відновлення системи після відмов: відновлення кеша і відновлення транзакції.

У разі виходу з ладу операції можуть бути переведені на точну копію бази даних. У звичайних операціях зміни одночасно оновлюються в обидві бази даних, тому друга база даних – точна копія першої, її репліка.

Переваги

- Швидкість – система працює з мінімальними чи відсутніми порушеннями бізнес-процесів.
- Вартість – оскільки ціни на вторинне зберігання знизились.
- Пошкоджені дані можуть бути перебудовані з другої копії бази даних

Відновлення кеша. Ця методика передбачає повторне виконання денних операцій до моменту відмови. Монтується остання резервна копія бази даних. З журналами систем проводяться консультації, після чого всі транзакції, що відбулися після цієї копії, повторюються.

Переваги:

– Простота
– Не вимагається спеціальних процедур перезапуску.

Недоліки:

– Це вимагає багато часу для повторного фіксування транзакцій. Зайнята база даних може бути недоступною для очночасного обслуговування нових транзакцій та повторного виконання втрачених одночасно.
– Послідовність транзакцій може відрізнятись на репліці. Це може спричинити проблеми з транзакціями, які є критичними.

Відновлення транзакції. Цей метод передбачає, що СУБД скасовує будь-які зміни, що відбулися після останньої контрольної точки. Незмінні образи бази застосовуються до бази даних.

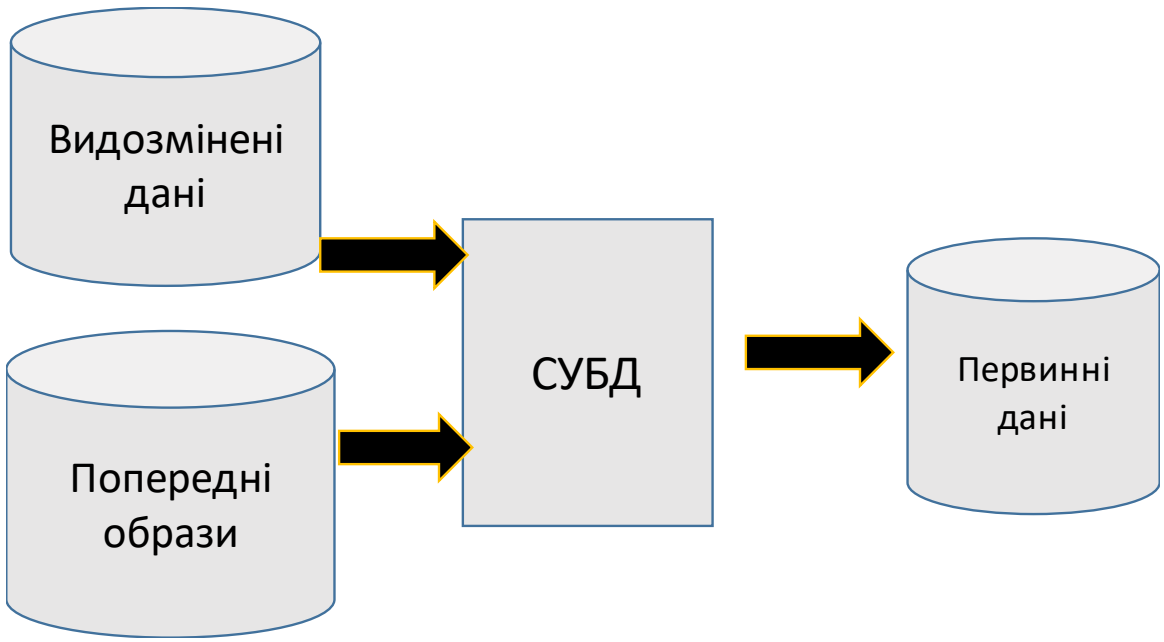


Рис. 4.20. Метод відновлення кешу

Цей метод використовується для змін у записах, де транзакція перервана або припинена аномально. База даних починається з попередньої копії бази даних

(можливо, з останньої контрольної точки). Після цього застосовуються образи для переміщення бази даних до точки відмови.

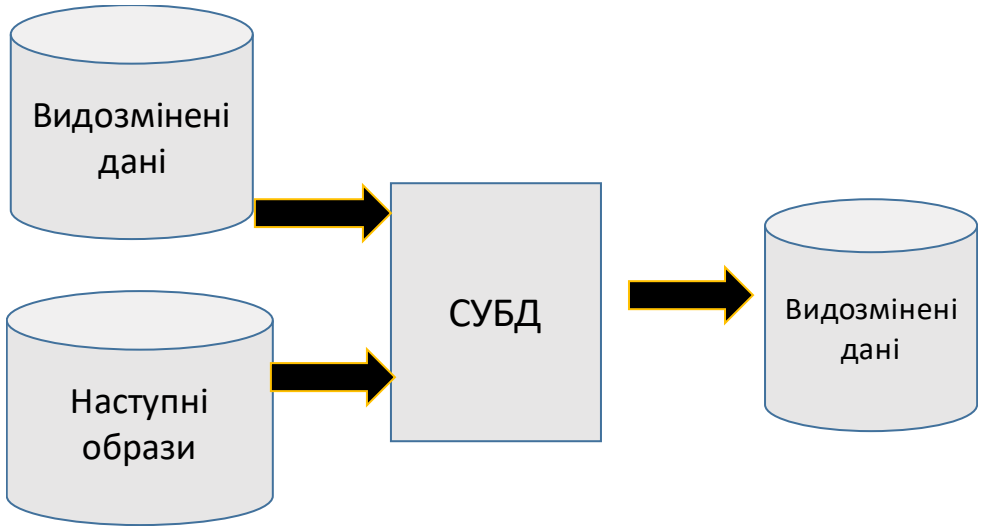


Рис. 4.21. Метод відновлення транзакції

Переваги:

– Швидше, ніж відновлення кешу
– Застосовуються контрольні точки даних. Це копії даних на конкретний момент, а не повторення процесів. Тому відновлення послідовності операцій вже не є проблемою.

4.4.5. Метод відновлення працездатності інформаційно-комунікаційних систем після катастроф, спричинених людським фактором, на основі програмного представлення системної інфраструктури

Cloud системи стали тенденцією останніх п'яти років у сфері ІКТ через їх керованість, еластичність, надійність, безпеку та сервісну орієнтацію. Дуже важливо, що хмарні системи тепер підтримують різні рівні SLA, що є потужним інструментом для створення та розвитку різних інформаційних систем та послуг. Перехід до хмар означає, що частина обов'язків переміщується до cloud провайдера. Клієнти отримують ресурси як сервіс. Ця концепція спричина повсюдність міграції у хмару. Це також означає, що частина всієї системи не керується споживачем ресурсу. У реальному світі це зазвичай допомагає

компаніям підвищити ефективність використання їхніх людських та фінансових ресурсів. Відповідальність за апаратне забезпечення переходить до постачальника cloud ресурсів.

Однак в cloud ресурсах існує певний ризик використання. Головне, що лише апаратні ресурси контролюються cloud постачальниками. У той же час всі елементи створеної інфраструктури контролюються споживачами хмарних ресурсів. Це означає, що інфраструктура організована на хмарних ресурсах (SaaS, PaaS або IaaS), але сама інфраструктура створюється та підтримується інженерами-клієнтами [139].

Існує кілька підходів до управління віддалено розташованою інфраструктурою [18]. Найпростіший і найстаріший полягає в ручному керуванні. Відповідно до цього підходу всі компоненти інфраструктури створюються та підтримуються за допомогою інструментів, наданих веб-інтерфейсом або інтерфейсом командного рядка cloud постачальника (наприклад, Azure UI, Azure CLI та ін.). Це класичний підхід, який був популярним протягом першого десятиліття 21 століття [45, 46]. Головною перевагою цього підходу є низький рівень знань та навичок, необхідних для роботи інженера-експлуатанта. Єдина вимога полягає в тому, щоб зрозуміти і керувати елементами інтерфейсу cloud постачальника. Крім того, є дуже ризикований недолік. Він може бути сформульований з точки зору теорії катастроф.

Коли виникає катастрофічне явище, система переходить у стан, з якого сама не здатна повернутися до робочого стану. Наприклад, вся інфраструктура може бути випадково видалена, або критичні компоненти можуть бути певним чином порушені. Як правило, подібні катастрофи не можуть бути вирішені на стороні постачальника, тому що інфраструктура не знаходиться під його контролем. У випадку відновлення баз даних більшість cloud постачальників використовують диференціальні резервні копії для відкатки робочих баз даних клієнта. Проте відновлення більшості компонентів ІТС не підтримуються cloud

постачальником. Тим більше, що це може бути чутливим у випадку використання програмного забезпечення як компонентів Cloud-сервісу.

У випадку ручного технічного обслуговування інфраструктури знадобиться багато часу, щоб відновити систему у робочий стан. Цей час збільшується із збільшенням кількості відновлюваних компонентів хмар і кількості залежностей між такими компонентами. Аварійне відновлення стає складним і ресурсоємним завданням. Тим часом клієнт втратичає прибуток. Більш того, якість сприйняття користувачів різко знизиться. У цьому випадку тривалість відновлення є найважливішим індикатором ефективності підтримки інфраструктури. Складність катастроф полягає в їх низькій передбачуваності. Катастрофа може прийти з нізвідки. Вона не може бути представленм регулярними формулами чи законами. У практичній IT-інженерії не дуже корисно використовувати складні методи розрахунку індексів надійності у випадку катастроф. Коли виникає катастрофа, індекси надійності не допоможуть відновити систему.

Тому важливо мати план відновлення інфраструктури після катастроф. Як правило, це може ґрунтуватися на рішеннях автоматизації. Більшість із них використовують поняття IaaC. Ця концепція передбачає, що інфраструктура може бути представлена як програмний код, який може бути додатково інтерпретований деяким інструментом автоматизації для хмарного API. Це дозволяє коду повторно використовуватися, якщо це необхідно, а це означає, що тривалість відновлення може бути зменшено до тривалості застосування змін до інфраструктури відповідно до заздалегідь визначеної програми.

Відповідно до популярного визначення, аварійне відновлення - це набір політик, інструментів та процедур, що забезпечують відновлення важливої технологічної інфраструктури або системи після природної або спричиненої людиною катастрофи.

Аврійне відновлення – це особлива галузь науки та техніки, яка охоплює проблему значно ширше, ніж просто сферу інформаційних та комунікаційних технологій. Так чи інакше, аварійне відновлення cloud ITC підтримує всі принципи та основи загальної теорії.

Дослідження систем аварійного відновлення можуть бути згруповані в кілька комплексів: перші розглядають питання про конфіденційність та технології безпеки, другі розглядають сам процес відновлення після аварійного завершення, треті пропонують деякі розширення для процесу аварійного відновлення, наприклад "багатоцільовий" підхід, який дозволяє відновлювати дані на декількох серверах за кількома методами. Зокрема, запропоновано використовувати традиційний базовий канал TCP/IP, образ та реплікацію. Досліджено використання гнучкої методології для поліпшення відновлення складних систем при катастрофічних сценаріях. Вони зосереджені на підході до аварійного відновлення, щоб подолати унікальні проблеми, з якими стикаються організації під час аварійного відновлення. Досліджено наслідки аварійного відновлення бізнесу. Вони доводять складність прогнозування катастроф, їх невизначеність, тривалість і ступінь. Вони пропонують рамки підтримки рішень для захисту організацій від наслідків катастрофи. Автори сформулювали своє завдання як багатофакторне завдання змішаного цілочисельного лінійного програмування для одночасного розподілу внутрішніх та зовнішніх ресурсів як для відновлення, так і для продовження роботи. Зокрема, їх метою є мінімізація тривалості відновлення. Представлено погляд на аварійне відновлення з точки зору постачальника хмарних технологій, тому що постачальники хмар мають надавати послуги своїм клієнтам, навіть якщо центр обробки даних внаслідок стихійного лиха вийшов з ладу.

Проте всі ці дослідження не охоплюють точних технологічних підходів та методів аварійного відновлення у хмарних системах. Наукова новизна запропонованого підходу виходить з припущення про використання концепції IaaC для аварійного відновлення функціонування системи.

Досліджено ефективність такого підходу у порівнянні з ручним відновленням після аварій. Дослідження базується на експериментальному підході з використанням реального приватного хмарного середовища [140], організованого на базі OpenStack та автоматичного формування інфраструктури з використанням HashiCorp Terraform [1]. Дослідження складається з набору

експериментів, що відрізняються кількістю хмарних компонентів та середньою кількістю залежностей на компонент. Здійснено імітацію спричиненої людиною катастрофи видаленням раніше створеної інфраструктури. Після цього виконувалися дії з аварійного відновлення у випадку ручного обслуговування та представлення інфраструктури як коду.

Концепція IaaC пропонує представлення хмарної інфраструктури будь-якої складності як програмного коду. Основною сферою її використання є автоматизація керування конфігурацією системи. Проте можливість представлення стану інфраструктури як керованого коду та застосування змін у коді на поточній інфраструктурі робить цю концепцію дуже корисною у випадку аварійного відновлення. Формально ми маємо справу з програмною моделлю інфраструктури, яка застосовується до реальних cloud систем.

Давайте розглянемо блок-схему роботи системи IaaC загалом та у випадку аварійного відновлення зокрема. Спрощений метод роботи представлений на Рис. 4.22 [12].

Для роботи методу потрібна дія користувача або якийсь попередній тригер, щоб почати виконання алгоритму. Після появи такого тригера система порівнює поточну модель з збереженим станом хмари, щоб визначити, чи існують деякі зміни в інфраструктурі з боку оператора. Цей етап є етапом планування. Результатом цього етапу є зручний опис ресурсів, які потрібно створити або видалити. У цьому випадку місцева стан є представленням стану хмарної системи після попередньої активації алгоритму керування конфігурацією.

У випадку катастрофи поточний стан системи відрізнятиметься від місцевого стану системи IaaC. Ця функція змушує оператора видаляти локальну інформацію про стан перед запуском блок-схеми IaaC.

Другий етап – застосування конфігурації. На цьому етапі керування системою здійснюється за допомогою її API для створення інфраструктури відповідно до заздалегідь визначеної конфігурації у коді програми.

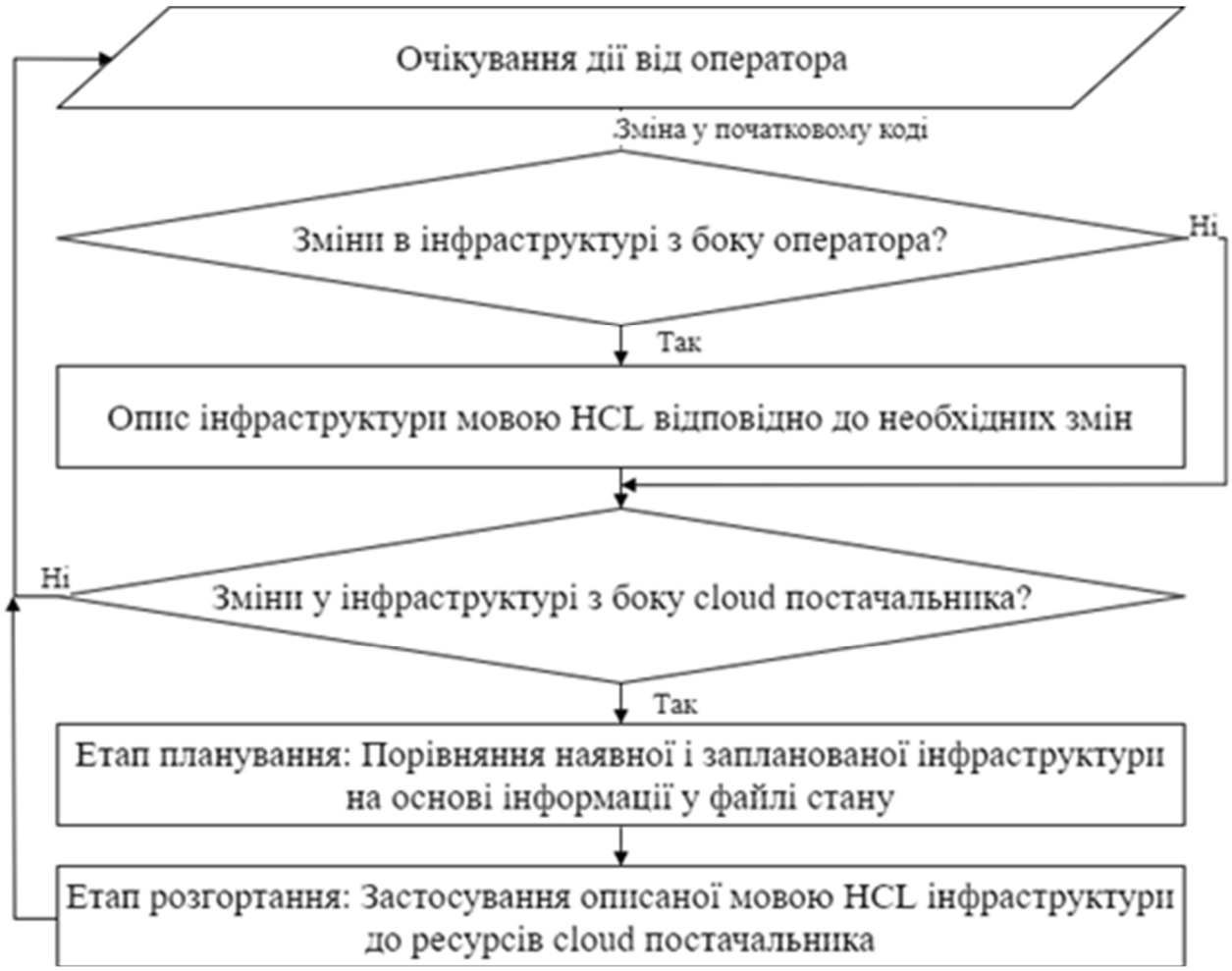


Рис. 4.22. Блок-схема створення хмарної інфраструктури з використанням Terraform, включаючи можливість відновлення після катастроф

Terraform від HashiCorp – це один з найбільш перспективних інструментів керування cloud системою, який швидко розвивається. Він включає концепцію IaaC і розроблений з використанням мови програмування Go. Він має безкоштовну дистрибутивну версію для кількох ОС та платформ. Завдяки простому синтаксису, заснованому на форматі JSON, цей інструмент дуже популярний серед операторів IT та інженерів DevOps.

Структурна схема типового проекту Terraform представлена на Рис. 4.23. Він складається з основного файлу, який представляє JSON опис потрібної інфраструктури. Основний файл зазвичай вимагає набору визначених користувачем змінних. Значення цих змінних збираються у спеціальному файлі з розширенням tfvars. Цей файл не може бути додатково доданий до системи

керування вихідним кодом, оскільки він може містити конфіденційну інформацію.

Зважаючи на вимогу багаторазового використання коду, найкращою практикою є використання модулів ресурсів. Ці модулі зазвичай містять сукупність залежних ресурсів, які не можуть існувати окремо, або їх окреме існування не має сенсу.

Після того, як код опису інфраструктури буде застосовано до ресурсів хмар, інструмент Terraform буде генерувати локальний файл у форматі JSON, який є деяким знімок з інфраструктури. Цей файл використовується для порівняння змін – це код із станом хмарної системи щоразу, коли ініціюється блок-схема IaaC.

План експерименту включає в себе приватне хмарне середовище на основі хмарної системи OpenStack. Ця приватна хмара є основою для створення та управління програмними інфраструктурами.

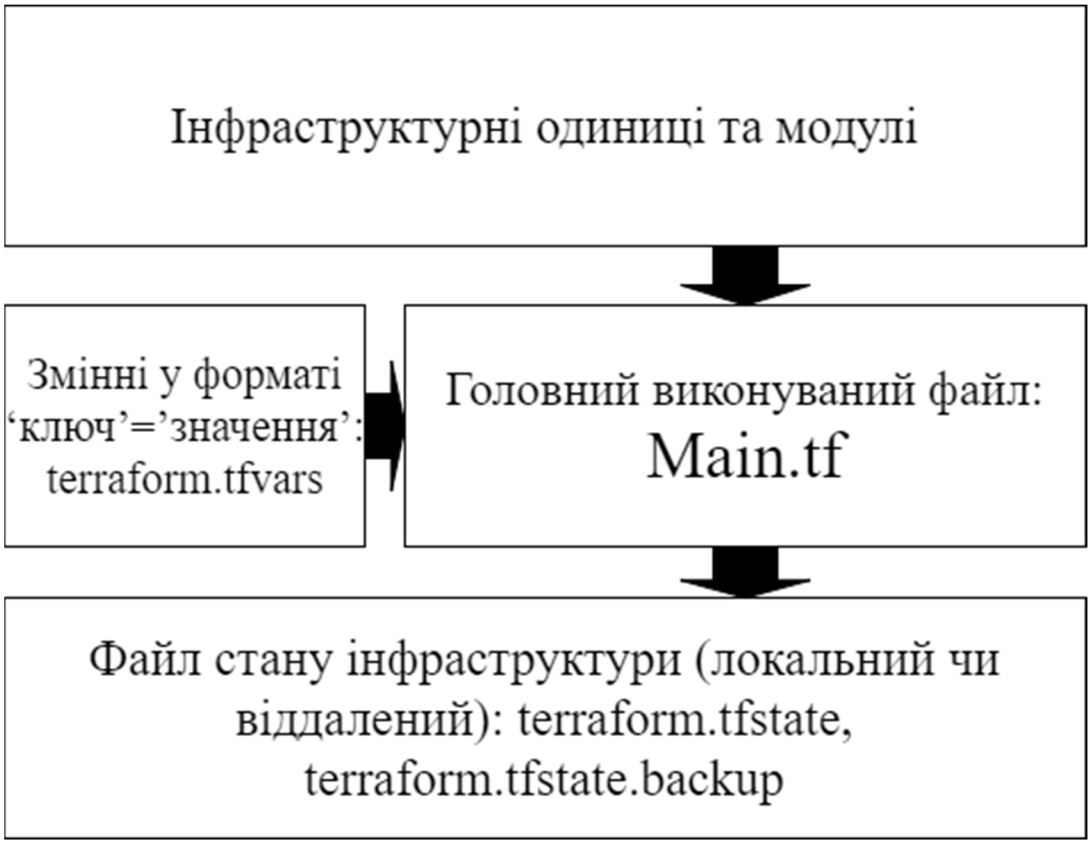


Рис. 4.23. Блок-схема типової структури проекту Terraform

Використано просту OpenStack конфігурацію з контролером і вузлами, розташованими на локальних комп'ютерах під операційною системою Centos 7 з віртуалізацією KVM. Контролер підключений до вузлів за допомогою зовнішніх мережевих адрес. Було підготовлено базовий обораз для розгортання віртуальної машини на Centos 7.

Приклад графу, представлений на Рис. 4.24, показує залежності хмарних ресурсів, створені інструментом IaaC Terraform на основі програмно-визначеної інфраструктури.

Рис. 4.24. Приклад графічного представлення хмарних ресурсів та їх залежностей, створених Terraform, на основі кодового представлення хмарної інфраструктури

Катастрофа була спричинена раптовим ручним руйнуванням всієї раніше створеної інфраструктури. Це означає, що хмарна система не містить будь-яких ресурсів, створених вручну або за допомогою блок-схеми IaaC. Після імітації катастрофи розпочато процес відновлення. Ми дослідили два типи дій відновлення: відновлення інфраструктури вручну та активація інструмента Terraform.

Таке моделювання було виконано з урахуванням різної кількості створюваних компонентів і різної кількості залежностей між cloud ресурсами. Середнє число залежностей для кожного компонента хмари розглядається як половина кількості створених хмарних компонентів.

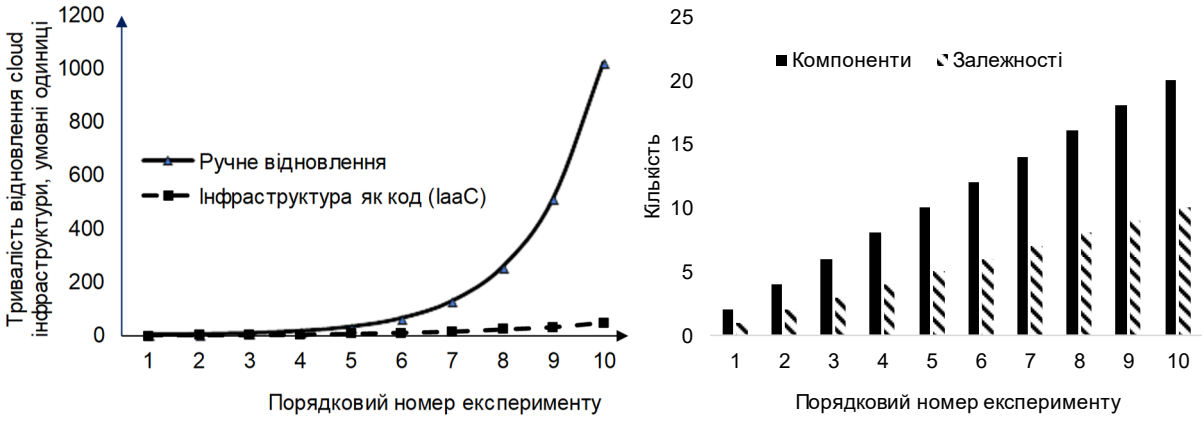
Результати дослідження представлені на Рис. 4.25. Нижче на графіку показано умови відповідного експерименту та його індекси. На верхній частині Рис. 4.25 показано співвідношення між тривалістю аварійного відновлення та

складністю хмарної системи (визначеною кількістю хмарних компонентів та середньою кількістю залежностей на кожний компонент хмари).

Як видно з Рис. 4.25, різниця між тривалістю відновлення після катастрофи при ручному та автоматичному відновленні різко зростає зі збільшенням складності хмарної системи. Представлені результати чітко виявляють користь від використання інструментів керування IaaC, таких як Terraform для аварійного відновлення в умовах хмарної інфраструктури. Цей виграш збільшується із збільшенням складності хмарного середовища. У нашому випадку він досягав значення 1000.

Ми розглядаємо порівняння ефективності різних засобів автоматичного аварійного відновлення як сферу для подальших досліджень, пов'язаних з темою представленої роботи.

Загалом, представлено додаткову особливість використання систем керування конфігураціями, яка може бути використанп для відновлення після раптових, як правило, спричинених людським фактором катастроф, таких як повне видалення всієї інфраструктури хмарної інфраструктури ITC.



а) б)

Рис. 4.25. Співвідношення між ручним та автоматичним відновленням працездатності у вимірі тривалості відновлення з урахуванням кількості cloud компонентів та середньої кількості залежностей між елементами у cloud інфраструктурі

4.5. Висновки до четвертого розділу

1. Розроблено нову модель корпоративного клієнта інформаційно-комунікаційної системи надання послуг. Ця модель відрізняється від відомих тим, що реалізована у вигляді програмного узагальненого коду, який внаслідок ініціалізації його параметрами, дає змогу утворити як завгодно багато об'єктів, що представляють корпоративну клієнтську інфраструктуру. Дана модель у подальшому використана у процесі навантажувального тестування як елемент інформаційно-телекомунікаційної системи надання послуг. Перевагою моделі є її масштабованість, тобто додавання нових елементів інфраструктури клієнта або ж видалення існуючих реалізується зміною у програмному коді і повторною ініціалізацією об'єкта інфраструктури корпоративного клієнта. Модель базується на принципі функцій корисності, який сформульовано у другому розділі роботи.

2. Розроблено нову модель спільної cloud інфраструктури. Ця модель реалізована у вигляді програмного коду, і як і попередня, базується на технології імітаційного моделювання. Вона дає змогу представити спільну мультиклієнтську інфраструктуру у вигляді ініціалізованого параметрами програмного об'єкта, який поєднано із клієнтським програмним об'єктом. Разом із логікою надання послуг ці дві моделі формують розподілену гетерогенну інформаційно-телекомунікаційну систему надання послуг, поведінка якої під впливом користувацького навантаження досліджена у шостому розділі роботи.

3. Розроблено та досліджено комплексну систему інформаційної безпеки у телекомунікаційному сегменті розподіленої гетерогенної інформаційно-телекомунікаційної системи. Дана система базується на моделюванні зловмисної поведінки та мережних атак на інформаційні ресурси до застосування заходів захисту та після їх імплементації. Результати показують, що досягнуто зниження впливу атак на інформаційно-телекомунікаційну систему більш ніж на 90%. Для підтвердження адекватності використаних моделей та системи у цілому проведено дослідження процесу реальної мережної атаки на велику інформаційну мережу та ефективності боротьби з такого роду інформаційними

загрозами. Встановлено, що запропонована система функціонує втричі ефективніше.

4. Досліджено процес відновлення інфраструктури після катастрофи на базі ручної та автоматичної активності. На підставі аналізу встановлено, що одним із найважливіших показників ефективності цього процесу є тривалість відновлення після аварій. Тому у процесі моделювання ми сфокусувалися на вивченні цього параметра. Ми оцінили тривалість в разі ручного та автоматичного відновлення на основі концепції IaaC. Отримані результати підтвердили переваги підходу IaaC для відновлення після аварій. Значення виграшу залежить від складності хмарної системи і в наших умовах досягає 1000 разів.

РОЗДІЛ 5. МОДЕЛІ НАДАННЯ ПОСЛУГ В ГЕТЕРОГЕННИХ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ СИСТЕМАХ

5.1. Процеси безперервної інтеграції та доставки інформаційно-комунікаційних послуг

5.1.1. Функціональна модель системи безперервної інтеграції та доставки інформаційно-комунікаційних послуг

Сьогодні основна частина різнорідних досліджень систем, орієнтованих на послуги, розглядає процес надання послуг. Його дослідження потрібне через значний вплив процесу на якість сприйняття користувача. Надання сервісу є складним завданням у сучасній розподіленій ІТС. Воно охоплює кілька шарів системної моделі: програми, сеанс, транспорт та мережу. Щоб забезпечити якість обслуговування, ми повинні вирішувати завдання на кожному рівні в сильній координації.

Найпоширенішою практикою є концентрація уваги в основному на мережевому рівні. Це пов'язано з надійністю, безпекою, масштабованістю та іншими проблемами, характерними для мереж. Проблеми комунікації в мережах з пакетною комутацією добре відомі і детально описані в багатьох джерелах, але вони все ще не розв’язані повністю [230, 231].

З іншого боку, великий вплив на надання послуг формує прикладний рівень. Через технологічний прогрес він змінюється дуже швидко, і послуги, розроблені лише кілька років тому, можуть зіткнутися з проблемою несумісності платформи. Через це екземпляр сервісу може бути недоступним деякий час, поки не з'явиться нова версія з виправленням помилок. Цей процес повинен повністю контролюватися та автоматично підтримуватися. Найкращою практикою у цьому випадку є використання процесу безперервної інтеграції [239, 240].

Доставка послуг [265] включає в себе процес інтеграції, який орієнтований на безперервний аналіз послуг на рівні застосування системної моделі. СІ є ключовим процесом забезпечення якості послуг. Він включає в себе контроль коду системи, аналіз якості коду, автоматизоване тестування, перетворення

початкового коду на виконувані файли та розгортання. CI повинен бути орієнтований на платформу початкового коду сервісу (мова програмування, допоміжні середовища, бази даних, тестові седовища).

Багато відомих учених, таких як Мартін Фаулер та багато інших, вивчили цей процес [268]. Відомо, що CI заснований на деякому CI-сервері, який підтримує весь процес інтеграції та спілкується з усіма іншими необхідними серверами. У даній роботі ми зосереджені на реалізації цього процесу за допомогою вільного програмного забезпечення.

CI гарантує, що деякі ізольовані зміни в початковому коді сервісу будуть швидко перевірені та перевірені в той момент, коли вони входять у великий код, розміщений на деякому сервері керування вихідним кодом. Важливою перевагою використання цієї методики є те, що всі зміни та результати швидко повідомляються зацікавленим сторонам. Основною метою CI є швидке виявлення дефектів, звітність та корекція. Існує програмне забезпечення для автоматизації процесу CI. Там повинно бути деяке основне середовище: налаштована мережа, платформа віртуалізації, достатньо обчислювальних ресурсів. На ранньому етапі щоденне формування виконуваних файлів було загальним стандартом. Еволюція цієї думки призвела до того, що формування виконуваних файлів має бути інтегроване для кожної істотної зміни у коді сервісу. Представлено модель системи, в якій CI сервер обробляє кожну зміну вихідного коду. Цього можна досягти, використовуючи стек автоматизованих інструментів [263].

CI базується на кількох загальних практиках: єдине сховище початкового коду, автоматизація створення, практика самотестування, підтримка кожного завдання, швидке створення, середовища тестування та експлуатації повинні бути еквівалентними, існує єдина точка виконання, система формує автоматизовані звіти, підтримує автоматичне розгортання.

Типовий процес CI складається з декількох взаємопов'язаних етапів. Пропонована модель процесу представлена на Рис. 5.1. Рекомендовано використовувати сервер GitLab як систему керування кодом сервісу, сервер

SonarQube як систему перевірки якості коду, PostgreSQL як систему керування базами даних, TomCat і Docker-сервери як два різних середовища розгортання, сервер Nexus як сховище артефактів процесу. Також можна використовувати формування RPM пакетів інтегрованого сервісу та передавати сформований пакет в локальний репозиторій для цілей розповсюдження. Необхідно відзначити, що сховище з пакетами rpm має бути приватним, однак локальне сховище з деякими залежностями, необхідними для побудови сервісу, має бути загальнодоступними. Ми використовуємо дві методики розгортання сервісу: класичну технологію хостингу та докеризацію застосунку на основі мікросерверного підходу.

Якість сервісу залежить від багатьох факторів, таких як доступність кожного сервера, продуктивність мережі, узгодженість баз даних, надійність системи на окремому пристрої, завантаження системи тощо. Все програмне забезпечення, необхідне для розробки такої системи, може вільно використовуватись і доступно для CentOS на основі Linux. Це означає, що система може бути інтегрована в інформаційне середовище без додаткових витрат [17].

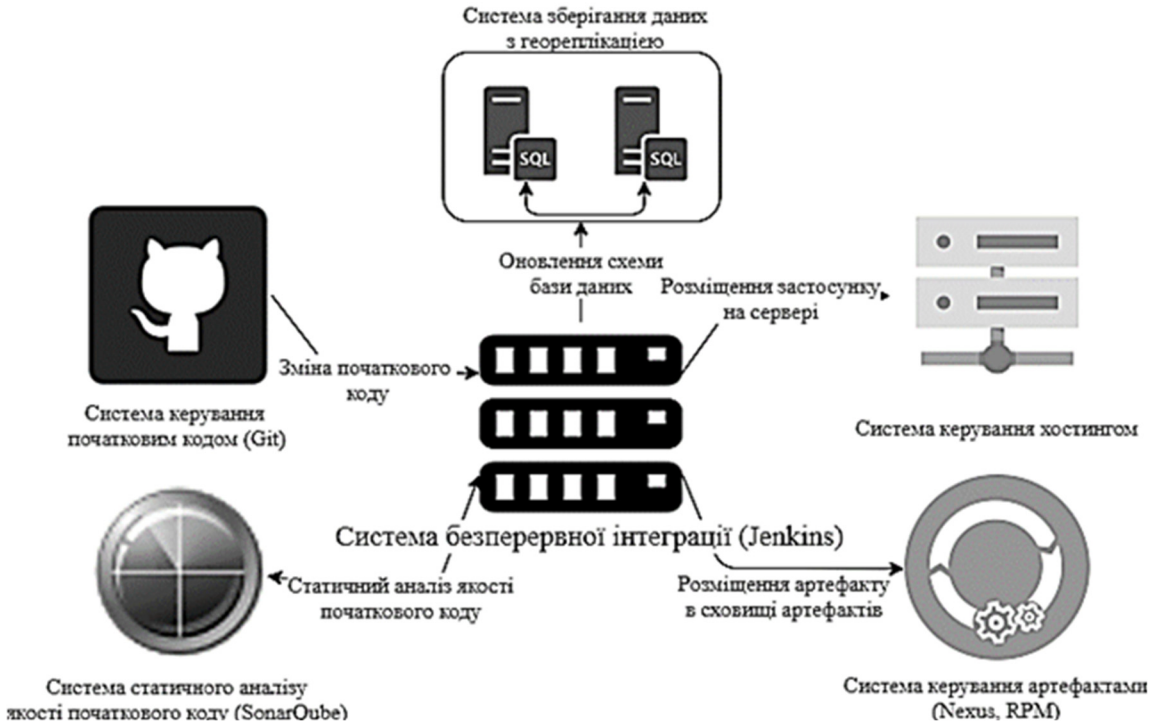


Рис. 5.1. Запропонована реалізація CI процесу

Метою керування початковим кодом є безперервний аналіз змін коду командою розробників. Кожен сервіс містить в своїй основі програмне забезпечення. Таким чином, життєвий цикл сервісу включає в себе життєвий цикл програмного забезпечення. Потрібно керувати початковим кодом через гнучкість та зручність використання. По-перше, це забезпечує присутність єдиної точки присутності коду разом із віддаленим сховищем, де всі члени команди розміщують код. По-друге, це забезпечує технологію розгалуження. Розгалуження дозволяє виконувати незалежні підзавдання, продовжуючи подальше об'єднання в єдиний контрольований проект, який є остаточним сервісом. Система керування початковим кодом є відправною точкою процесу безперервної інтеграції, і її функціонування може бути відображено робочим процесом на Рис. 5.2. Ми використовуємо GitLab сервер керування початковим кодом у системі безперервної інтеграції.

Він зобов'язує членів команди перейти до своїх локальних сховищ, провести розробку призначених компонентів, а потім об’єднати зміни в основному відгалуженні, призначеному для розробки сервісу. Кожен розробник сервісу використовує власне відгалуження. Лише одна особа має можливість об'єднати зміни у всіх відгалуженнях. Ця методика забезпечує єдину робочу версію, яка буде існувати у головному відгалуженні.

VCS є відправною точкою в процесі CI. Кожне оновлення головної гілки – тригер для сервера безперервної інтеграції (у нашому випадку – Jenkins). Це означає, що кожна зміна вихідного коду буде проаналізована, і буде виконана перевірка результату виконання цього процесу.

Крім того, має існувати зворотній зв'язок від Jenkins до GitLab (сервер керування вихідним кодом), щоб інформувати цей сервер про результат перетворення початкового коду на виконувані файли.

Цю версію також може перевірити сервер SonarQube на основі статичного аналізу коду. Необхідно забезпечити високу якість коду на основі заздалегідь визначених правил для відповідної мови програмування, зібраних в профілі

якості. Ми використали сервер SonarQube з конфігурацією, що представлена на Рис. 5.3.

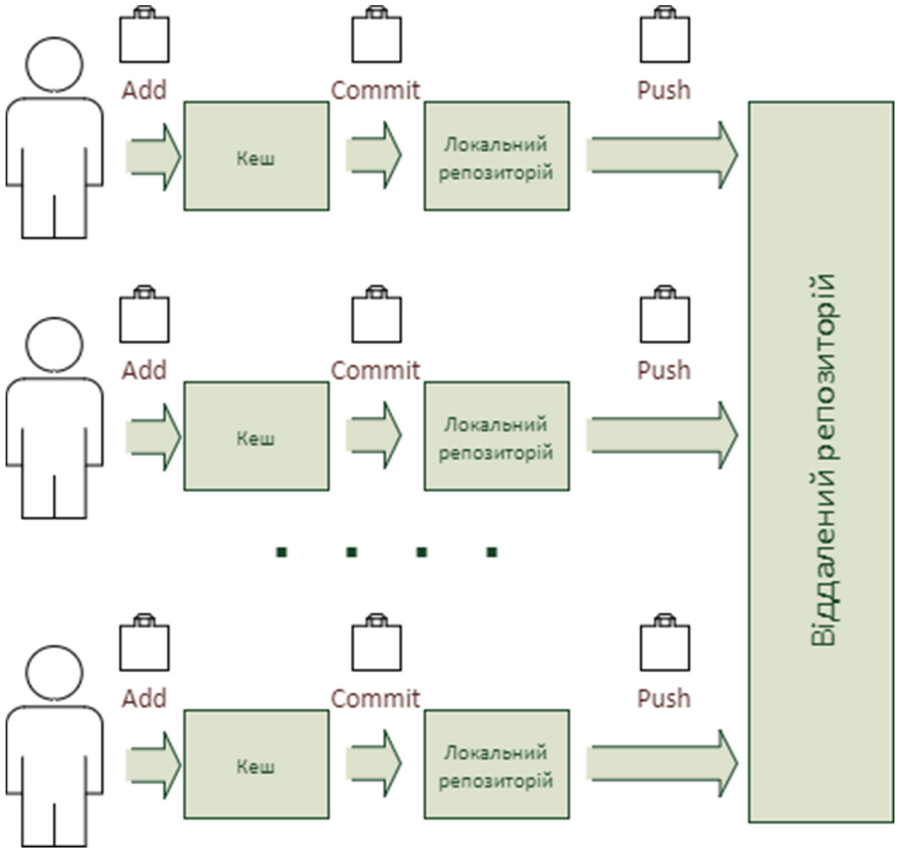


Рис. 5.2. Модель функціонування системи контролю версій Git

Результати сканування повинні аналізуватися за допомогою деяких показників. Це може бути виконано на основі порогів якості. Це вбудована функція SonarQube, яку можна налаштувати відповідно до конкретних вимог до обслуговування. Як правило, пороги якості включають такі показники, як покриття за допомогою тестів, надійність коду, безпека коду, підтримуваність коду.

Пороги якості – це найкращий спосіб запровадити політику якості у випадку статичного аналізу коду. Вона повинна відповісти на питання, чи готовий сервіс бути доставленим клієнту. Поки процес аналізу ведеться, SonarQube генерує проблему кожен раз, коли код порушує правило якості в профілі. Проблема характеризується складністю та впливом, який може бути блокуючим критичним, суттєвим, незначним або інформаційним. Складність повідомляє про

те, що повинен бути застосований набір дій протягом певного часу для вирішення проблеми.

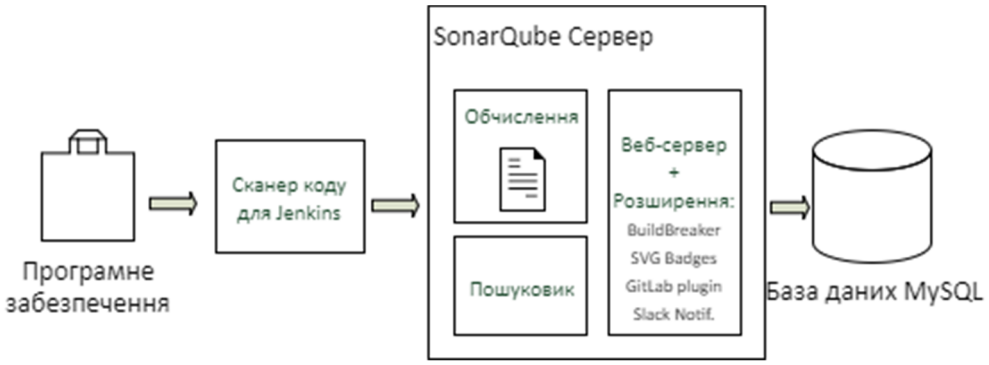


Рис. 5.3. Модель процесу статичного аналізу якості програмного коду із використанням сервісу SonarQube

Використано Jenkins як сервер, що контролює процес безперервної інтеграції. Це найважливіша складова процесу, що має роль оркестратора. Він очікує, поки не спрацює тригер початку процесу безперервної інтеграції від VCS. Відразу після того, як тригер виявлено, Jenkins розпочне процес збірки. Він реалізується шляхом додавання плагіна GitLab до Jenkins. Збірка починається з аналізу якості статичного коду на базі SonarQube-сервера. Для цього Jenkins використовує плагін під назвою Sonar Scanner, який робить повне сканування і надсилає результати на сервер SonarQube.

Jenkins рекомендовано використовувати за принципом розподіленої архітектури. Принаймні, нам потрібні 2 віртуальні машини. Перша є основою для основного вузла Jenkins, який є безперервним інтегратором. Друга – це машина для виконання збірки. Необхідність цього підпорядкованого вузла дуже мотивована. Нам потрібен вузол, налаштований для побудови певного типу проекту. Він повинен включати кодову платформу, інструмент створення, деякі додаткові елементи, необхідні для успішного збирання проекту. Така архітектура CI відокремлює процеси інтеграції та збирання, що дає можливість узагальнити CI процес незалежно від технологій, що використовуються в процесі розробки сервісів.

У процесі досліджеггя маємо справу з веб-застосунком на базі Java. Тому ми потребуємо Java Development Kit, попередньо встановленої на машині збирання, а також Maven, який є інструментом для створення Java-проектів.

Концепцію реалізації Jenkins наведено на Рис. 5.4.

Рис. 5.4. Структура системи безперервної інтеграції

Як результат, Jenkins відповідає на питання, чи проект готовий до розгортання та доставки замовнику. Існує кілька умов для успішного проходження до розгортання: успішне підключення до сервера баз даних, успішні тести, успішне створення виконуваних файлів і проходження аналізу якості коду. У випадку, якщо одна з цих умов не виконана, розгортання відповідної версії буде скасовано. Цей процес гарантує, що лише працездатна та тестована версію може бути розгорнута.

Легко побачити переваги розподіленої архітектури Jenkins. Підключення іншої виконавчої машини забезпечить незалежне середовище розробки. Таким чином, та ж послідовність кроків може використовуватися для різних проектів і послуг на основі різних програмних технологій.

Безперечно, вибір сервера баз даних – це процес, залежний від проекту. У нашому випадку ми маємо справу з базою даних SQL. Тому MySQL або PostgreSQL можуть бути реалізовані як RDBMS. Якщо початковий код доступний, нам потрібно змінити параметри з'єднання з базою даних, щоб сервіс працював з вибраною СУБД. Інший спосіб полягає у визначенні відповідних змінних в описі Jenkins проекту.

Інший момент стосується архітектури СУБД. Базовою практикою є відокремлення баз даних для тестування та основних проектних. Крім того, ми повинні бути впевнені щодо узгодженості даних у випадку відмови сервера бази даних або аварії. У нашому випадку ми використовуємо архітектуру master-slave без дублювання кластерів (Рис. 5.5).

Ми використали зовнішній інструмент під назвою pgpool, щоб зробити нашу архітектуру бази даних більш масштабованою. Як видно з Рис. 5.5, використовується лише один пул, але легко створити додатковий. Єдине, що потрібно – це створити достатню кількість віртуальних машин, тому що для кожного екземпляра бази даних потрібна одна віртуальна машина.

Запропонована нами архітектура забезпечує відмовостійкість даних, пов'язаних з користувачем. У випадку, якщо один екземпляр знаходиться в неробочому стані, існуюча пара повинна працювати як master-slave. Таким чином, база даних буде недоступна лише у випадку, якщо два з трьох екземплярів зникнуть. Додавання іншого пулу (аналогічного) зменшить ймовірність відмови бази даних.

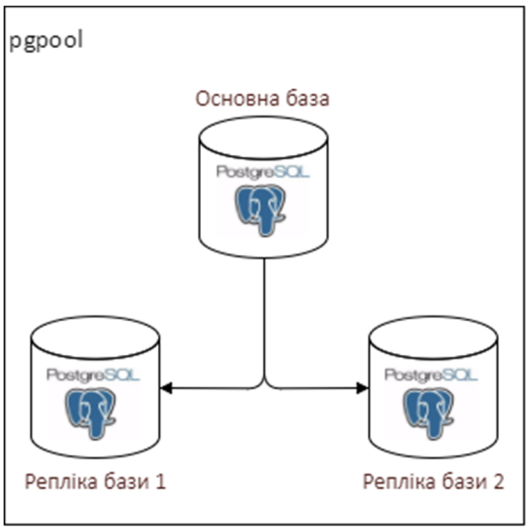


Рис. 5.5. Архітектура системи зберігання даних користувача в процесі CI

Якщо всі попередньо визначені кроки будуть успішними, сервіс може бути розгорнутий та розміщений в деякому сховищі, звідки він буде доступний для завантаження для деяких користувачів. Ця мета забезпечується за допомогою сервера репозиторіїв, Nexus у нашому випадку. Крім того, Nexus є хорошим

інструментом для створення локальної копії сховищ, які використовуються в процесі побудови сервісу, наприклад, репозиторій yum для репозиторіїв CentOS та Maven. Локальна копія дозволяє швидко довантажувати необхідні залежності. Слід підготувати файл rpm нашого сервісу для розміщення в сховищі. Це окреме завдання.

Після того, як всі процеси завершено, сервіс нарешті готовий до розгортання. Існує кілька способів вирішення цього завдання. Найбільш доречним сьогодні є докерування програми з відповідним середовищем.

5.1.2. Підвищення ефективності повного групового розсилання у розподілених ІТС

Розвиток ІТС характеризується переходом від персональних до спільних платформ обслуговування, і це ставить нові завдання, які пов’язані із процесами надання послуг із належною якістю. У роботі виконано моделювання та дослідження особливостей методу повного впорядкованого групового розсилання [20].

Для інформаційних технологій властивий швидкий прогрес засобів і методів обробки інформації. Бурхливий розвиток та еволюція обчислювальних систем і значні вимоги до різного роду ресурсів спричини повсюдне використання розподілених гетерогенних систем для надання послуг [20].

Важливим компонентом цих систем є розподілені георепліковані бази даних, які дають змогу як підвищити відмовстійкість системи в цілому, так і наблизити дані до користувача. Це забезпечується процесом реплікації даних, і дані знаходяться в актуальному стані у всіх копіях бази. З іншого боку, перехід до баз даних такого типу призводить до характерних проблем розподіленого опрацювання даних, пов’язаних із, як правило, значною географічною віддаленістю процесів між собою, а завдання, пов'язані з розподіленим опрацюванням інформації, можуть виконуватися конвеєрним способом в різні інтервали часу і навіть в різних часових зонах. У зв’язку з цим синхронізація процесів та узгодження даних у розподілених ІТС є актуальним завданням [20].

Організація узгодженого функціонування процесів потребує визначеного алгоритму взаємодії [91, 116, 192]. Загальноприйнятим для класу систем паралельних обчислень вважається організація взаємодії процесів через спільні змінні середовища. Такий метод передбачає, що деякий процес записує нове значення в одну з таких змінних, а інший – виконує зчитування. Однак, у розподілених ІТС процеси комунікують через обмін повідомленнями. У зв'язку з цим обмін даними і взаємна координація потребує використання методів надсилання і приймання повідомлень [20].

Особливістю розподілених систем є те, що точний точний час настання певної події (надсилання або ж отримання повідомлення) не є важливим. Визначальним є порядок настання цих події [118]: необхідно точно встановити, чи сталася деяка подія, асоційована з процесом, раріше або пізніше певної події в іншому процесі. Тому виконання процесів описується в термінах настання подій. [20].

Удосконалення комунікації процесів на рівні сервісів є важливою і актуальною сферою у галузі сервісної інженерії. На сьогодні якість сприйняття послуг або досвід кінцевого користувача формується не лише мережею передавання даних, а визначається також процесами взаємодії елементів сервісів на прикладному рівні ІТС. Одним із основних методів такої комунікації є повне впорядковане групове розсилання повідомлень між процесами елементарних сервісів у ІТС. Проте, протокол, який реалізує згаданий метод, не враховує можливості втрати повідомлень у процесі передавання. На практиці це може призвести до відмов під час обміну повідомлення між елементарними сервісами, і як наслідок, до недоступності сервісу, який використовують клієнти. Зважаючи на викладене, підвищення ефективності цього методу в контексті зменшення числа відмов є актуальним завданням. Це забезпечить зниження імовірності відмови у процесі користування композитними застосунками в ІТС [20].

Дослідження методу повного впорядкованого групового розсилання проведемо на основі комунікації у каналах зв'язку двох типів між процесами: канал FIFO та non-FIFO [8].

Збереження порядку переданих по каналу повідомлень характерне для черг повідомлень FIFO. Отже, у випадку надсилання повідомлення m_1 раніше, ніж повідомлення m_2 по деякому каналу ІТС на прикладномму рівні, доставка першого повідомлення відбудеться у першу чергу, а другого повідолення – у другу чергу.

Нехай представимо канал Non-FIFO як деяку множину, у яку процес-відправник додає призначене для передавання повідомлення, а вилучення їх із черги у випадковому порядку реалізує процес-адресат.

Рис. 5.6 дає змогу охарактеризувати процес оновлення інформації у базах даних із застосуванням двох досліджуваних типів каналів. Отримані результи дають можливість стверджувати, що за наявності каналу типу non-FIFO алгоритм повного впорядкованого групового розсилання втрачає ефективність, оскільки мітки часу повідомлень, які перебувають у системній черзі, не впорядковані, що призведе до важких наслідків для постачальника послуг – втрати інформації користувача. Однак у випадку використання каналу FIFO черга повідомлень є впорядкованою, тобто значення часової позначки кожного наступного повідомленя більше від попередньої [20].

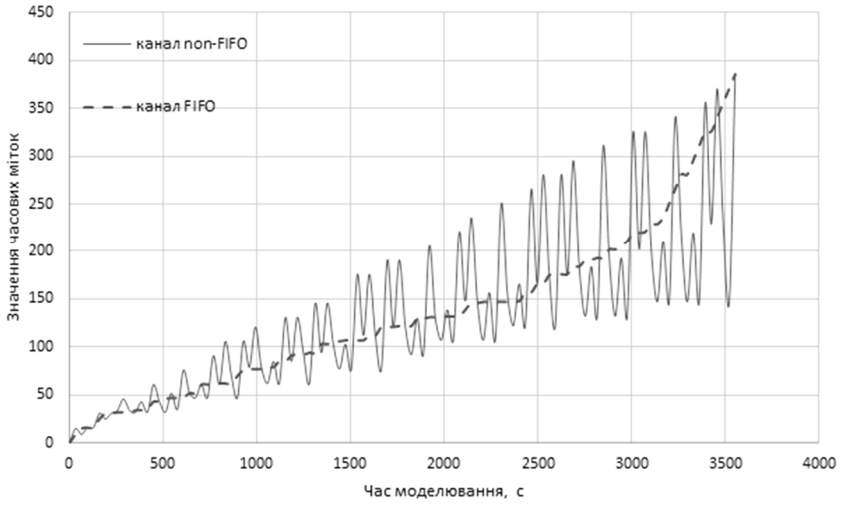


Рис. 5.6. Характеристика процесу призначення повідомленням часових позначок у каналах FIFO та non-FIFO у випадку застосування алгоритму повного впорядкованого групового розсилання

Середню тривалість синхронізації процесів прикладного рівня з точки зору повного циклу групової комунікації подано на Рис. 5.7 за час моделювання для двох досліджуваних типів каналів зв'язку між процесами: FIFO і non-FIFO. З ориманих результатів можемо спостерігати, що алгоритм повного впорядкованого групового розсилання втрачає свою ефективність під час використання каналу non-FIFO, виходячи із діапазону тривалості синхронізації – для цього типу каналу це значення знаходиться в межах 1,7 – 3,2 с, а у випадку каналу FIFO воно знаходиться в межах 0,7 с. Необхідно прояснити, що у процесі моделювання ІТС із каналом non-FIFO не було враховано втрату повідомлень [20].

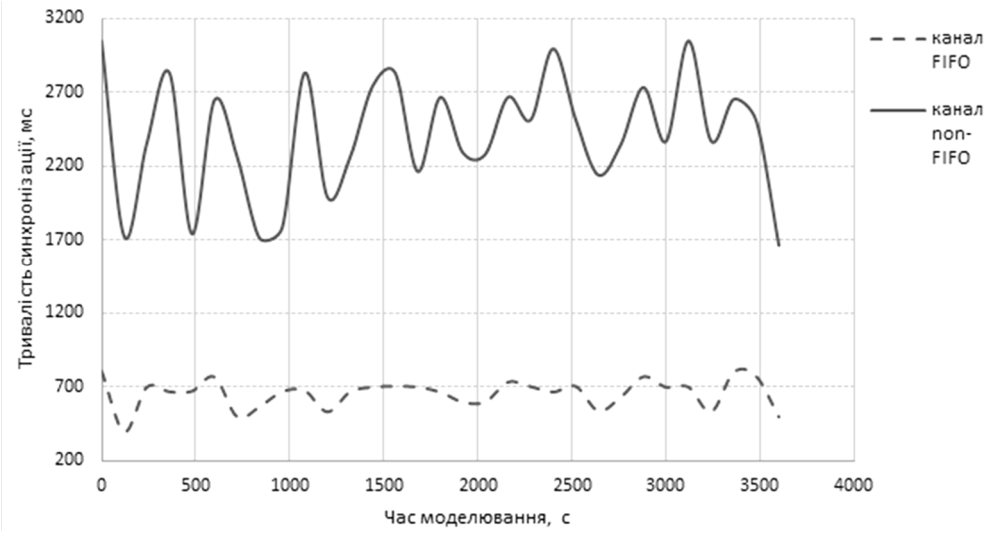


Рис. 5.7. Усереднене значення часу синхронізації даних у розподілених базах у процесі моделювання з використанням двох досліджуваних каналів (без врахування ефекту втрати повідомлення)

Результати, представлені вище, дають можливість підсумувати, що стандартний метод складно адаптувати до використання в реальних ІТС, оскільки він ефективно функціонує лише із використанням каналу FIFO. У такому випадку залишається лише покладатися на засоби надійного передавання даних нижчих рівнів [20].

Підвищення ефективності досліджуваного методу повного групового розсилання базується на надсиланні взаємних підтверджень щодо успішності доставлення повідомлень, що забезпечує адаптиність удосконаленого методу до

передавання у каналах non-FIFO. Алгоритм функціонування модифікованого методу можна описати таким чином.

1. Якщо поступає запит на модифікацію інформації в розподіленій базі даних, координуючий процес пересилає широкомовне повідомлення всім іншим процесам, які перебувають в системі. У повідомлення слід помістити такі параметри: часову позначку, унікальний ідентифікатор процесу, положення аналізованого повідомлення у локальній черзі процесу, який готує його надсилання. Алгоритми взаємного виключення забезпечують можливість використання останнього параметра. Ці алгоритми впорядковують запити ще перед початком запису/зчитування даних з розподіленої георепліковаої бази. Після одержання запиту інші процеси, крім координатора, аналізують контент отриманого повідомлення, у якому першочерговий пріоритет має параметр розташування в локальній черзі процесу-відправника [20].

2. Розташування повідомлення у черзі відправника є визначальним для встановлення розташування повідомлення у локальній черзі процесів-отримувачів. Слід врахувати, що може настати такий момент, коли певна кількість процесів можуть надіслати повідомлення з однаковими ідентифікаторами положення в черзі, отже у такому разі конфлікт запропоновано розв’язувати на підставі часових позначок процесів.

3. Після опрацюваня запиту і визначення його розташування у локальній черзі процес-одержувач надсилає повідомлення-підтвердження до відправника запиту. Необхідно відзначити, що ідентифікатор процесу-відправника повинен бути доданим у повідомлення.

4. У цей же час процес-ініціатор запиту очікує на підтвердження від всіх процесів, наявних в ІТС.

5. Підтвердження отримання забезпечує визначення того, чи не відбулося втрати повідомлення з моменту його надсилання. Приймемо, що тривалість очікування підтвердження обмежена вчасі. У випадку, коли за цей період не надходять всі повідомлення-підтвердження, то повідомлення-запит будемо розглядати як втрачене.

6. Останній етап – це надсилання процесом-відправником повідомлення до процесів, підтвердження від яких уже отримано, про видалення цього повідомлення з черги на опрацювання.

Метод, запропонований у роботі, має ряд недоліків [20]:

- часті втрати повідомлень суттєво збільшують частку службового трафіку у ІТС;
- не розглядаємо втрату повідомлення щодо видалення з локальної черги процесів, а приймаємо, що надійність комунікаційного каналу достатня для доставлення повідомлення-підтвердження.

У цілях дослідження здійснено імітаційне моделювання процесу комунікації на прикладному рівні між процесами розподіленої гетерогенної ІТС.

У скороченому вигляді процес імітаційного моделювання виглядає так [20]:

- утворюється розподілена геореплікована база даних (передбачає множину реплік бази даних, що розміщені в різних територіальних регіонах);
- симулюється процес обміну повідомленнями в ІТС. Ці повідомлення абстрагують механізм опрацювання даних.

У процесі моделювання слід гарантувати узгодженість реплік розподіленої георепліковаої бази даних за рахунок впорядкування повідомлень, що поступають у ІТС.

Алгоритм функціонування моделі представлено програмною реалізацією на мові програмування C# із застосуванням технології мультипотоковості. Відзачимо, що у цьому експерименті застосовано умовну мультипотоковість, що обмежено відображає особливості реальної розподіленої системи. Цим пояснюється обмеження, які накладено на процес моделювання. [20].

Блок-схемою на Рис. 5.8 представлено удосконалений метод повного групового розсилання. Відмінність його від відомого полягає у обмеженні максимального часу очікування на підтвердження від процесів адресатів на запит, насділаний процесом-відправником. Тому на прикладному рівні хід координації процесів не допускає зависання комунікації, що можливо за умови безмежного очікування на підтвердження. Способом розв’язання комунікаціної

відмови є повторна спроба здійснення координації, що гарантує отримання користувачем реакації ІТС на свої дії у обгрунтований проміжок часу [20].

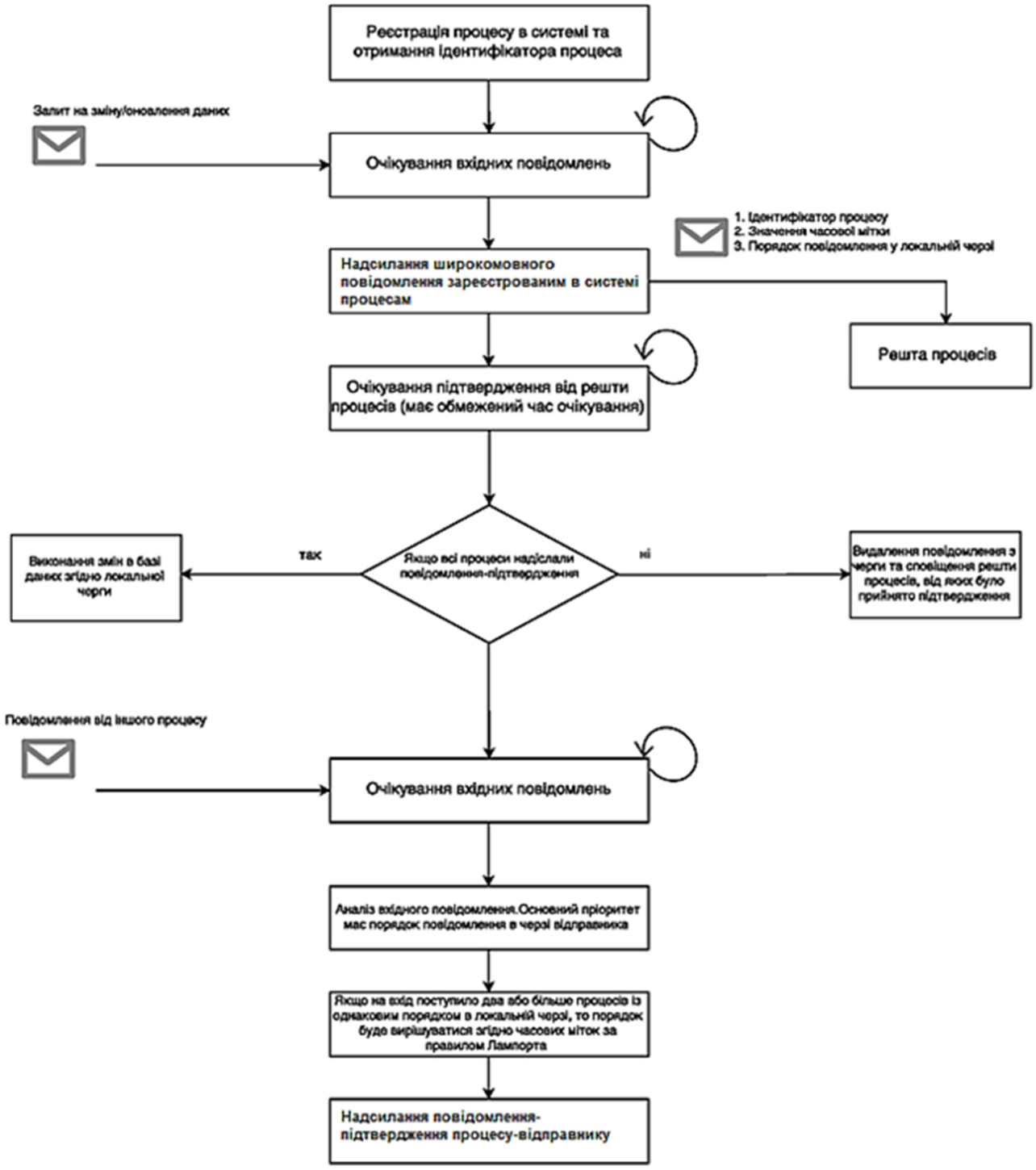


Рис. 5.8. Представлення удосконаленого методу повного групового розсилання у розподілених ІТС у вигляді блок-схеми

Рис. 5.9 відображає порівняння ефективності роботи стандартного та удосконаленого алгоритму групової комунікації для non-FIFO каналу зв’язку.

З аналізу результатів легко бачити, що удосконалений метод повного групового розсилання демонструє свої переваги, коли для комунікації використано модель каналу non-FIFO у порівнянні зі стандартним. Відзначимо, що у випадку удосконаленого методу значення часових позначок в черзі на обслуговування постійно зростає, однакові значення відсутні. Ці характеристики представленого методу відповідають вимогам синхронізації подій і часу в розподілених ІТС і забезпечує консистентність геореплікованої бази даних [20].

На Рис. 5.10 відображено зміну середнього часу узгодження реплік баз даних у процесі моделювання для двох методів групової комунікації з використанням non-FIFO каналу. [20].

Рис. 5.9. Порівняння ефективності функціонування стандартного та удосконаленого алгоритму повного групового розсилання для non-FIFO каналу зв’язку

Здійснено імітацію втрати одного повідомлення, щоб показати відмінності поведінки системи із використанням двох досліджуваних методів групової комунікації. У випадку втрати повідомлення стандартний метод відмовляє, тобто тривалість синхронізації стрімко росте, а процес припиняється. Цього не спостерігається із застосуванням удосконаленого методу. Відзначимо, що у момент втрати повідомлення суттєво зростає тривалість узгодження. У цьому випадку тривалість узгодження близька до максимального часу очікування підтвердження [20].

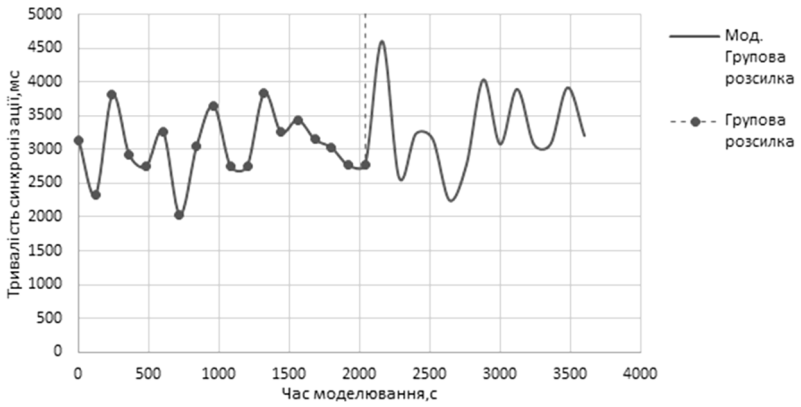


Рис. 5.10. Зміна усередненого часу узгодження реплік баз даних у процесі моделювання для двох методів групової комунікації з використанням non-FIFO каналу (одне повідомлення втрачається)

В результаті аналізу виявлено, що відомий метод має недолік у процесі обслуговування подій з non-FIFO чергами. Він не здатний ефективно функціонувати в ІТС, для яких характерні високі інтенсивності надходження запитів на обслуговування. Проведені дослідження дали змогу запропонувати та обґрунтувати удосконалений метод повного впорядкованого групового розсилання. Виконано моделювання його роботи для підтвердження того, що вдалося знизити імовірність відмови у процесі комунікації на прикладному рівні в сучасних розподілених ІТС, тим самим досягнувши підвищення ефективності повного групового розсилання [20].

5.2. Модель програмованого компонента інформаційно-телекомунікаційної системи

5.2.1. Інструментальні засоби розробки програмованих компонентів

На сьогоднішній день існує велика кількість програмних засобів, які забезпечують дослідження поведінки елементів ІТС і навіть систем цілому. До таких собів належать мережні симулятори (наприклад NS, OPNET, NetSim, OmNET++). Завдяки цим технологіям є можливість вивчати поведінку систем та їх компонентів, а також досліджувати вплив розроблених методів та модей на функціонування ІТС. В основ їх роботи лежить моделювання дискретних подій з постійним кроком. Важливо зазначити, що через використовуютання цими

засобами методів математичної статистики для оцінки стану системи в певний момент часу, періоди роботи реальної ІТС перетворюються у десятки секунд модельного часу [2, 5]. Це може стати проблемою, коли відбувається моделювання процесів у реальному масштабі часу. До типових завдань подібного класу належать: дослідження процесів функціонування буфера ІТС, дослідження процесів, які відбуваються з чергами мережних пристроїв [40].

Вищенаведене пояснює причину того, для проведення експериментів розроблено програмний компонент ІТС. Ісінструментальні засоби розробки такого компонента – Qt5.2 і C++. Мультиплатформенність згадаих засобів є однією з основних переваг обраного підходу [40].

Вивчення поведінки розроблених програмних компонентів ІТС є першочерговим завданням з огляду на потребу отримання надійних результатів дослідження на їх основі у майбутньому. В основу вивчення покладено рекомендації RFC 2544 (стандарти із функціонуванян глобальних мереж). Цей документ є основою для проведення тестів мережних пристроїв фізичного та рівня об'єднання мереж відповідно до мережної моделі TCP/IP [40, 86].

Верифікація роботи генератора мережного трафіку передбачає оцінку його максимальної продуктивності, при якій забезпечується стабільний інтервал між пакетами. Оцінка середньоквадратичного відхилення значень інтервалу між пакетами визначає похибку роботи програмного генератора та суттєво впливає на адекватність моделі. Верифікація програмного генератора здійснюється за допомогою набору дослідів, у якиз відрізняються параметри генерованого трафіку [40].

Рис. 5.11 представляє значення інтервалів між 100 000 пакетів. Інтенсивність потоку у всіх дослідах становить 31,25 кбіт/с, інтервал між пакетами 10мс, а довжина пакету – 2500 байт [40].

Статистичні характеристики цього випадкового процесу такі [40]:

- середнє значення тривалості інтервалу між пакетами – $1 * 10^7$ нс;
- дисперсія інтервалу між пакетами – $1,7 * 10^{12}$ нс;
- середньоквадратичне відхилення – $1,3 * 10^6$ нс;

– абсолютне вдхилення середнього значення – 4,2 *10^3 нс.

Продуктивність програмних компонентів ІТС, як і у випадку апаратних вузлів, безпосередньо впливає на процес обслуговування інформаційних потоків. Однак, слід пам’ятати, що програмні компоненти мають бути розгорнуті на деякій апаратній інфраструктурі, тому їх продуктивність взаємопов’язана, здебільшого прямопропорційно [40].

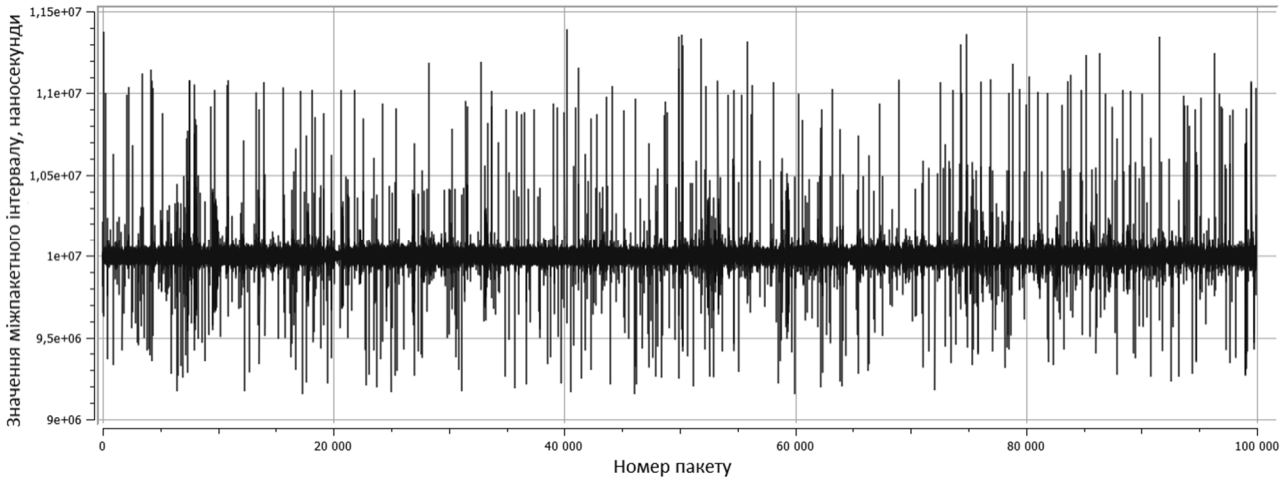


Рис. 5.11. Випадковий процес значень інтервалу між двома послідовно згенерованими пакетами

Деякий програмний компонент, який розгорнуто на різних фізичних серверах з різною продуктивністю, може по-різному реалізувати процес опрацювання пакетів з точки зору часових характеристик. Оцінка продуктивності програмного компонента здійснюється із залученням таких фізичних ресурсів [40]:

– процесор Intel Core i5-2410M із тактовою частотою ядра 2.30 ГГц;
– оперативна пам’ять DDR3 6 ГБ;
– мережний інтерфейс Realtek PCIe FE Family Controller 1 Гбіт/с.

Характеристики кожного досліду для дослідження продуктивності програмного компонента ІТС подано у таблиці (Таблиця 5.1). Число пакетів задано у 100 000, а міжпакетний інтервал обрано постійним – 10 мс. Інші параметри підлягали зміні у процесі моделювання [40].

Таблиця 5.1. Дослідження середньої тривалості обробки пакету програмним комутатором на обраному апаратному сервері [40]

Дослід	Довжина пакету, байт	Середній час затримки, мкс
1	64	1,24
2	128	1,79
3	512	1,98
4	1024	2,12
5	2048	2,2

5.2.2. Особливості функціонування програмованих компонентів як вузлів інформаційно-телекомунікаційної інфраструктури

Тестування вимагає введення маршрутизатора в режим перевантаження, через це необхідно визначити максимальну продуктивність програмного маршрутизатора. Було обрано два комп’ютери, на яких було проведено експеримент. Конфігурація цих комп'ютерів забезпечує високу продукитивність роботи програмного забезпечення. На двох комп'ютерах було встановлено операційну систему Microsoft Windows 7 Ultimate [39]. На першому комп’ютеру встановлено генератор трафіку, який приймає трафік одночасно, виходячи з цього можна обчислювати різницю між моментами відправлення та отримання кожного пакету. Враховуючи це генератор обчислює їх оброблення на маршрутизаторі та тривалість проходження пакетів мережею. Структурна схема експерименту показана на Рис. 5.12.

Під час першого тесту було задано потік пакетів, під час якого маршрутизатор функціонує нормально. У кожному наступному тесті інтенсивність потоку збільшувалася до того моменту, поки не почали формуватися черги, які суттєво впливають на затримку пакетів. Вибране значення розміру для програмного маршрутизатора вхідного буферу становить 24 пакети.

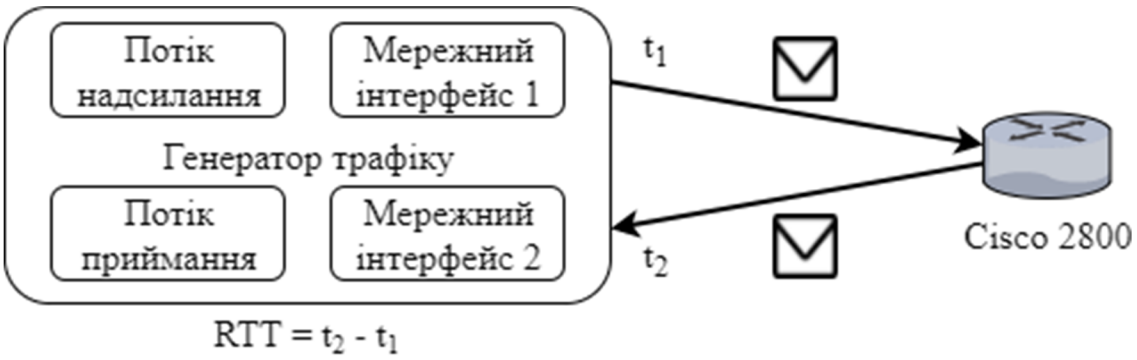


Рис. 5.12. Концепція визначення продуктивності програмного компонента ІКС

Під час першого тесту було обрано наступні параметри потоку: розмір пакетів становила 1460 байт, а міжпакетна затримка – 1,25 мс. В результаті тесту отримано інформаційний потік із інтенсивністю 800 пак/с. Інтенсивність інформаційного потоку у процесі моделювання змінювалася, як показано кривою на Рис. 5.13.

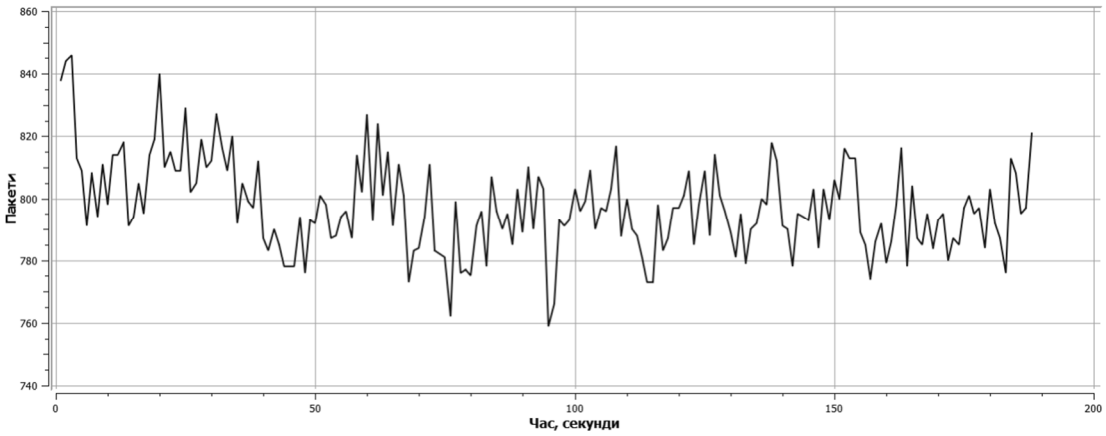


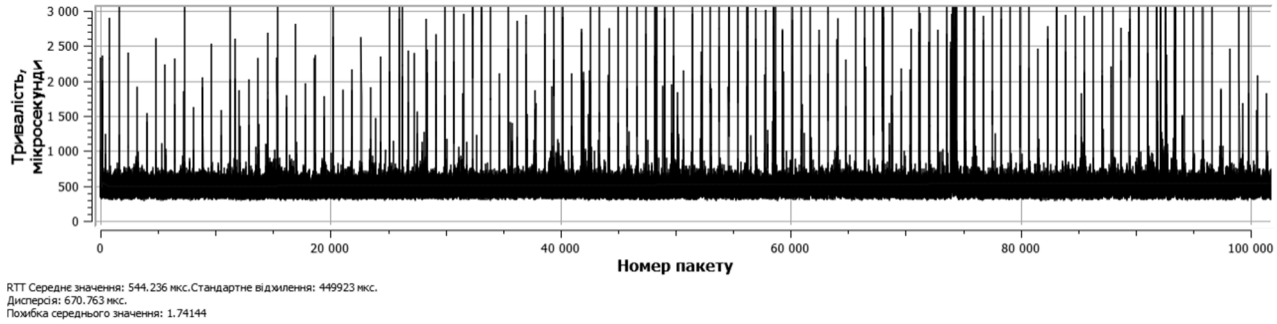
Рис. 5.13. Характеристика інтенсивності одержаного інформаційного потоку

На Рис. 5.13 видно, що інтенсивність згенерованого потоку не є стабільною, але середнє значення все ж таки становить 800 пакетів за секунду (характеризується розмахом значень, що становить близько 40 пакетів). Внаслідок багатопотоковості та особливостей даної операційної системи виникає ця нестабільність.

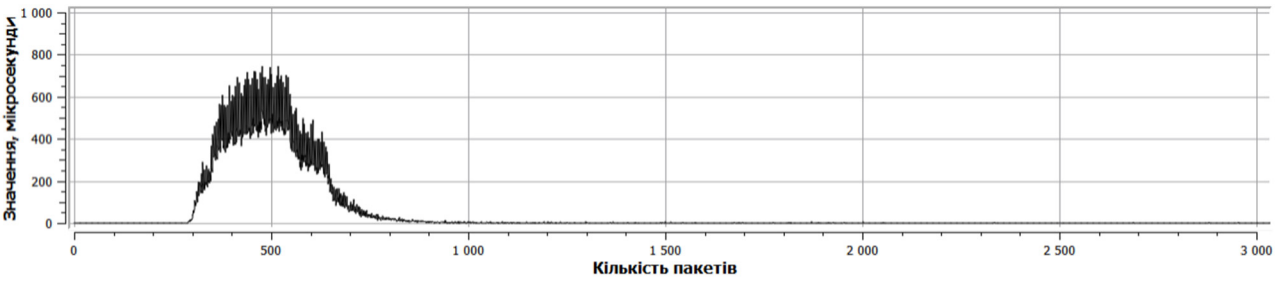
Рис. 5.14 подає графіки, які відображають ряд значень часу петлі, через яку проходять усі пакети потоку та частота розподілу цих значень за кількістю.

Результатом аналізу отриманих графіків є те, що затримка передавання від генератора і до генератора отримана 544 мкс, а розмах значень складає менше

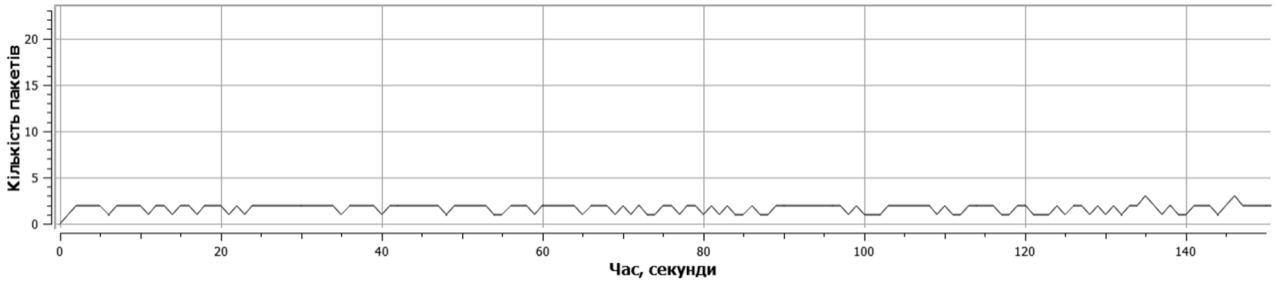
400 мкс. Отже, маршрутизатор працював у нормальному режимі, без перевантаження, тобто у ньому не було втрат та не відбувалося формування.



(а)



(б)



в)

Рис. 5.14. Затримка пакетів згенерованого потоку: (а) –значень тривалості двосторонньої затримки, (б) – гістограма статистичного розподілу, (в) – міра завантаженості буфера програмного компонента ІКС

Нижче подано результати проведення наступних тестів (Таблиця 5.2). Виконуючи аналіз отриманих результатів, бачимо, що реалізації алгоритму FIFO забезпечила максимальну продуктивність програмного компонента з урахуванням його платформи на рівні 1925 пак/с.

Таблиця 5.2. Визначні параметри функціонування програмного компонента ІКС [40]

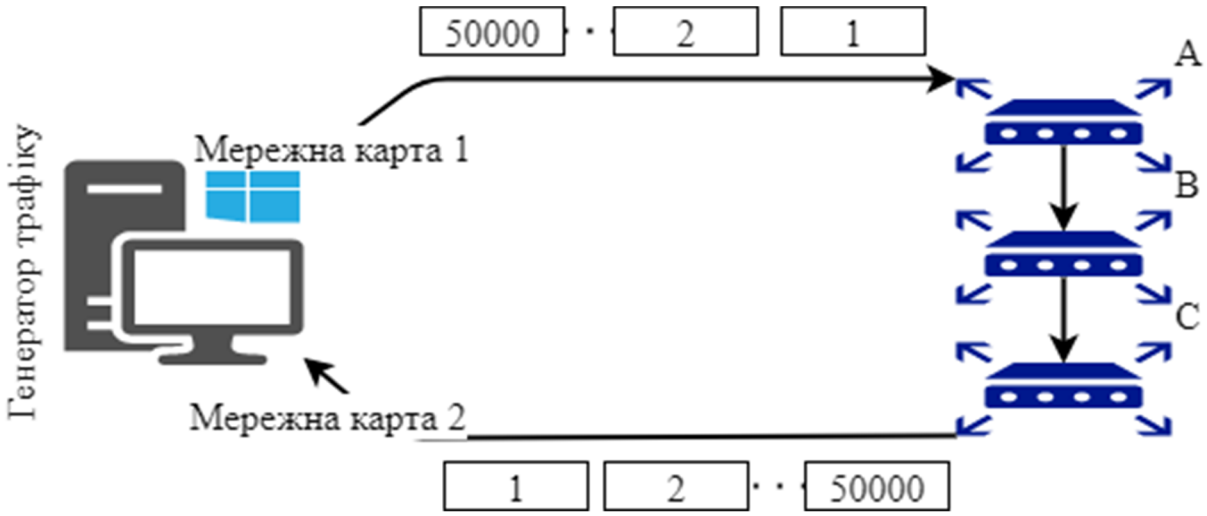
Усереднена швидкість потоку, пак/с	Подвійна наскрізна затримка, мс	Втрати пакетів, %	Число пакетів у черзі програмного компонента
800	0,544	0	1
1000	0,568	0	2
1175	0,560	0	2
1400	0,530	0	5
1750	0,745	0	8
1925	1,273	0	9
1975	5,370	7	15
2375	11,897	15	24

5.2.3. Оцінка адекватності функціонування програмованих компонентів інформаційно-телекомунікаційних систем

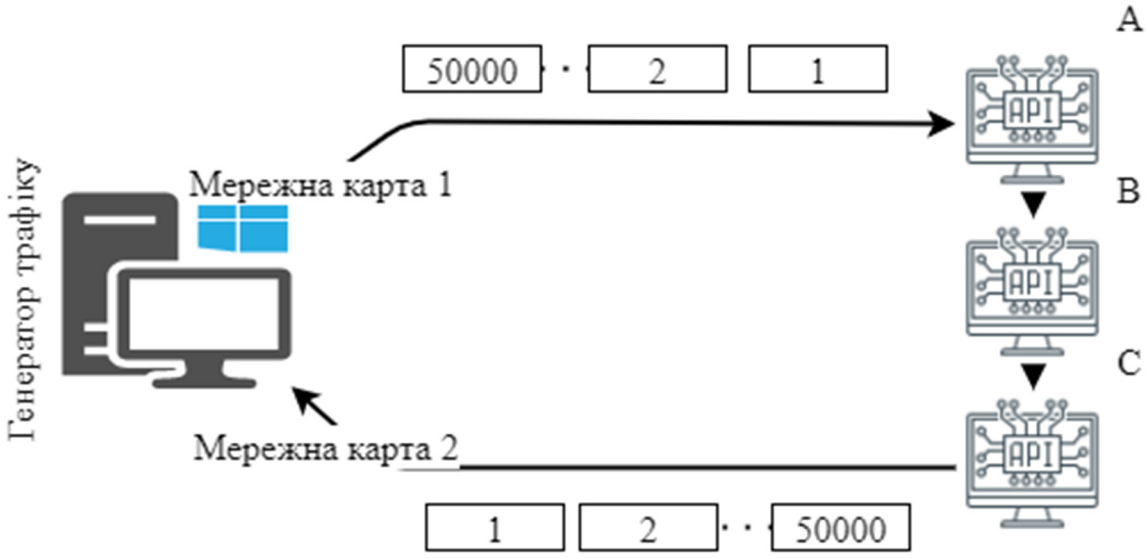
Для того, щоб підтвердити адекватність моделі, було проведено експеримент, у якому порівнювались тривалості оброблення пакетів з використанням апаратного маршрутизатора та розробленого програмного компонента ІКС [40].

Впродовж 30 хвилини проведено спостереження за інтенсивністю вхідного трафіку на інтерфейсі маршрутизатора (схема експерименту подана на Рис. 5.15). Було побудовано графіки в режимі реального часу, які відображають ряд значень тривалості обігу петлі, яку проходять усі пакети потоку та густину розподілу цих значень. Водночас мережевим аналізатором Wireshark записувалась фрагментація, групування пакетів інформаційного потоку, який був переданий та підтвердження доставки пакетів. Під час фрагментації було зафіксовано значення часової затримки та групування пакетів. Використовуючи програму Wireshark, можна було спостерігати за ходом обміну даними на мережному рівні з фільтруванням пакетів, які стосуються службових завдань операційної системи і можуть спотворити оцінку часових параметрів якості надання послуг [132].

Для того, щоб оцінити значення часової затримки пакетів, яку було внесено апаратним маршрутизатором, було проведено експеримент, під час якого інформаційний потік проходив через маршрутизатори А та А-В-С.



а



б

Рис. 5.15. Експерименти для порівняння часу опрацювання пакетів:
а) фізичний маршрутизатор, б) програмний компонент ІТС

Враховуючи результати експериментальних досліджень, які було отримано, підтверджено адекватність апаратного маршрутизатора, одержаних у ході імітаційного моделювання програмного компонента ІКС [40].

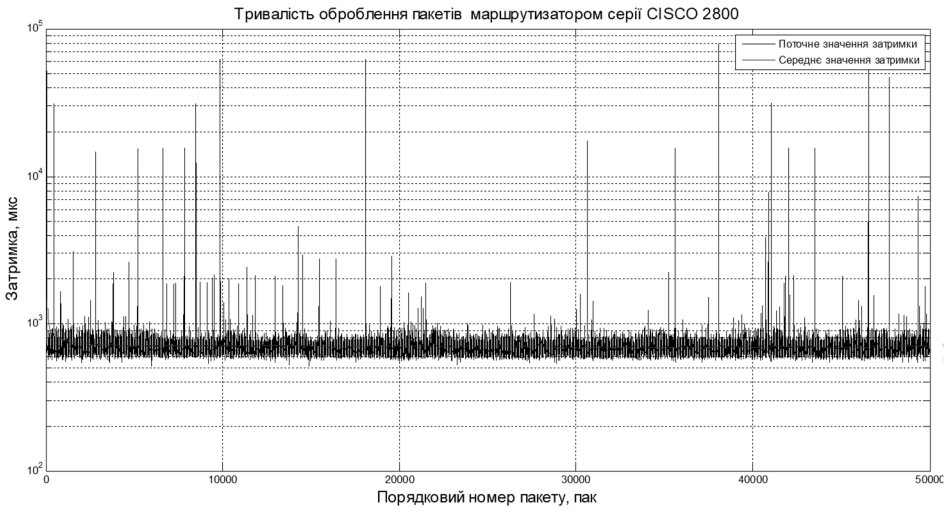


Рис. 5.16. Тривалість затримки пакетів у апаратному маршрутизаторі

Під час оброблення одного й того ж інформаційного потоку, у апаратному маршрутизаторі А, середнє значення тривалості затримки пакетів становить 695.91 мкс, а у програмному – 693.47 мкс (Рис. 5.18). Часова похибка обслуговування пакетів програмним компонентом дорівнює 2.44 мкс. Як бачимо, розподілів затримок подібні між собою достатньо, щоб говорити про достовірність і обрунтованість результату (Рис. 5.19).

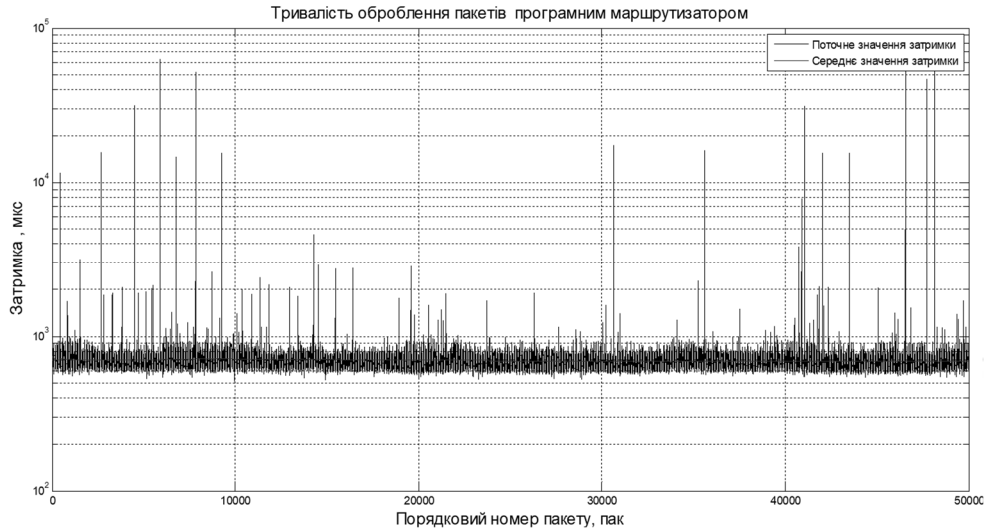


Рис. 5.17. Тривалість затримки пакетів у програмному компоненті ІКС

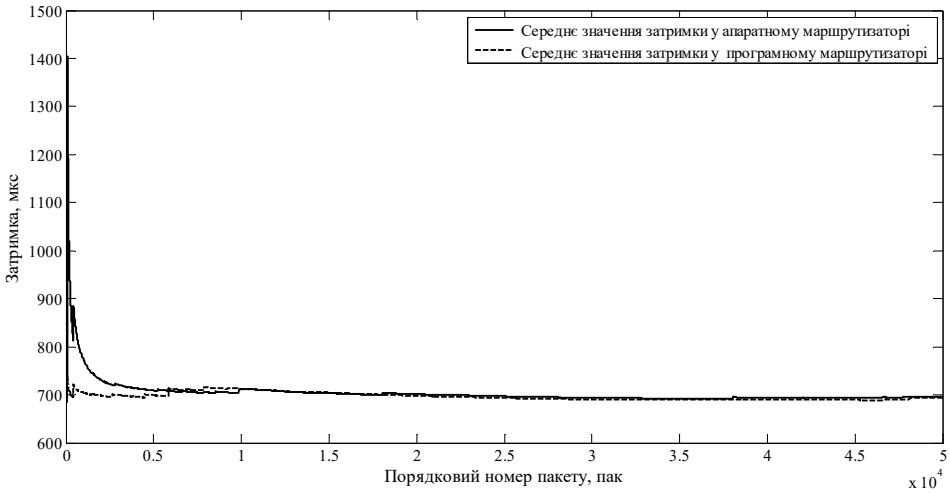


Рис. 5.18. Оцінка середніх значень тривалості оброблення пакетів програмним компонентом ІКС та фізичним вузлом

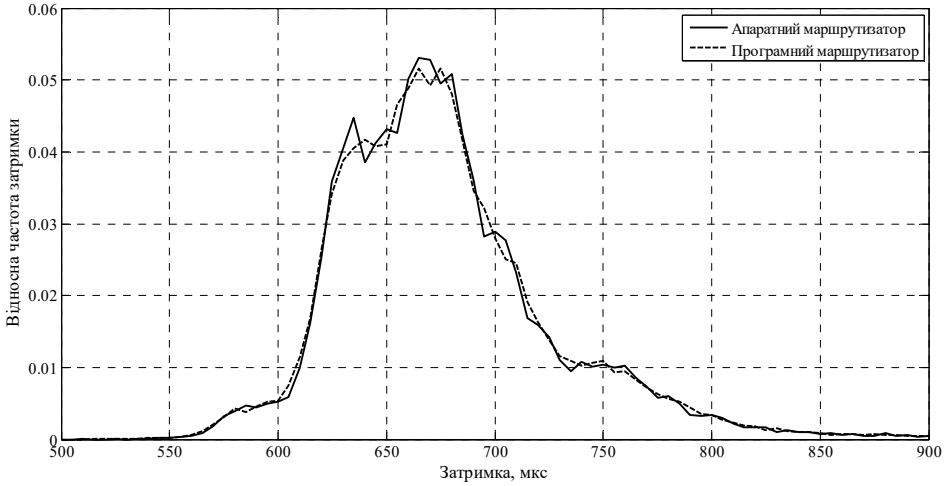


Рис. 5.19. Оцінка густини розподілу тривалості оброблення пакетів апаратним маршрутизатором та програмним компонентом ІКС

Для того, щоб оцінити тривалість оброблення пакетів реальним маршрутизатором та визначити часову затримку (з кінця в кінець) було проведено експеримент, під час якого використовувалось маршрутизатора, через які проходив згенерований інформаційний потік. У режим функціонування через консоль було сконфігуровано маршрутизатори, з допомогою алгоритму обслуговування черг FIFO. Після того, як налаштували мережу, було проводено протягом 30 хвилин спостереження по тривалості оброблення пакетів.

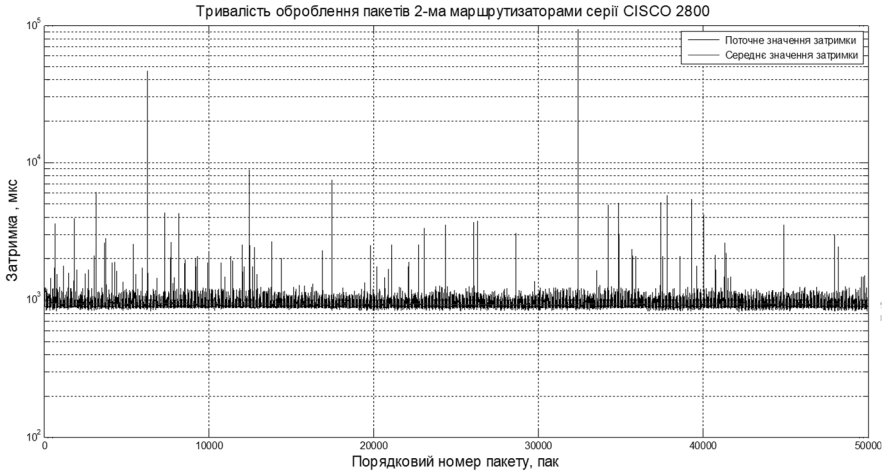


Рис. 5.20. Затримка пакетів 3-ма апаратними маршрутизаторами

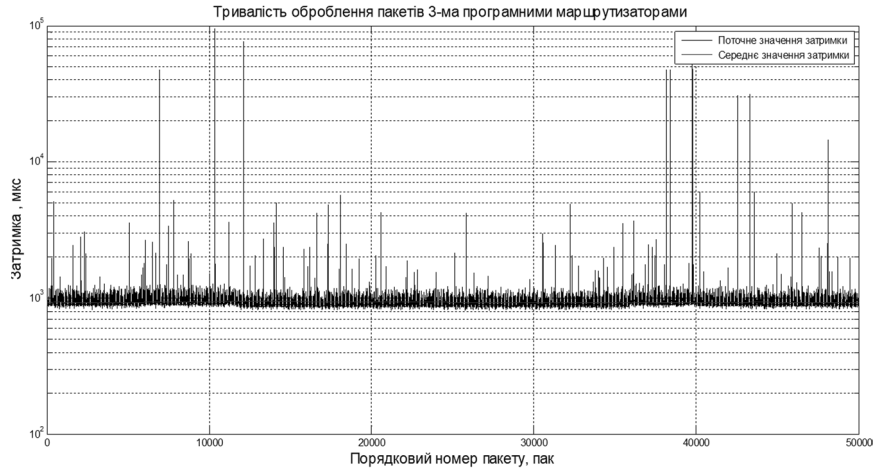


Рис. 5.21. Затримка пакетів 3-ма програмними компонентами ІКС

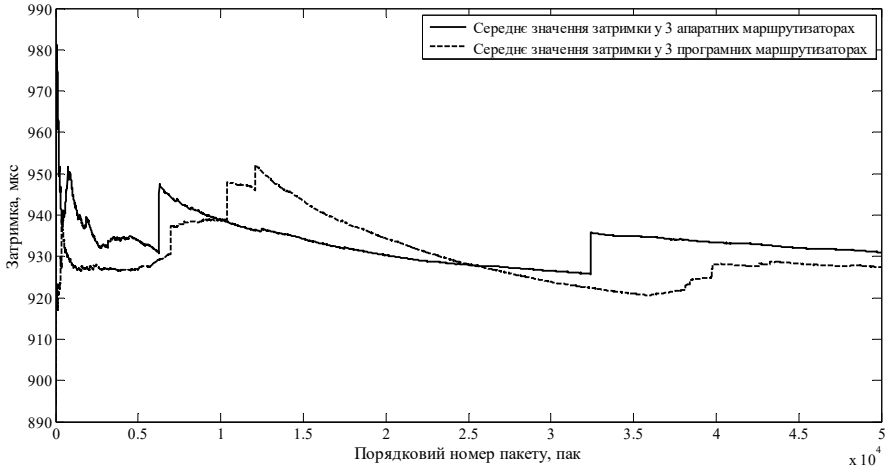


Рис. 5.22. Оцінка середніх значень тривалості оброблення пакетів 3-ма програмними компонентами ІКС та апаратними маршрутизаторами

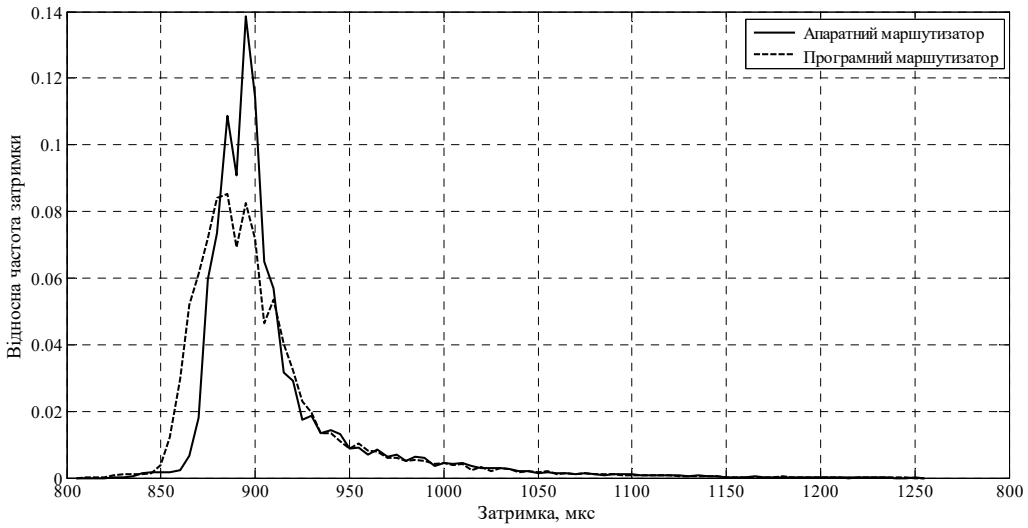


Рис. 5.23. Оцінка густини розподілу тривалості оброблення пакетів 3-ма апаратними маршрутизаторами та програмними компонентами ІКС

За отриманими оцінками, середнє значення тривалості затримки пакетів апаратними маршрутизаторами становить 930,97 мкс, а програмними компонентами ІКС – 927,4 мкс. Абсолютна похибка – 3,57 мкс. Отримані результати показують, що густина розподілу тривалості оброблення пакетів програмними компонентами ІКС наближається до густини розподілу тривалості оброблення пакетів апаратними маршрутизаторами. Оцінимо середню тривалість затримки пакетів апаратними маршрутизаторами

$$T_{ср} = \frac{T_{(A-B-C)_3} - T_{(A)_1}}{2}. \quad (5.1)$$

Підставивши отримані значення у (5.1), отримаємо $T_{ср}$ = (930,97-695,91)/2=117,53 мкс.

5.3. Моделі надання послуг в розподілених інформаційно-комунікаційних системах з урахуванням природи інформаційних потоків

5.3.1. Імітаційна модель розгортання сервісу у cloud інфраструктурі

До компонентів досліджуваної системи (Рис. 5.24) віднесемо такі [38]:

- менеджер віртуальної інфраструктури (VIM), що реалізує керування ІТС [234, 235];

- менеджер віртуальних системних функцій VNF, який реалізує підтрмку системних функцій, що впливають на стан компонентів системи. Менеджер VNF взаємодіє з оркестратором для отримання ресурсів для роззгортання програмного компоненту ІТС [68];
- підсистема виявлення ресурсів – забезпечує актальну інформацію про наявні системні ресурси. Ця підсистемп забезпечує постійне оцінювання всіх типів ресурсів системи (фізичних, віртуальних, комунікаційних). Отримані дані щодо доступної пропускної здатності каналів та доступності складових частин сервісує основою надання сервісу чи перенесення його складових частин на альтернативний сервер. Запити від цієї підсистеми надсилаються до оркестратора щодо перенесення віртуальних машин у цілях балансування навантаження та ефективного використання системних ресурсів [89];
- менеджер міграції є тією сутністю, яка відповідає за міграцію віртуальних машин і передає інформацію підсистемі керування віртуальною мережею щодо зміни способу доступу до перенесеної віртуальної машини;
- менеджер віртуальної комунікаційної мережі – організує функціонування комутаторів VDE, що забезпечують групування віртуальних машин у віртуальні локальні мережі, що спрощує їх обслуговування.
- віртуальний системний комутатор за призначенням є аналогом Ethernet-комутатора. Утворені хмарні сервіси, які асоційовані зі спільним VDE, доступні один одному так, ніби вони перебувають у реальній мережі. Їм властива висока швидкість переконфігурації мережі без негативного втручання у вхідну чергу запитів та незалежно від необхідної кількості портів. Це важливо за наскрізного передавання трафіку. Керування потоками здійснюється балансуванням кількості пакетів між віртуальними буферами вхідних/вихідних портів системи.
- оркестратор відповідає за організацію взамодії у віртуалізованій ІТС і постачає мережні послуги на віртуалізованій інфраструктурі. Цей елемент системи вміщує кореневий інтерфейс, завданням якого є контроль над

рорзгортанням та наданням сервісів; опрацьовує дані сервісів моніторингу та конфігурування; на основі даних від підсистеми виявлення ресурсів [237] організовує міграцію та повідомляє менеджеру міграції про переміщення віртуальних машин, а менеджеру віртуальної комунікаційної мережі надає інформацію про новий спосіб доступу до перенесених ресурсів. Він же здійснює контроль над дотриманням пріоритезації сервісів [236].

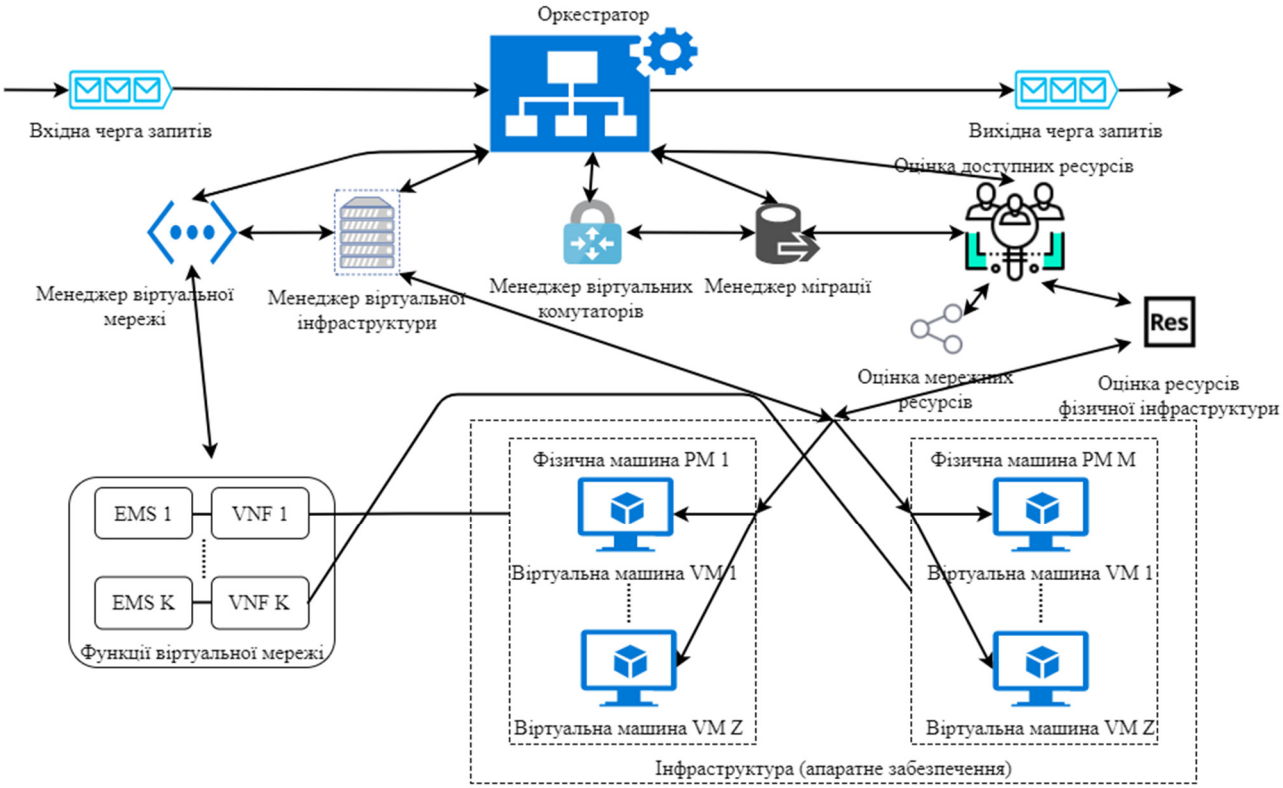


Рис. 5.24. Модель керування розгортанням інформаційної послуги в ІТС

Проаналізуємо процес обслуговування запиту на надання послуги у описаній вище системі. Нехай експоненційний закон визначає розподіл інтенсивності поступлення цих запитів. В даному випадку мова йде про дискретну випадкову величину. Процес розпочинається із надходження запиту до черги вхідного інтерфейсу. Оркестратор викликає підсистему виявлення ресурсів, щоб встановити, чи всі необхідні компоненти досліджуваної послуги доступні. Фактично, здійснюється перевірка наявності та стану обчислювальних ресурсів на рівні фізичних, а також віртуальних компонентів, щоб гарантувати доступність всіх складових частин сервісу. Підсистема виявлення ресурсів

працює паралельно, тобто відбувається перевірка доступності вільних фізичних та мережних ресурсів, визначається оптимальних шлях для передавання компонентів по шляхах із найменшим завантаженням. Коли одна із перевірок дає негативний результат, то згадані підсистеми сповіщають інші компоненти про недостатні обчислювальні чи мережні ресурси. Після цього підсистема виявлення ресурсів сповіщає оркестратор та надає інформацію про незайняті ресурси менеджеру міграції, який, у свою чергу, вибирає фізичний сервер та здійснює міграцію недоступного елемента сервісу (перенесення віртуальної машини), для чого залучає менеджер віртуальної комунікаційної мережі. На підставі цих даних менеджер віртуальних комутаторів приймає рішення, в якій із віртуальних локальних мереж розташовувалася перевантажена запитами віртуальна машина і у якій віртуальній локальній мережі знаходиться недовантажена віртуальна машина, і за допомогою віртуального комутатора здійснює переміщення логічної частини елемента сервісу та реалізує перегляд конфігурації ІТС з урахуванням впроваджених змін. Процес міграції і функціонування ІТС здійснюється із залученням оркестратора і менеджера віртуальних інтерфейсів. За умови, що число запитів на на деякий елемент сервісу постійно зростатиме, менеджер віртуальної інфраструктури здійснює використання віртуалізації мережних функцій. Цей спосіб дає змогу реалізувати довільний програмний компонент, який функціонально відповідає деякому фізичному мережному пристроєві та ініціювати на ньому оброблення запитів на обслуговування.

Пісдсистема виявлення ресурсів здійснює моніторинг стану системних ресурсів. Вона забезпечує оцінку доступних ресурсів різного призначення у системі. Така оцінка у роботі базується на максимальній ефективності використання ресурсів фізичних серверів. Із цією метою система постійно перевіряє присутність вільних ресурсів та їх доступність через телекомунікаційну мережу [171, 173, 238].

Вільні обчислювальні ресурси деякої віртуальної машини, що реалізує розміщення M_i елементу сервісу оцінимо як

$$CPU_{pr} = \frac{\sum_{i=1}^{k} M_i \times CPU_i}{\sum_{i=1}^{k} M_i}, \tag{5.2}$$

$$RAM_{pr} = \frac{\sum_{i=1}^{k} M_i \times RAM_i}{\sum_{i=1}^{k} M_i}, \tag{5.3}$$

де M_i – кількість елементів сервісу у i -тій віртуальній машині; CPU_i – зайнятість процесора віртуальної машини, спричинена i-тим елементом сервісу; RAM_i – зайнятість оперативної пам'яті віртуальної машини, яка припадає на i-тий елемент сервісу; k – кількість віртуальних машин на одному фізичному сервері ІТС.

Алгоритм, призначений для оцінювання зайнятості ресурсів, представлено на Рис. 5.26. Він забезпечує пришвидшене оброблення запитів та проводить аналіз завантаженості кожної віртуальної машини, базуючись на інформації про обчислювальні ресурси кожної віртуальної машини.

Позначимо через $z=\overline{1,Z}$ деяку віртуальну машину, яка розміщує $i=\overline{1,K}$ елементів сервісу $j=\overline{1,S}$, а через $m=\overline{1,M}$ позначимо фізичний сервер, який розміщує $Z_m=\overline{1,Z}$ віртуальних машин; $P=\{P_1,\ldots,P_n,\ldots P_N\}, n=\overline{1,N}, P_n=\{e_1,\ldots e_l,\ldots e_L\}, l=\overline{1,L}$ – множини шляхів у структурі ІТС, де N – число шляхів між елементами i та $i+1$ сервісу, L – число каналів шляху n, що поєднують віртуальні машини VM_i та VM_{i+1}.

Оброблення розпочинається із перевірки вхідної черги запитів на використання елементів сервісу. Коли там розташовано $f=\overline{1,F}$ запитів на деякий елемент сервісу j, то ініціюється запуск аналізатора, який оцінює необхідні ресурси для обслуговування цього елементу сервісу (дані про необхідні ресурси процесора, пам'яті та пропускну здатність мережі заносяться в таблицю), так, щоб гарантувати доступ до елемента за час $t<t_{кр}$ та належний тип віртуальної машини для цього елемента сервісу. Також здійснюється

перевірка доступності елемента i послуги j. Довільний елемент послуги потребує для реалізації своїх функцій апаратних ресурсів фізичного сервера та віртуальної машини, на яких його розміщено. Підсистема виявлення ресурсів перевіряє наявність достатніх апаратних ресурсів z -тої віртуальної машини для надання користувачеві ще одого екземпляру елемента i. У випадку достатності ресурсів, слід перевірити, чи вистачить пропускної здатності каналу для встановлення зв'язку елемента з кінцевим користувачем

$$C_{P_n} > R_{i,j}, \tag{5.4}$$

де $R_{i,j}$ - інтенивність потоку даних, джерелом якого є *i*-тий елемент послуги *j*. Визначені у момент перевірки стану мережі доступні ресурси будуть записані підсистемою виявлення у масив $\overline{A}$, який визначатиме чергу на обслуговування запитів. Елементами масиву є достатня для надання елемента сервісу пропускна здатність каналу. Підсистема виявлення повідомляє цю інформацію оркестратору, який переносить елемент у вихідну чергу запитів Q_2.

Підсистема виявлення здійснює перевірку зайнятості пропускної здатності каналів та надає оптимальний шлях для комунікаціх компонентів по мінімально навантажених каналах. Визначимо частку пропускної здатності каналу, яка припадає на кожен елемент сервісу у процесі його надання, відносно загального числа сервісів, які передаються у даному каналі.

Ваговий коефіцієнт інтенсивності потоку даних *i*-ого елементу *j*-ого сервісу відносно загальної інтенсивності трафіку всіх сервісів, які передаються у *l*-ому каналі зв'язку:

$$k_{i,j|l} = \frac{R_{i,j|l}}{\sum\limits_{i,j} R_{i,j|l}}. \tag{5.5}$$

Таким чином, частка вільної пропускної здатності *l*-ого каналу зв'язку

$$k_{0|l} = 1 - \frac{\sum_{i,j} R_{i,j|l}}{C_l}, \tag{5.6}$$

де C_l – пропускна здатність *l*-ого каналу.

Частки використання пропускної здатності $\mathrm{Pr}_{i,j|l}$ ропорційна до $k_{i,l}$:

$$\mathrm{Pr}_{i,j|l} = k_{i,j|l} \cdot \left(1 - k_{0|l}\right) \cdot 100\% \tag{5.7}$$

Нехай $\mathrm{Pr}_{i,j|l}$ визначає пріоритет елемента щодо інших елементів, чий трафік передається у *l*-ому каналі зв'язку. Тобто, завершення надання цього елемента звільнить найбільшу пропускну здатність каналу.

Нехай деяка частка пропускної здатності каналу використовується неефективно, за умови, що $\min\left(k_{0|l} \mid \mathrm{P}_n\right)$ досягає максимуму. У такому разі, менеджери віртуальної комутації та міграції забезпечують завантаження частки вільної пропускної здатності шляхом її перерозподілу між елементами із більшою потребою.

Якщо вільних ресурсів для надання ще одного екземпляру елемента чи наступного елемента не вистачає, тобто справедливі умови-обмеження

$$CPU_{available}(z) > CPU(i,j) \tag{5.8}$$

$$RAM_{available}(z) > RAM(i,j) \tag{5.9}$$

$$C_{P_n} \min\left(k_{0|l} \mid \mathrm{P}_n\right) > R_{i,j} \mid P_n, \tag{5.10}$$

то відбувається копіювання *z*-ої віртуальної машини на інший фізичний сервер $\mathrm{PM_m}$. Запит на міграцію елемента передається оркестратору та менеджеру міграції, який, у свою чергу, забезпечує переміщення перевантаженої віртуальної машини $\mathrm{VM_z}$ у тісній комунікації із менеджером віртуальної комутації. У випадку неможливості міграції запит отримає відмову в обслуговуванні, а підсистема виявлення ресурсів знову почне аналіз ресурсів.

Оркестратору буде відомий максимум вільних віртуальних і телекомунікаційних ресурсів, тобто оркестратор може котролювати міграцію елементів сервісу на менш завантажені фізичні сервери.

5.3.2. Дослідження поведінки інформаційно-комунікаційної cloud системи у процесі розгортання сервісу

Дослідження ефективності керування розгортанням сервісів здійснено із використанням розробленої імітаційної моделі ІТС. Дискретні події є основою функціонування переважної частини моделей ІТС, однак відзначимо, що це не завжди вдало відображає процеси, які відбуваються в системі [92]. Основою системи розгортання є модель розміщення віртуальних машин на ресурсах, які надані фізичними серверами та постачання сервісу із контролем вільних доступних ресурсів. Функціональна модель на мові UML подана на Рис. 5.25 [106, 107].

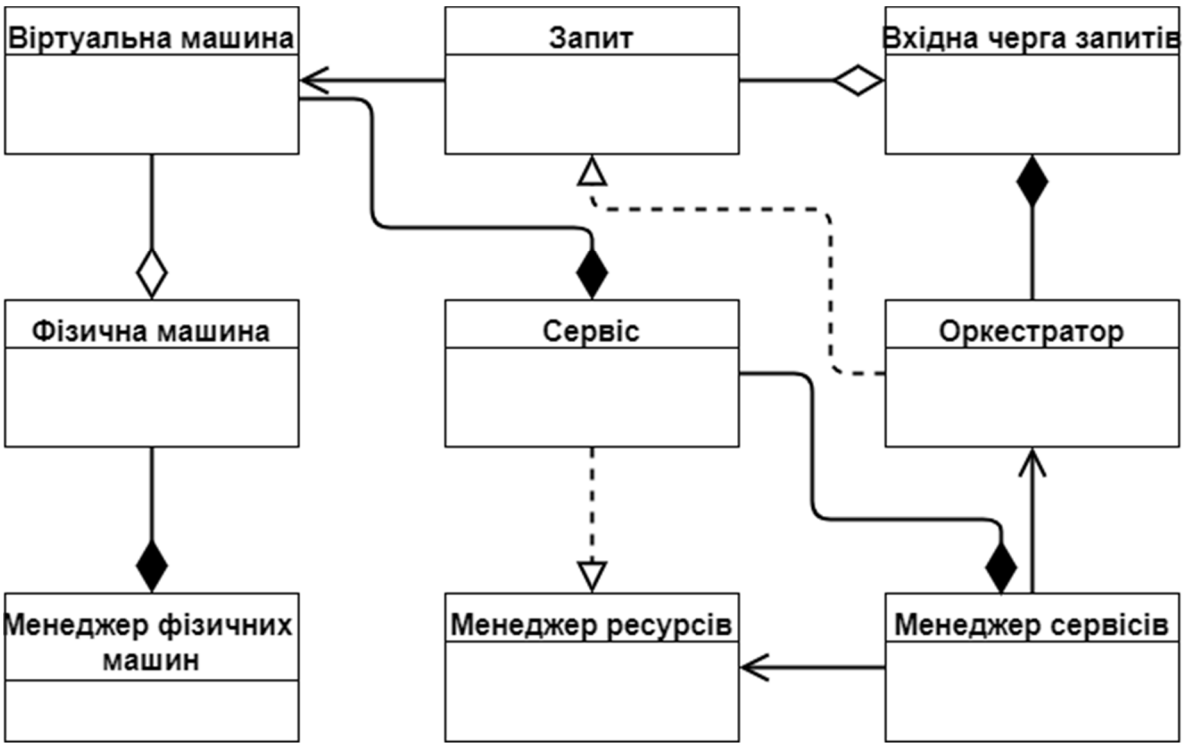


Рис. 5.25. UML модель системи керування розгортанням сервісів у ІТС

Протокольна діаграма процесу надання послуг користувачеві із розгортанням cloud інфраструктури на основі розробленої системи керування наданням послуг представлена на Рис. 5.27.

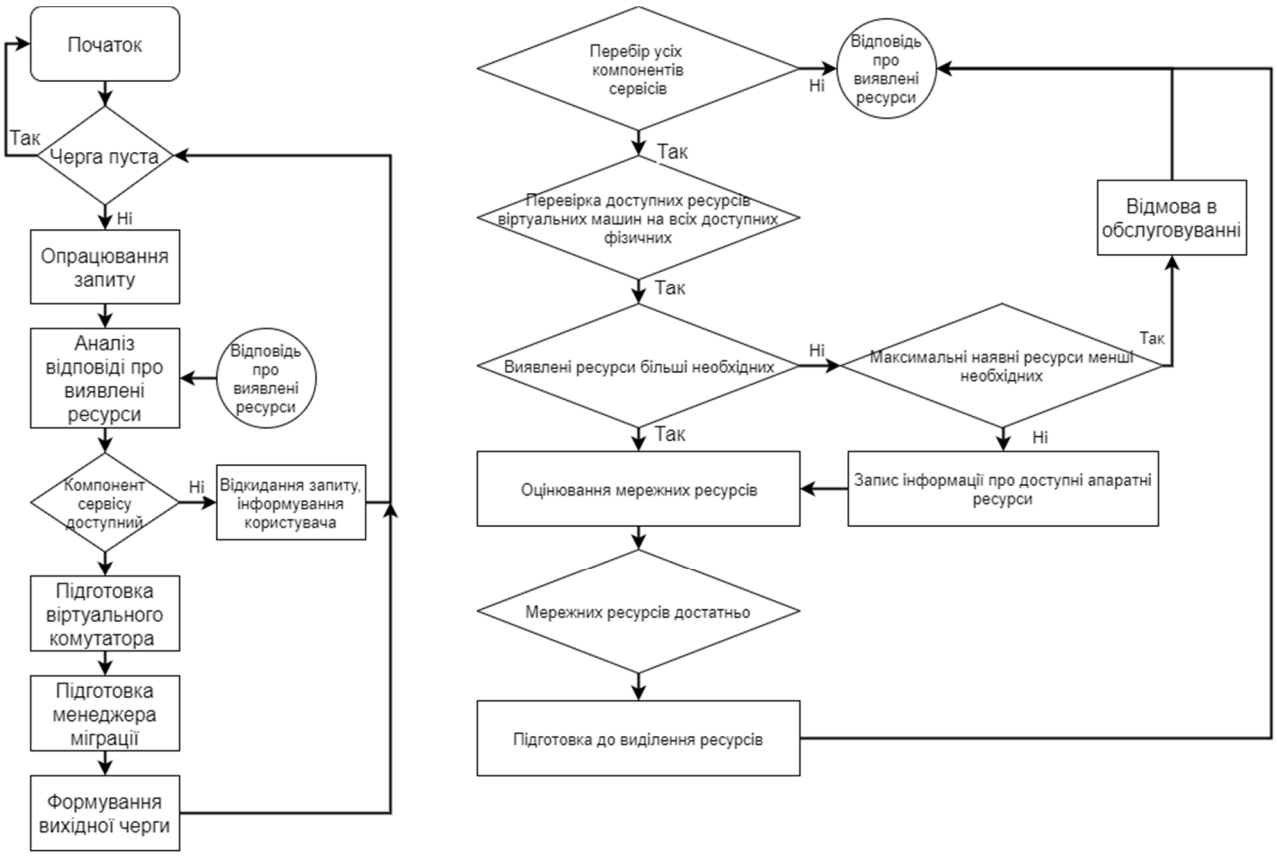


Рис. 5.26. Алгоритм оцінювання вільних і доступних ресурсів

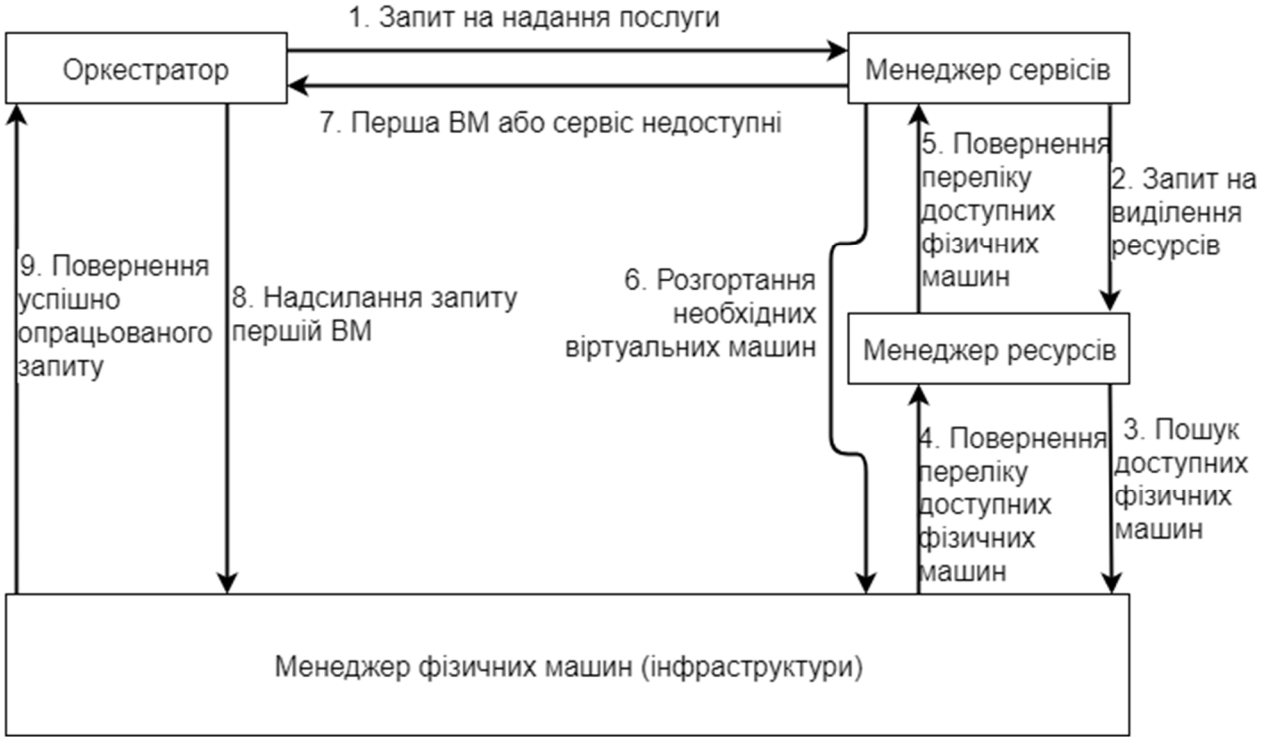


Рис. 5.27. Протокольна діаграма процесу надання сервісу

5.3.3. Покращення значень метрик якості обслуговування на основі вдосконаленого методу оцінки доступних ресурсів у процесі розгортання інформаційно-комунікаційного сервісу

Розроблено програмне забезпечення на мові програмування C++ у середовищі Qt5.4, основою якого є представлена у роботі імітаційна модель. Cloud інфраструктура ІТС довільної конфігурації задається на основі параметрів: число фізичних серверів, обсяг апаратних ресурсів для розгортання кожного елемента сервісу, число послуг та їх тип.

По замовчуванню інфраструктура має типову конфігурацію, хоча її елементи можуть бути незалежно сконфігуровані. З'єднання між елементами інфраструктури реалізовані на базі випадково згенерованої матриці суміжності. Інтерфейс конфігурування ІТС та керування процесом розгортання представлено на Рис. 5.28.

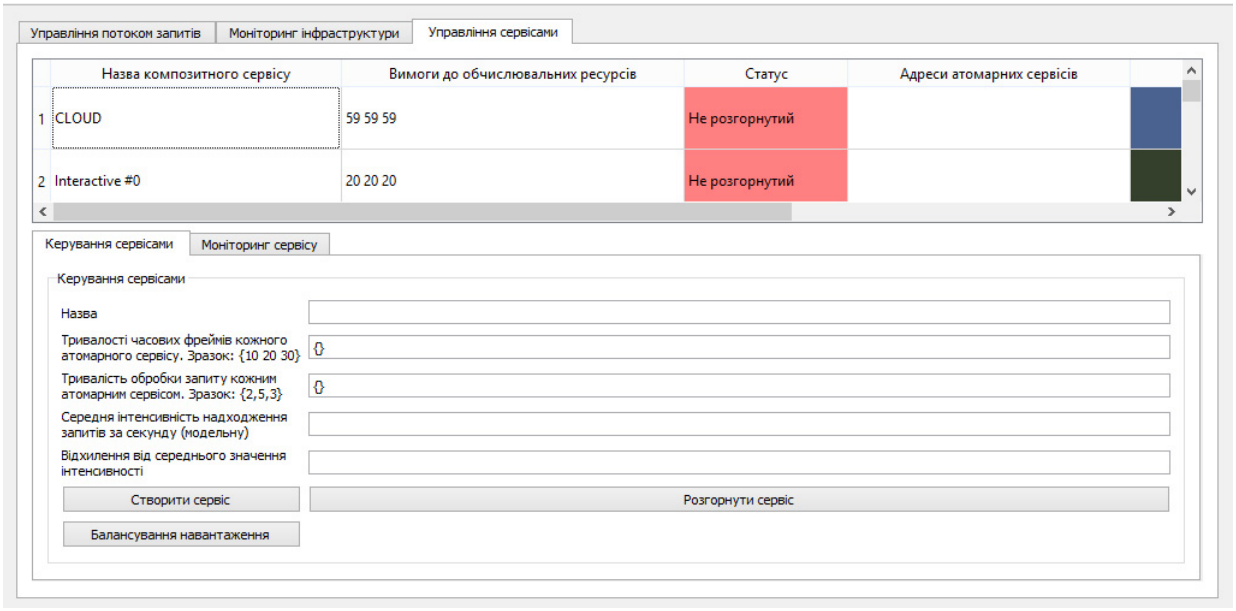


Рис. 5.28. Інтерфейс конфігурування моделі ІТС

У моделі утворено множину послуг, які підлягають розгортанню у змодельованій системній інфраструктурі. Ця інфраструктура задається матрицею: по горизонталі –фізичні сервери, по вертикалі – віртуальні машини, які були розгорнуті на кожному сервері. Параметри розгорнутих послуг

згруповано у вигляді таблиці (Таблиця 5.3), а результат їх розгортання у інфраструктурі подано на Рис. 5.29.

Таблиця 5.3. Параметри сервісів ІТС [38]

Номер з/п	Вектор-стовпець необхідних апаратних ресурсів	Розміщення елементарних сервісів
1	{59, 59, 59}	Розташування 1 {1001, 2001, 3001}
2	{20, 20, 20}	Розташування 2 {1002, 1003, 2002}
3	{20, 20, 20}	Розташування 3 {2003, 3002, 3003}
4	{20, 20, 20}	Розташування 4 {4001, 4002, 4003}
5	{20, 20, 20}	Розташування 5 {4004, 5001, 5002}
6	{20, 20, 20}	Розташування 6 {5003, 5004, 6001}

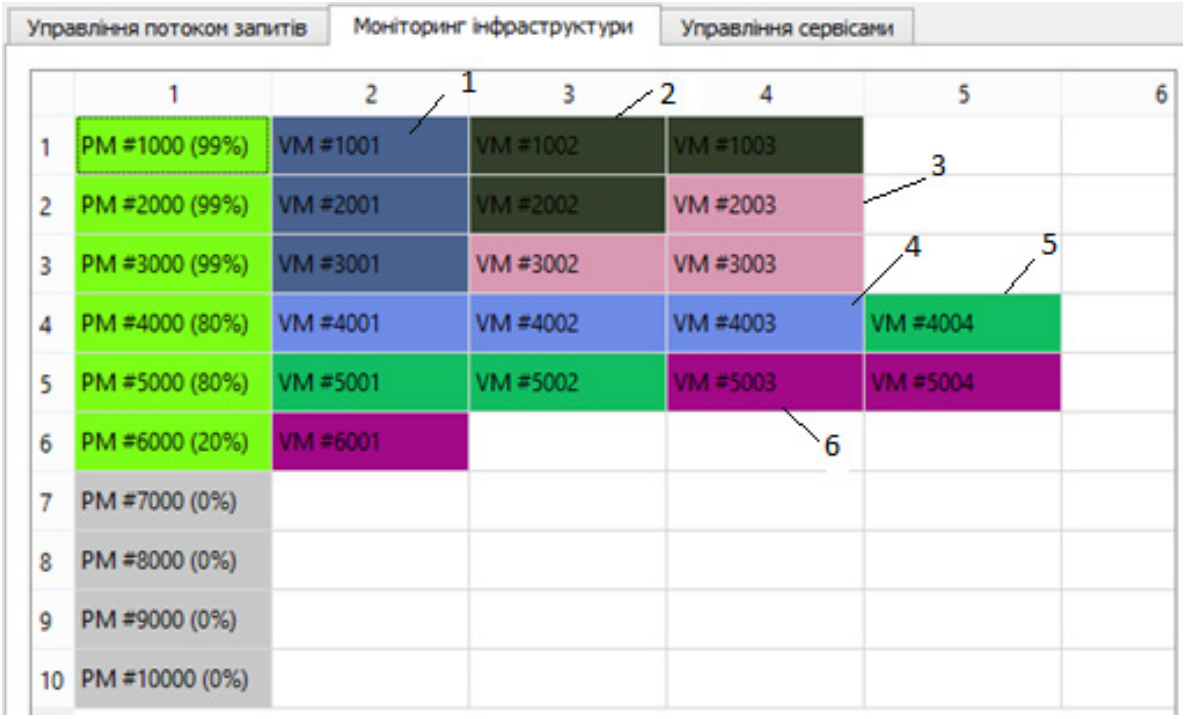


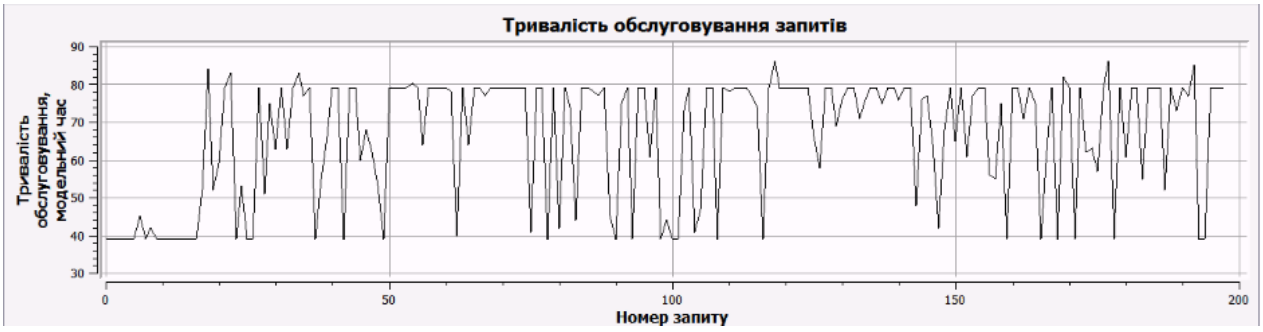
Рис. 5.29. Результат розгортання сервісів у ІТС

Трафік запитів, який надходить на систему, згенеровано із застосуванням логнормального (інтервал між надходженням запитів) та експоненційного (інтенсивність надходження запитів) законів розподілу. У результаті отримано мультисервісний трафік, характеристики якого близькі до реальних.

Робота моделі розпочинається із одночасної активації усіх сервісів і генераторів трафіку кожного типу послуги. Віртуальні машини існують як завгодно довго. Виділено три стадії процесу моделювання:

– перша стадія – аналіз процесу функціонування системи зі відомою архітектурою. Відбувається дослідження завантаженості фізичних ресурсів серверів, порівнюються значення наскрізної затримки передавання пакетів, кількість опрацьованих та відкинутих запитів. Особливу увагу відводено сервісу, для якого спостерігається максимальне число відкинутих запитів.
– друга стадія – постійний моніторинг параметрів системи та параметрів обслуговування запитів на сервіси. Це здійснюється на основі розробленої моделі керування розгортанням сервісів у cloud інфраструктурі ІТС.
– третя стадія – застосування алгоритму балансування навантаження для тих послуг, тривалість обслуговування запитів яких є максимальною. Узгоджене застосування алгоритму балансування навантаження та інтегрованого керування розгортанням запитів вдалося знизити затримку передавання запитів (буде показано нижче), що сприяє підвищенню продуктивності мережі та якості надання послуг.

Процес моделювання здійсено з та без застосування алгоритму балансування та моделі керування розгортанням сервісів. Рис. 5.30 а, б, в, г, д, е представляє аналіз тривалості обслуговування запитів на надання всіх типів сервісів у процесі моделювання. У точці переходу на роботу системи із застосуванням розроблених алгоритму та моделі відзначаємо зниження затримки та продовжуємо здійснювати контроль за ресурсами кожного фізичного серверу.



а)


Тривалість обслуговування запитів
Тривалість обслуговування, модельний час
Номер запиту

б)


Тривалість обслуговування запитів
Тривалість обслуговування, модельний час
Номер запиту

в)


Тривалість обслуговування запитів
Тривалість обслуговування, модельний час
Номер запиту

г)


Тривалість обслуговування запитів
Тривалість обслуговування, модельний час
Номер запиту

д)

е)

Рис. 5.30. Аналіз часу обслуговування запитів на надання деякого типу сервісу

Таблиця 5.4. Результати розгортання сервісів та дослідження їх роботи [38]

Номер послуги з/п	Розміщення елементарного сервісу	Час сесії, мод. с	Число запитів на сервіс	Число успішних запитів	Час сесії із застосуванням методу оцінювання доступних ресурсів, мод. с	Число запитів на сервіс із застосуванням запропонованого методу	Число успішних запитів
1	Розташування 1 {1001, 2001, 3001}	70	212	212	70	407	407
2	Розташування 2 {1002, 1003, 2002}	150	394	391	**50**	739	739
3	Розташування 3 {2003, 3002, 3003}	130	367	365	130	711	706
4	Розташування 4 {4001, 4002, 4003}	80	405	404	80	695	692
5	Розташування 5 {4004, 5001, 5002}	90	374	372	90	732	728
6	Розташування 6 {5003, 5004, 6001}	60	332	332	60	740	740

Результат, подані на Рис. 5.30 свідчать, що найбільш популярним був другий сервіс, а середня тривалість обслуговування запитів на цей сервіс становила 150 секунд модельного часу. Масштаб модельного і реального часу однаковий.

Бачимо, що ресурсів для надання ще одного елемента другого сервісу замало. Буфери пам’яті віртуальних машин переповнені запитами. Тому оркестратор вирішує мігрувати елементи сервісу на інший фізичний сервер.

Рис. 5.31 відображає стан інфраструктури ІТС після завершення міграції віртуальних машин та застосування алгоритму балансування навантаження.

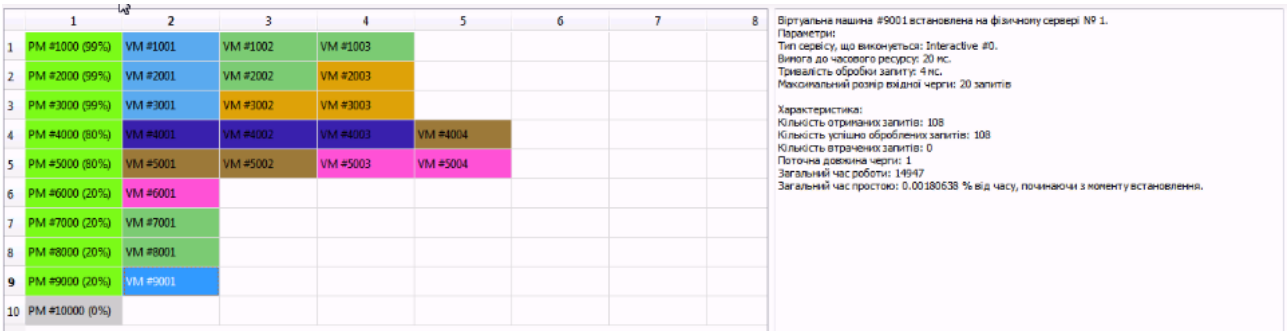


Рис. 5.31. Стан інфраструктури ІТС після міграції та застосування алгоритму балансування навантаження

Аналіз якості надання послуг (Рис. 5.32) свідчить, що перенесення елементів сервісу дало змогу знизити середню тривалість оброблення запитів та наскрізну затримку передавання пакетів з кінця в кінець з 150 до 55 мкс (майже утричі).

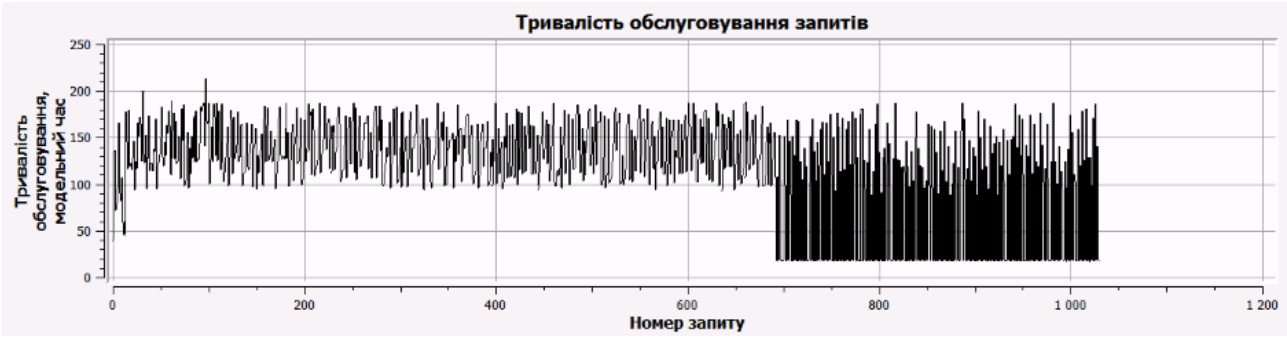


Рис. 5.32. Час обслуговування запитів на надання послуги другого типу до і після застосування алгоритму балансування та моделі керування розгортанням сервісів

5.3.4. Влив балансування сервісних потоків на тривалість очікування на надання послуг

Низька вартість трансляції через Інтернет та доступність Інтернет-трансляції у порівнянні з звичайними передавачами та обладнанням трансляцій зробили сервіс IPTV вибором багатьох постачальників в даний час.

Як правило, IPTV – це термін, який застосовується до постачання традиційних телевізійних каналів, фільмів та відеозапису за запитом у приватній або загальнодоступній мережі. З точки зору кінцевого користувача, IPTV повинна виглядати і працювати так само, як стандартне обслуговування платного телебачення з додаванням нових функцій та послуг.

Як правило, IPTV-сервери реалізують потік онлайн каналів шляхом багатоадресного передавання трафіку підключеним клієнтами, щоб мінімізувати навантаження на сервери під час потокового передавання на велику кількість клієнтів одночасно, тим самим передаючи відповідальність за передачу пакетів в мережеву інфраструктуру. У випадку "Відео за запитом" (VoD), для передачі запитуваного відео необхідно встановити односпрямоване з'єднання між клієнтом та сервером. Це призводить до висновку, що завантаження сервера прямо пропорційно кількості підключених клієнтів VoD. Тому необхідний набір серверах VoD, які повинні працювати одночасно для відповіді на запити клієнтів, слід оцінювати статистично з урахуванням числа підписників, а також механізму, який забезпечує балансування розподілу запитів клієнтів на VoD-сервери та забезпечує уникнення відмов і має велике значення для IPTV [129].

В даний час існує лише кілька механізмів балансування трафіку. Проведено дослідження того, як ці механізми обробляють трафік IPTV [30].

Для того, щоб визначити відповідь системи, яка реалізує певний балансовий механізм, по-перше, необхідно розробити модель трафіку IPTV. Ця модель повинна бути простою для її програмної реалізації і близькою до реального джерела IPTV.

Моделі для механізмів балансування трафіку також необхідні для математичного аналізу відповідей цих систем, коли вхідним впливом є трафік IPTV [276].

Як правило, контент високої чіткості (HD) кодується за допомогою MPEG-4 (H.264), що має високий коефіцієнт стиснення та призводить до дуже змінних темпів передавання даних (VBR) для стиснутого відео.

Основним завданням стиснення відео є видалення просторової та часової надлишковості в межах кожного кадру та між послідовними кадрами для ефективного використання пропускної здатності. Безперервне потокове відео перетворюється на послідовність кадрів на вході кодера. Після кодування кадри передаються періодично, і формують GoP. Кожна GoP містить I фрейм і певну кількість P та B фреймів. Наприклад, з кодеком MPEG GoP набуває вигляду I $B1$ $B2$ $P1$ $B3$ $B4$ $P2$ $B5$ $B6$ $P3$ $B7$ $B8$. Перший фрейм у кожній GoP – це I фрейм, який внутрішньо закодований без посилання на будь-які інші фрейми. Наступні P фрейми як внутрішньо-закодовані, так і кодовані по відношенню до попереднього P або I фрейму. Решта B фрейми також внутрішньо закодовані та закодовані між собою, і вони використовують як попередні, так і наступні P або I фрейми як основу [124].

Для цілей дослідження використовується дворівнева марківська модель трафіку. Модель розглядає як просторову, так і часову кореляцію в послідовностях, кодованих у форматі MPEG, тому вона може імітувати сильно змінні швидкості передавання даних (VBR) джерел IPTV. Модель містить ланцюг Маркова на рівні групи зображень (GoP) та ланцюг Маркова на рівні кадрів, тому він може фіксувати як внутрігрупові, так і міжгрупові GoP-кореляції.

Розмір кадру залежить від складності текстури та рухової складності відеоконтенту або його просторових та часових доменних кореляцій.

У просторових та часових областях ми класифікуємо відео на декілька рівнів, S і T, відповідно. Таким чином, можемо оперувати $S \times T$ станами для відображення кореляції в обох областях. Оскільки тривалість GoP менша за півсекунди, ми припускаємо, що просторові та часові співвідношення джерела відеозапису в кожному GoP залишаються на тому ж рівні або в тому ж стані. Тому ми можемо побудувати дискретний ланцюг Маркова рівня GoP, в якому кожен стан представляє часові та просторові кореляції відео в межах цього GoP.

Зі збільшенням кількості станів (збільшенням S і T), модель стає більш точною, але збільшується обчислювальна складність формування та

використання моделі. З емпіричних досліджень відомо, що вибір $S=3$ і $T=3$ забезпечує прийнятний компроміс між точністю та складністю моделі. Три рівні відображають стани з низькою (L), середньою (M) та високою (H) кореляцією.

Межі між станами повинні бути встановлені належним чином, щоб обмежене число станів можна було використовувати для точного збору статистики відео. Відповідно до експериментальних результатів з кількома відеопотоками, рівномірний розподіл кадрів в кожному стані дає досить хороші результати. Тому, імовірність перебування у стані $\Pr(S_i)=1/N$, де S_i є одним із станів, і $N = S \times T$ – загальна кількість станів.

В просторовій області, з урахуванням того, що тільки I фрейми незалежно внутрішньо-закодовані, розмір I фрейму використовується для визначення текстурної складності цілої GoP. У часовій області співвідношення розмірів першого P фрейму до розміру I фрейму в цій же GoP використовується для визначення часової кореляції, що представлено нижче. Фрейм $P1$ одночасно внутрішньо і міжфреймово закодований з єдиним посиланням на I фрейм. Розмір фрейму $P1$ визначається як $P1 = P1t \times \Delta\varphi$, де $\Delta\varphi$ представляє вектор переміщення від фрейму I до фрейму $P1$ в одній GoP, а $P1t$ – інформація про текстуру, що міститься у фреймі $P1$, яка майже повністю співпадає з інформацією про текстуру, закладену в I фреймі тієї ж GoP. Таким чином, відношення між розміром першого P фрейму та розміром I фрейму в межах однієї і тієї ж GoP, φ, показує кореляцію в часовій області $\varphi = \frac{P1}{I} = \Delta\varphi$. Збільшення $\Delta\varphi$ означає збільшення швидкості руху відео, і зменшення кореляції в часовій області.

Поєднання трьох станів у просторовій області з трьома станами у часовій області дає дев'ять станів для кожного GoP, як представлено на Рис. 5.33. Наприклад, стан LM представляє GoP з низькою кореляцією в просторовій області і з середньою кореляцією в часовій області. Будь-які два стани поєднані між собою. Імовірність переходу між двома станами визначається часовою та просторовою ймовірностями переходу. Оскільки часові та просторові процеси незалежні, то ймовірність переходу від стану LM до стану MM, наприклад,

визначається як $\Pr\{LM \to MM\} = \Pr s\{L \to M\} \times \Pr t\{M \to M\}$, де $\Pr s\{L \to M\}$ і $\Pr t\{M \to M\}$ визначає імовірність просторового переходу від стану L до M, та імовірність переходу в часовому домені від стану M до M, відповідно. Кожна імовірність $\Pr t$ і $\Pr s$ отримана шляхом підрахунку кількості переходів між двома пов'язаними станами.

Оскільки модель на рівні GoP не покриває сплескову природу інтенсивності надходження трафіку в межах GoP, модель слід розширити для урахування різних типів фреймів всередині кожної GoP.

Часовий крок для моделі Маркова на рівні фреймів відповідає тривалості відеофрейму. Для відео MPEG кожен стан на рівні GoP відповідає 12-елементному ланцюгу Маркова на рівні кадру, як показано на Рис. 5.33.

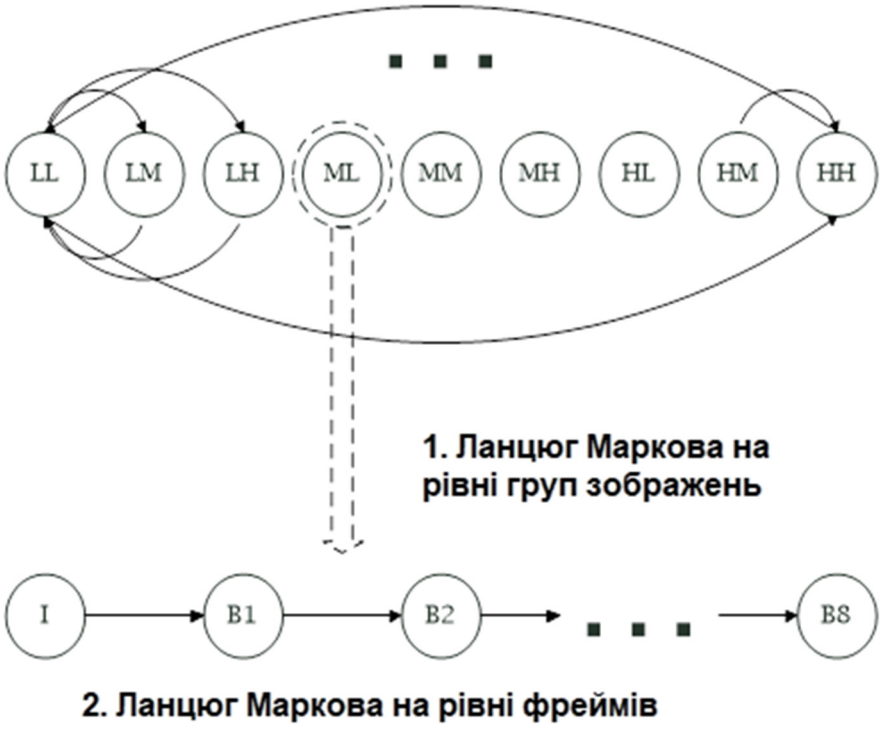


Рис. 5.33. Запропонована модель трафіку IPTV

Це являє собою 12 кадрів у GoP. Імовірність переходу між станами в межах GoP має детерміновану природу. Кожен стан відповідає іншому типу фрейму з іншою швидкістю надходження. Враховуючи обмежену кількість станів, велика

кількість кадрів з широким діапазоном розмірів кадрів належить до одного стану. Отже, критична проблема полягає в тому, як пов'язати розміри кадрів зі станами.

Розмір I фрейму визначається тільки кореляцією у просторовій області. Тому, розмір I фрейму у станах XL, XM та XH однаковий для $X=L,M,H$ в просторовій області. Найпростіший спосіб визначити розмір фрейму у кожному стані – це усереднити розмір всіх фреймів I, які належать до одного стану

$$\overline{I^X} = \sum_{I_j \in \{state\ XL,XM,XH\}} I_j \Big/ N^X \tag{5.11}$$

де N^X – кількість I фреймів у станах XL, XM та XH, а I_j є розміром фрейму I у j-ій GoP. I^X в (1) є середнім розміром фрейму I у станах XL, XM та XH, що може бути використано для подання розміру фрейму I у відповідних станах. Відповідно, розмір фрейму $P1$ може бути визначений для кожного із станів шляхом усереднення.

Для решти B та P фреймів, оцінимо спершу кореляцію всередині GoP. Коефіцієнт кореляції двох послідовностей i та j визначається як $R(i,j) = \frac{CoV(i,j)}{\sqrt{CoV(i,i)CoV(j,j)}}$, де CoV визначає коваріацію. Величина R між B/P фреймами та $P1$ у одній GoP підлягає оцінці. Величина R меншою мірою залежить від відеоконтенту, схеми компресії тощо.

Зазвичай, є значна кореляція між фреймами всередині GoP. Тому розміри решти P/B фреймів генеруються на підставі розміру фрейму $P1$ з використанням лінійного рівняння

$$\overline{F_T^K} = \propto_T^K \overline{P_1^K},\ \text{for}\ T \in \{B_1, B_2, .., B_8, P_2, P_3\} \tag{5.12}$$

де $\overline{F_T^K}$ визначає усереднений розмір фрейму конкретного типу, і розраховується подібно до (5.11), а K визначає три стани в часовій області. На відміну від підходу, де використовуються усереднені розміри фреймів P та B в GoP для представлення всіх фреймів P/B в одному і тому ж GoP, ми розрізняємо розміри фреймів P та B в різних розташуваннях. Це викликано тим, що HD відео має

значно більші розміри фреймів, ніж відеодані, досліджені раніше. Навіть з урахуванням того, що значення $\propto_T^K$ для різних типів фреймів P/B можуть бути близькими, після множення в (5.12) з суттєвим розкидом розмірів фрейму I розбіжність буде значною.

Однією з найпоширеніших реалізацій балансування навантаження за допомогою DNS є Berkeley Internet Name Domain (BIND). Це дозволяє дублювати адресні записи (A записи) для певного хоста з різними IP-адресами. Сервер іменування по черзі перетворює адреси на будь-яке ім'я, що має декілька записів A, і відомий як DNS round robin.

Як правило, коли клієнтський комп'ютер хоче ініціювати підключення до певного сервісу, він робить запит на розв'язання імені постачальника послуг [3]. Цей запит буде оброблений DNS-сервером, який у випадку Round Robin DNS буде виконувати роль контролера розподілу навантаження. Замість того, щоб повертати фіксовану адресу, сервер DNS повертає адресу пулу доступних адрес сервера. DNS-сервер проходить через всі адреси в пулі один за одним і рівномірно розподіляє запити серед усіх доступних постачальників послуг.

На Рис. 5.34 показана спрощена модель механізму розподілу навантаження Round Robin DNS.

Щоб створити модель для цього механізму, нам доведеться зробити деякі припущення:

1. Маршрутизація пакетів від джерела до пункту призначення по всій мережі відбувається миттєво.

2. Розмір запитів DNS та розмір відповідей DNS фіксуються для всього сценарію.

3. Всі черги мають нескінченні розміри. Практично, запити та відповіді DNS відносно невеликі, якщо порівнювати їх із обсягом оперативної пам'яті.

4. Весь трафік має однаковий пріоритет у чергах, не застосовується зважене чергування.

Метою є визначення того, скільки запитів можна обробити, перш ніж з'являтиметься тайм-аут запиту, скільки часу потрібно для завершення запиту DNS.

Припускаючи, що час, необхідний для отримання та обробки запиту, перевищує час, необхідний для відправки відповіді, загальний час, необхідний для завершення запиту, - це загальний час, необхідний для отримання та обробки всіх попередніх запитів у черзі на сервері, плюс час, необхідний для відправки відповіді клієнту. Ця формула повинна бути придатною відповідно до попереднього припущення:

$$T = (N+1) * \left(P + \frac{S_Q}{BW_1 * C_1} \right) + \frac{S_R}{BW_2 * C_2}$$

де T час (у мілісекундах), необхідний для завершення запиту DNS, N це кількість запитів DNS, що передують розглянутому запиту в серверній черзі, P це час (в мілісекундах), необхідний серверу для обробки певного запиту, S_Q це розмір (у кілобайтах) запиту DNS, BW_1 це пропускна здатність каналу зв'язку (в Мбіт / с) від клієнтів до DNS-сервера, BW_2 це пропускна здатність каналу зв'язку (в Мбіт) від DNS-сервера до клієнтів, C_1 це завантаженість каналу від клієнтів до сервер DNS (значення від 0 до 1), C_2 це завантаженість каналу від DNS-сервера до клієнтів (значення від 0 до 1) [168].

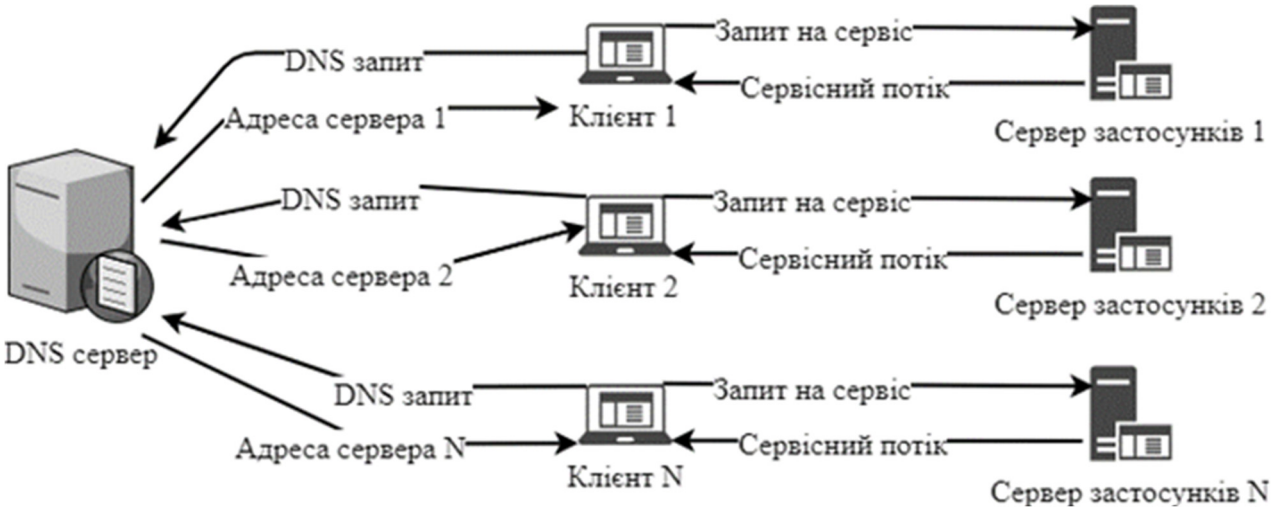


Рис. 5.34. Механізм балансування Round-robin DNS

У попередній формулі не розглядаються перевантаження, спричинені розміщеними запитами DNS. Для підрахунку перевантажень, спричинених запитами DNS, завантаженість повинна бути отримана як:

$$C' = \frac{N_T * S}{BW} + C$$

де N_T це загальна кількість запитів / відповідей у черзі сервера DNS, S це розмір запиту / відповіді, а BW це пропускна здатність каналу.

Тепер, застосовуючи вираз перевантаження в попередній формулі, ми отримуємо:

$$T = (N+1) * \left[P + \frac{S_Q}{BW_1 * \left(\frac{N * S_Q}{BW_1} + C_1 \right)} \right] + \frac{S_R}{BW_2 * C_2}$$

$$T = (N+1) * \left[P + \frac{S_Q}{\left(N * S_Q + BW_1 * C_1 \right)} \right] + \frac{S_R}{BW_2 * C_2} \quad (5.13)$$

Формула (5.13) дає змогу обчислити час, необхідний для завершення запиту DNS, але вона не розглядає черги, які можуть бути викликана DNS-відповідями, зробленими сервером. У цій формулі ми припускаємо, що час, необхідний для отримання та обробки запиту DNS, перевищує час відправки відповіді клієнту.

Якщо час, необхідний для відправки відповіді DNS, перевищує час, необхідний для отримання та обробки запиту, то загальний час, необхідний для завершення запиту, буде час, необхідний для відправки DNS-відповідей, плюс час, необхідний для отримання і обробки першого запиту. Таким чином, відповідна формула для обчислення загального часу від початку запиту до закінчення буде:

$$T = P + \frac{S_Q}{BW_1 * C_1} + \frac{(N+1) S_R}{BW_2 * \left(\frac{N * S_R}{BW_2} + C_2 \right)} \quad (5.14)$$

де T час (у мілісекундах), необхідний для завершення запиту DNS, N це кількість запитів DNS, що передують розглянутому запиту в серверній черзі, P

це час (в мілісекундах), необхідний серверу для обробки певного запиту, S_Q це розмір (у кілобайтах) запиту DNS, BW_1 це пропускна здатність каналу зв'язку (в Мбіт/с) від клієнтів до DNS-сервера, BW_2 це пропускна здатність каналу зв'язку (в Мбіт/с) від DNS-сервера до клієнтів, C_1 це завантаженість каналу від клієнтів на сервер DNS (величина між 0 та 1), C_2 це завантаженість каналу з DNS-сервера до клієнтів (величина між 0 та 1).

З використанням співвідношень (5.13) та (5.14), можемо визначити максимальну кількість DNS запитів, які можуть бути оброблені до появи тайм-аутів.

Коли час, необхідний для отримання та обробки запиту на сервері, перевищує час, необхідний для відправлення відповіді клієнту, максимальна кількість запитів клієнта, доступних до того, як DNS-запити почнуть затримуватися, становитиме

$$N_{total} = -\frac{C_2 * S_Q + BW_1 * C_1 * S_R}{S_R * \left(S_Q + BW_1 * C_1 + BW_1 * C_1 * P - BW_1 * C_1 * T_{timeout}\right)} + \\ + \frac{BW_1 * C_1 * C_2 * P - BW_1 * C_1 * C_2 * T_{timeout}}{S_R * \left(S_Q + BW_1 * C_1 + BW_1 * C_1 * P - BW_1 * C_1 * T_{timeout}\right)}$$

Повертаючись до співвідношення (5.14), ми можемо зобразити графік, який відображатиме час, необхідний для виконання запиту за різних обставин. Для наших цілей, зафіксуємо величини $S_Q = S_R = 512\,\text{bytes}$, $P = 0.04\text{sec}$, які є середніми значеннями для типового запиту DNS. Припустимо, що $BW_1 = BW_2$ і підставимо значення 100 Мбіт/с та 1 Гбіт/с. Підставляючи ці значення у формулу вище, ми отримуємо:

$$T = \frac{512 * N + 512}{12500000 * C_2 + 512 * N} + \frac{16}{390625 * C_1} + \frac{1}{25} \quad (100\ \text{Мбіт/с})$$

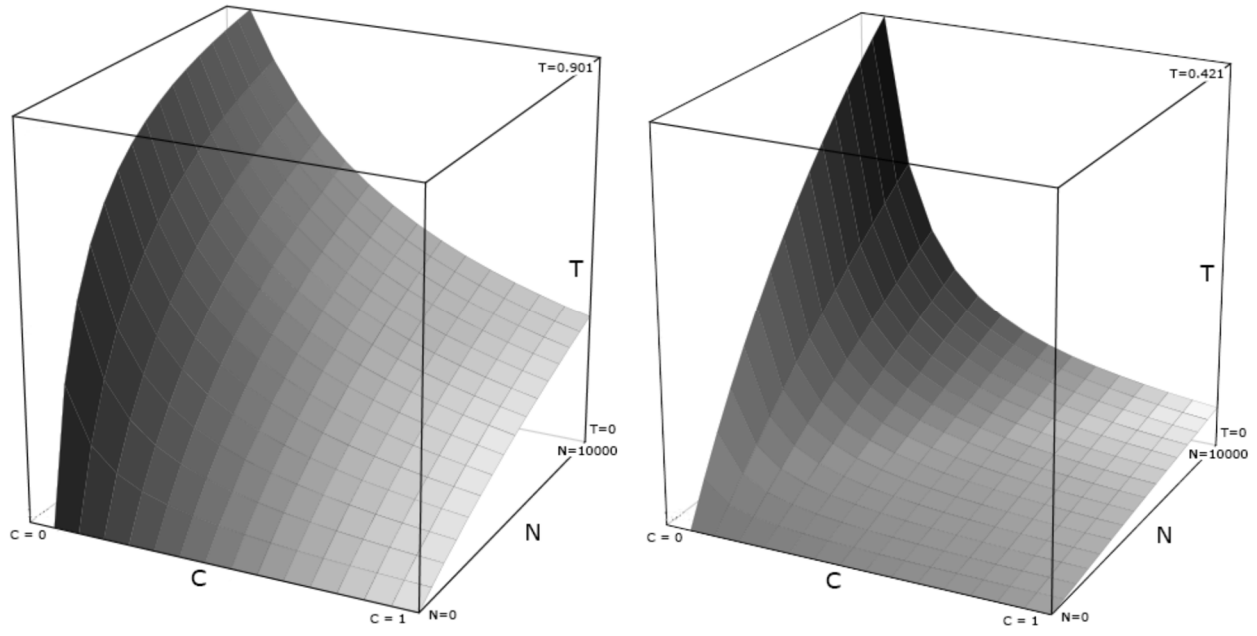
і,

$$T = \frac{512 * N + 512}{125000000 * C_2 + 512 * N} + \frac{8}{1953125 * C_1} + \frac{1}{25} \quad (1\ \text{Гбіт/с})$$

У графіках ми будемо використовувати той самий діапазон значень $(0 \sim 1)$ для c_1 та c_2 і позначимо їх через C. Використаємо діапазон значень $(0 \sim 10000)$ для N.

Порівнюючи графіки на Рис. 5.35 а та б, очевидно, що використання каналу з більш високою пропускною здатністю зменшить час обробки запиту приблизно на 47% у випадку найгіршого сценарію – де перевантаження знаходиться на піку, а кількість запитів у черзі сервера – 10000.

Ці результати з точки зору користувача дуже хороші, оскільки час простою 0,901 секунди ледве помітний. Але існують два серйозних аспекти, які ми не розглядали, – це кешування DNS на клієнтському комп'ютері, а також той факт, що у випадку виникнення помилки, "новий" сервер, до якого був переключений певний клієнт, не має можливості знати, де починати обробку.



а б

Рис. 5.35. Відношення часу, необхідного для завершення запиту, до кількості запитів в черзі та завантаженості каналу, використовуючи (а) канал 100 Мбіт/с та (б) канал 1 Гбіт/с

Перша проблема може призвести до простою до 5 хвилин, якщо кеш DNS не був примусово оновлений. Друга проблема призведе до збою, і користувачеві доведеться переходити назад до місця, де завершився невдалий сеанс. Обидві ці

проблеми можуть бути вирішені шляхом впровадження механізму зв'язку між клієнтами та серверами для оновлення стану серверів динамічно. Ця функція недоступна в загальному алгоритмі балансування навантаження Round Robin DNS.

5.4. Висновки до п’ятого розділу

1. Проведено аналіз процесу безперервної інтеграції та запропоновано архітектуру його реалізації. Показано приклад CI, яка використовує вільне програмне забезпечення та забезпечує високу якість та підтримку оновлення версії сервісу за допомогою системи контролю версій, статичного аналізу якості програмного коду, автоматизованого тестування та утворення виконуваних файлів. Цей процес забезпечує розгортання лише працездатної нової версії сервісу. Запропонована архітектура може бути впроваджена в сучасних інформаційно-комунікаційних системах, оскільки доставка послуг включає в себе сервісну інтеграцію, а також передавання послуг кінцевому користувачеві. Слід брати до уваги поведінку телекомунікаційної мережі не тільки під час процесу доставки, але й під час безперервної інтеграції сервісу, оскільки компоненти цього процесу є територіально рознесені та поєднуються мережними засобами. Зважаючи на ефективність роботи мережі в процесі передачі сервісних потоків, ми повинні враховувати потреби сервісну його пріоритет та важливість.

2. Проведено дослідження методу повного впорядкованого групового розсилання в розподілених ІТС на базі порядку настання подій. Ефективність стандартного методу проаналізовано на основі черг повідомлень типу FIFO і non-FIFO. Через нездатність цього методу працювати у випадку порушення порядку, а ще гірше – втрати повідомлення, його застосування у реальних розподілених ІТС ускладнене. Тому розроблено удосконалений метод повного впорядкованого групового розсилання, який забезпечує ефективну синхронізацію подій в розподілених ІТС навіть у випадку втрати повідомлення процесу-відправника на прикладному рівні. Для цих цілей у метод введено обмеження тривалості очікування відповіді від усіх процесів-адресатів, чим досягнуто підвищення

імовірності працездатності вього розподіленого процесу з рахунок зростання обсягів службових даних, яка передаються між процесами, що є елементами комунікації.

3. Для підтвердження адекватності розробленої моделі досліджено процес обслуговування мультимедійних інформаційних потоків апаратним маршрутизатором та програмним компонентом гетерогенної розподіленої інформаційно-телекомунікаційної системи. Отримано продуктивність програмного компонента у режимі без утворення черг. Адекватність перевірено на основі порівняння затримки пакетів у програмному та апаратному вузлах. Для підвищення точності оцінок затримки дослідження повторено із трьома послідовно включеними вузлами мережі. Дано оцінку тривалості формування інформаційного потоку програмним генератором трафіку та показано її частку у тривалості передавання пакету від генератора до одержувача. Максимальне відхилення затримки пакетів між програмним та апаратним компонентами знаходиться в межах 0,5 %. Підтверджено статистичну гіпотезу про відповідність законів розподілу затримок у програмному та апаратному вузлах.

4. Доведено, що балансування навантаження за допомогою інтегрованої системи керування розгортанням інформаційних сервісів дає змогу зменшити тривалість обслуговування запитів у середньому у 3 рази. Інтегральна оцінка використання телекомунікаційних та інформаційних ресурсів показала, що запропонований метод дав змогу зменшити затримку надання сервісу та розвантажити найбільш завантажені сервери.

5. Для дослідження балансування потоків IPTV розроблено дворівневу марківську модель такого потоку на основі як просторову, так і часову кореляцію в кодових послідовностях MPEG. Модель описує динаміку процесу зміни кадрів і груп зображень в потоці IPTV та дає змогу оцінити характер потоку у процесі балансування навантаження за методом Round Robin DNS. Для дослідження затримок обслуговування запитів у розподіленій системі з балансуванням навантаження згідно згаданого методу розроблено модель пересилання запитів та виведено співвідношення для оцінки затримки для каналів зі змінною

пропускною здатністю. Наведено порівняльні оцінки затримки, які дають змогу стверджувати, що зростання пропускної здатності каналу у 10 разів призводить до зниження затримки обслуговування запиту у середньому у 2,5 рази, що пояснюється обмеженнями методу балансування навантаження.

РОЗДІЛ 6. ПОБУДОВА ТА ТЕСТУВАННЯ ПОВЕДІНКИ СКЛАДНОЇ ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНОЇ СИСТЕМИ В УМОВАХ ЗРОСТАННЯ НАВАНТАЖЕННЯ

6.1. Методи розгортання cloud інфраструктур на основі принципу «платформа як сервіс»

6.1.1. Принцип віртуалізації та делегування функцій обслуговування

Cloud обчислення – це тип обчислень на базі Інтернету, який забезпечує загальні обчислювальні ресурси та дані для комп'ютерів та інших пристроїв за запитом [109]. Це модель для забезпечення широкомасштабного доступу до спільного пулу настроюваних обчислювальних ресурсів (наприклад, комп'ютерних мереж, серверів, сховищ, програм та сервісів), які можуть бути швидко та з мінімальними зусиллями керування надані користувачеві. Рішення у сфері хмарних обчислень і зберігання надають користувачам та підприємствам різні можливості зберігати та обробляти свої дані в приватних або сторонніх центрах обробки даних, які можуть бути розташовані далеко від користувачів. Хмарне обчислення покладається на розподіл ресурсів для досягнення узгодженості та економії ресурсів [29, 60].

Віртуалізація є потужним інструментом масштабованих систем, і віртуалізовані системи стають дедалі популярнішими. Але існує велика різниця між віртуалізованими серверами та віртуалізованою інфраструктурою. Віртуалізований сервер керується лише як цілісна система. Це означає, що якщо потрібно керувати веб-сервісом або базою даних на цьому сервері, то слід використовувати інші інструменти керування, ніж ті, що використовуються для керування самим віртуалізованим сервером. Віртуалізована інфраструктура [172] є іншою: вона робить всі компоненти розподілених ІТС видимими та керованими за допомогою одного інструмента, і робить операційній системі фізичного пристрою ці компоненти невидимими. Інфраструктура – це простір, призначений для взаємопов'язаних компонентів системи [41, 100].

6.1.2. Основні провайдери cloud сервісів

Розглянемо процес розгортання множини веб-серверів та їхнього функціонування з балансувальником навантаження та базою даних з резервним копіюванням. Розгортання цієї конфігурації на віртуальних серверах відносно подібне до розгортання її на звичайних серверах. Перевагою є більша гнучкість і краще використання ресурсів, але управління окремими компонентами сервісу насправді не покращується.

На противагу, у cloud системах надається спеціалізований веб-інтерфейс, який дає змогу заповнювати його робочий простір пристроями, такими як веб-сервери, сервери застосунків, шлюзи тощо. Через інші компоненти цього ж інтерфейсу доступне конфігурування параметрів обраних cloud елементів: елементи моніторингу компонентів, зв'язок із іншими компонентами, системні конфігурації, які впливають на продуктивність та пропускну здатність. Cloud приховує аспекти розподілу ресурсів та керування основною інфраструктурою сервера, що робить систему надання послуг простою та адаптивною.

Amazon Elastic Compute Cloud (EC2) утворює центральну частину платформи хмарних обчислень Amazon Web Services (AWS), дає можливість користувачам орендувати віртуальні обчсилювальні компоненти, на яких запускаються власні сервіси. EC2 забезпечує масштабоване розгортання застосунків, надаючи веб-сервіс, за допомогою якого користувач може завантажувати образ машини Amazon Machine Image (AMI) для налаштування віртуальної машини, яку Amazon називає "екземпляром", що містить будь-яке програмне забезпечення, яке необхідне. Користувач може створювати, запускати та припиняти серверні екземпляри, коли це потрібно, оплачуючи за час використання активних серверів – цим пояснюється термін "еластичний" в назві. EC2 надає користувачам контроль над географічним розташуванням екземплярів, що дозволяє оптимізувати затримку та забезпечує високий рівень надлишковості [196].

Термін обчислювальної одиниці (ECU) був введений компанією Amazon як абстракція обчислювальних ресурсів. EC2 використовує віртуалізацію Xen.

Кожна віртуальна машина, яка називається "екземпляром", функціонує як віртуальний приватний сервер. Ієрархія сервісних рівнів базується на еластичних обчислювальних одиницях. Продуктивність ідентичних віртуальних машин може відрізнятися.

Google Cloud Platform – це сервіс хмарних обчислень від Google, який пропонує хостинг у тій самій допоміжній інфраструктурі, яку Google використовує для продуктів кінцевого споживача, таких як Пошук Google і YouTube. Cloud Platform надає розробникам продукти для створення низки ITC – від простих веб-сайтів до складних програм.

Google Cloud Platform є частиною набору корпоративних сервісів від Google Cloud, і надає набір модульних хмарних сервісів із великою кількістю інструментів розробки. Наприклад, хостинг і обчислення, хмарне зберігання, API перекладів та API передбачень. App Engine тепер підтримує Node.js, Ruby, Java 8, Python 2.7 або 3.5, Go 1.8, а також PHP 7.1 і .NET Core, всі підтримують 99.95% SLA на App Engine.

Microsoft Azure – це сервіс хмарних обчислень, створений компанією Microsoft для побудови, розгортання та керування сервісами через глобальну мережу центрів обробки даних, керованих Microsoft. Вона надає програмне забезпечення як сервіс, платформу як сервіс та інфраструктуру як сервіс і підтримує різні мови програмування, інструменти та засоби, включаючи як програмне забезпечення Microsoft, так і стороннє програмне забезпечення та системи.

Microsoft Azure використовує спеціалізовану операційну систему, яка називається Microsoft Azure, для запуску свого базового шару: кластеру, що розміщується в центрах обробки даних Microsoft, який керує обчислювальними та запатентованими ресурсами комп'ютерів і забезпечує ресурси (або їх підмножини) для запущених програм на Microsoft Azure. Microsoft Azure описується як "хмарний шар" на вершині ряду серверів Windows Server, які використовують Windows Server та індивідуальну версію Hyper-V, відому як гіпервізор Microsoft Azure, для забезпечення віртуалізації сервісів.

OpenStack – це відкрита програмна платформа для хмарних обчислень, яка в основному використовується як IaaS, за допомогою неї віртуальні сервери та інші ресурси стають доступними для клієнтів. Програмна платформа складається з взаємопов'язаних компонентів, які керують різноманітними апаратними пулами для обробки і зберігання та мережевих ресурсів у центрі обробки даних. Користувачі можуть керувати ними через веб-панель інструментів, за допомогою інструментів командного рядка або за допомогою інтерфейсу REST API.

OpenStack Networking (Neutron) – це система управління мережами та IP-адресами. OpenStack Networking гарантує, що мережа не є вузьким місцем або не обмежує розгортання хмари та надає користувачам можливість самообслуговування навіть у конфігураціях мережі.

OpenStack Networking надає мережеві моделі для різних систем або груп користувачів. Стандартні моделі включають в себе плоскі мережі або VLAN, які відокремлюють сервери та трафік. OpenStack Networking управляє IP-адресами, що дозволяє використовувати спеціальні статичні IP-адреси або DHCP. Плаваючі IP-адреси дозволяють динамічно перенаправляти трафік на будь-які ресурси IT-інфраструктури, тому користувачі можуть перенаправляти трафік під час технічного обслуговування або у випадку відмови.

SmartCloud від IBM – це набір сервісів хмарних обчислень для бізнесу, що пропонуються компанією IBM. Хмара IBM включає IaaS, SaaS PaaS, що пропонуються через публічні, приватні та гібридні моделі розгортання хмар, на додаток до компонентів, які складають ці хмари.

IBM надає у користування три апаратні середовища для хмарних обчислень. Ці платформи пропонують вбудовану підтримку для віртуалізації. Для віртуалізації IBM пропонує рішення IBM Websphere, які підтримують моделі та відкриті стандарти для віртуалізації.

Рівень керування хмарної системи IBM включає в себе проміжне програмне забезпечення IBM Tivoli. Інструменти управління надають можливість регулювати використання за допомогою автоматичного надання та відключення,

моніторингу операцій та використання лічильника при відстеженні витрат та розподілу білінгу. Останній шар структури забезпечує інтегровані інструменти завантаження. Робочі навантаження для хмарних обчислень – це послуги або екземпляри коду, які можуть бути виконані для задоволення конкретних бізнес-потреб. IBM пропонує інструменти для взаємодії, розробки та тестування на основі хмар, розробки застосунків, аналітики, інтеграції між компаніями та безпеки.

6.1.3. Властивості інформаційних інфраструктур, побудованих на основі cloud сервісів

Основна ідея хмари полягає в тому, що надається доступ до всієї розподіленої ІТС через Інтернет, при цьому невідомі деталі побудованої інфраструктури, яку вона використовує [103].

До основних властивостей хмарних систем слід віднести такі:

- Гнучкість – масштабування вгору та вниз, щоб відповідати вимогам бізнесу. У сучасній економіці така гнучкість є ключовою. Надається можливість відрегулювати витрати на ІТ, щоб відповідати найближчим потребам організації.
- Безпека – дані та елементи системи в хмарі захищені значно краще, ніж у випадку орендованої інфраструктури.
- Продуктивність – cloud сервіс дає змогу реалізувати інфраструктуру з динамічно змінною продуктивністю, що досягається шляхом перемикання сервісних рівнів використовуваних хмарних компонентів.
- Вартість – використання хмарної технології зменшує витрати на обслуговування. Багато з прихованих витрат, які зазвичай пов'язані з впровадженням програмного забезпечення, налаштуванням, апаратним забезпеченням, технічним обслуговуванням та навчанням, залучаються до прозорої абонентської плати.
- Відкритість – стандарти Інтернету та веб-сервіси дозволяють підключати послуги одна до одної. Це означає, що надається можливість

централізувати свою інформацію та отримувати доступ до неї з будь-якої точки світу, на будь-якому комп'ютері або мобільному пристрої.

6.1.4. Характеристики надання послуг в інформаційно-комунікаційних системах, побудованих на основі cloud сервісів

Хмарні обчислення є результатом еволюції та прийняття існуючих технологій та парадигм. Мета хмарних обчислень полягає в тому, щоб дозволити користувачам отримувати вигоду від усіх цих технологій без потреби в глибоких знаннях або досвіді з кожним з них. Хмара має на меті скоротити витрати та допомагає користувачам зосереджуватися на основному бізнесі, а не на долати IT-перешкоди. Основною технологією для функціонування хмарних обчислень є віртуалізація. У хмарних системах програмне забезпечення віртуалізації віртуально ділить фізичний обчислювальний пристрій на множину віртуальних пристроїв, які доступні окремо і керовані для виконання обчислювальних завдань. За допомогою віртуалізації на рівні операційної системи, яка, по суті, створює масштабовану систему декількох незалежних обчислювальних пристроїв, ресурси простої обчислювальної техніки можуть бути виділені та використані більш ефективно. Віртуалізація забезпечує оперативність, необхідну для прискорення операцій з інформаційних технологій, а також зменшує витрати за рахунок збільшення ступеню використання інфраструктури [93]. Автономні обчислення автоматизують процес, за допомогою якого користувач може надавати ресурси за запитом. Мінімізуючи залученість користувачів, автоматизація прискорює процес, зменшує витрати на робочу силу та зменшує імовірність помилок у роботі [94, 95]. Користувачі регулярно стикаються з складними діловими проблемами. Хмарні обчислення приймають поняття з SOA, які можуть допомогти користувачеві порушити ці проблеми в службах, які можуть бути інтегровані для забезпечення сервісу. Хмарні обчислення надають всі свої ресурси як послуги, а також використовують усталені стандарти та найкращі практики, отримані в області SOA, щоб забезпечити універсальний та легкий доступ до хмарних сервісів стандартним способом.

Хмарне обчислення також використовує концепції утилітарних обчислень для надання показників для використовуваних сервісів [96, 97]. Такі показники є основою загальнодоступних моделей, які використовують плату за використання. Крім того, вимірювані сервіси є важливою частиною циклу зворотного зв'язку у автономному обчисленні, що дозволяє надавати послуги масштабу за запитом та виконувати автоматичне відновлення після відмов. Хмарні обчислення є своєрідною обчислювальною мережею; вона розвивалася, вирішуючи проблеми якості сервісу та проблеми надійності [98, 99]. Хмарне обчислення надає інструменти та технології для створення інтенсивних паралельних застосунків з більш доступними цінами порівняно з традиційними паралельними обчислювальними методами.

Програмне забезпечення як сервіс (SaaS). Ресурси, надані споживачеві, полягають у використанні застосунків постачальника, що працюють у хмарній інфраструктурі. Програми доступні з різних клієнтських пристроїв через інтерфейс тонкого клієнта, наприклад, веб-браузер або інтерфейс програми. Споживач не керує та не контролює базову інфраструктуру системи, включаючи мережу, сервери, операційні системи, сховища або навіть окремі можливості програми, за винятком обмежених для користувача конкретних параметрів конфігурації застосунків [246, 247].

Платформа як сервіс (PaaS). Ресурси, надані споживачеві, полягають в тому, щоб забезпечити розгортання на хмарну інфраструктуру створеного або придбаного клієнтом застосунку, який реалізовано з використанням мов програмування, бібліотек, сервісів та інструментів, підтримуваних провайдером. Споживач не керує та не контролює базову інфраструктуру хмар, включаючи мережу, сервери, операційні системи або сховище, але контролює розгорнуті програми та, можливо, конфігураційні параметри для середовища хостингу сервісів.

Інфраструктура як сервіс (IaaS). Ресурси, надані споживачеві, полягають у забезпеченні обробки, зберігання, мережевої взаємодії та інших основних обчислювальних ресурсів, коли споживач може розгортати та запускати довільне

програмне забезпечення, яке може включати операційні системи та програми. Споживач не керує та не контролює основну інфраструктуру хмари але контролює операційні системи, зберігання та розгорнуті додатки; і, може мати обмежений контроль над окремими компонентами мережі [61, 62, 78].

Хмарні обчислення демонструють такі ключові характеристики:

- Адаптивність для організацій може бути вдосконалена, оскільки хмарне обчислення може збільшити гнучкість користувачів при повторному наданні, додаванні або розширенні ресурсів технологічної інфраструктури.
- Скорочення витрат. Модель розгортання публічної хмари перетворює капітальні витрати (наприклад, купівлю серверів) на операційні витрати. Це знижує бар'єри для входу в бізнес, оскільки інфраструктура, як правило, надається третьою стороною і не потребує придбання для одноразових або неінтенсивних обчислювальних завдань. Ціноутворення на корисній обчислювальній основі є дрібнозернистим, з опціями білінгу на основі використання. Крім того, для реалізації проектів, що використовують хмарне обчислення, потрібні менші внутрішні інформаційні навички.
- Незалежність від пристрою та місцезнаходження дозволяє користувачам отримувати доступ до систем із використанням веб-браузера незалежно від їх місцезнаходження та пристрою, який вони використовують (наприклад, ПК, мобільний телефон). Оскільки інфраструктура розташована за межами офісу (зазвичай, вона надається третьою стороною) та доступ до неї здійснюється через Інтернет, користувачі можуть підключатися з будь-якого місця.
- Технічне обслуговування застосунків cloud обчислення простіше, оскільки їх не потрібно встановлювати на кожному комп'ютері користувача, і їх можна отримати з різних місць (наприклад, різних робочих місць, під час подорожей тощо).

- Різноманітність дозволяє розподіляти ресурси та витрати у великій кількості користувачів, що дозволяє забезпечити:
  - централізація інфраструктури в місцях з меншими витратами (наприклад, нерухомість, електрика тощо)
  - збільшення завантаженості пікових навантажень (користувачам не потрібен інженер і утримування власних ресурсів та обладнання для максимально можливого навантаження)
  - Підвищення ефективності систем, які часто використовуються лише на 10-20%.
- Продуктивність контролюється IT-фахівцями від постачальника послуг, а послідовність і вільно пов'язані архітектури будуються за допомогою веб-сервісів як системного інтерфейсу.
- Продуктивність може бути збільшена, коли декілька користувачів можуть одночасно працювати над однаковими даними, а не чекати збереження та надсилання електронною поштою. Час може бути збережений, оскільки інформацію не потрібно повторно вводити, коли поля співпадають, а також не потрібно інсталювати оновлення програмного забезпечення на комп'ютер.
- Надійність покращується завдяки використанню декількох резервних систем, що створює добре розроблене хмарне обчислення, яке підходить для безперервності бізнесу та аварійного відновлення.
- Масштабованість та еластичність за допомогою динамічного ("на вимогу") надання ресурсів на дрібнозернистому рівні самообслуговування в режимі реального часу. Це дає змогу мастабувати систему, коли потреба у використанні змінюється, якщо ресурси не використовуються.
- Безпека може покращитись завдяки централізації даних, збільшенню ресурсів, орієнтованих на безпеку тощо, але проблеми можуть зберігатись у зв'язку з втратою контролю над певними конфіденційними даними та відсутністю безпеки для збереження.

Три аспекти є новими в Cloud Computing:

1. Ілюзія нескінченних обчислювальних ресурсів, наявних на вимогу, таким чином вона усуває потребу користувачів Cloud Computing у довгостроковому плануванні використання ресурсів.

2. Масштабованість – обсяг ресурсів, наданих провайдером, може визначатися за потребою користувача або на підставі широкого кола об'єктивно зібраних метрик.

3. Можливість платити за користування обчислювальними ресурсами на короткостроковій основі (наприклад, процесори на годину та зберігання на день) і запускати їх у міру необхідності, що досягається можливістю позначення ресурсів неактивними.

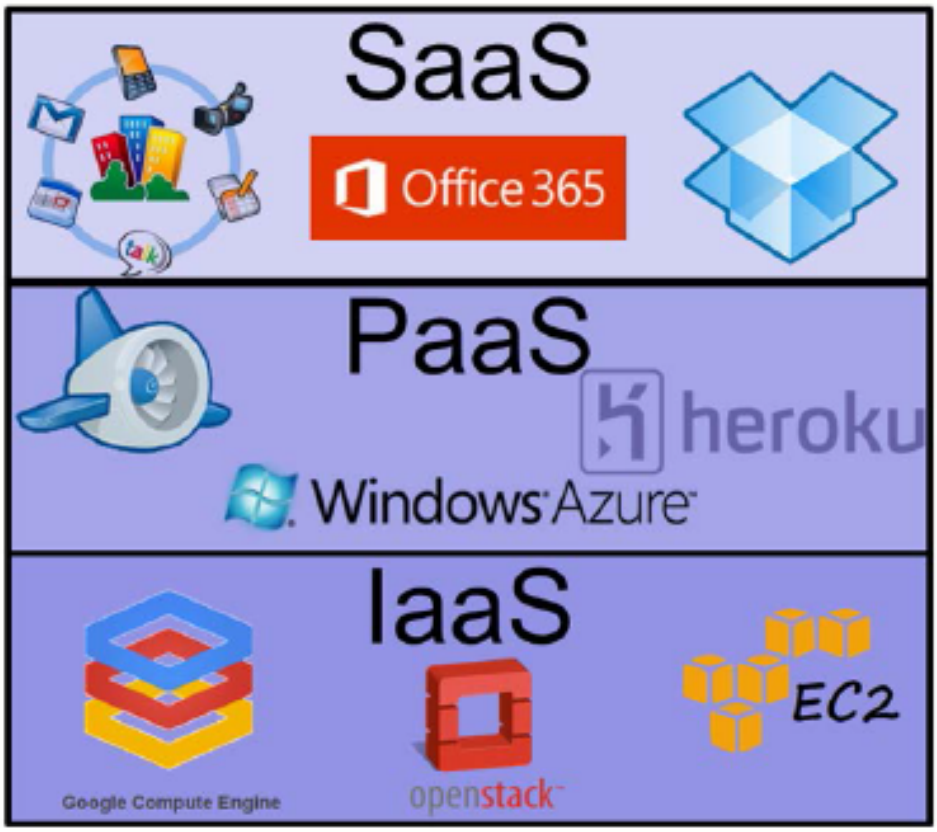


Рис. 6.1. Сервіси cloud провайдерів

Хоча сервіс-орієнтована архітектура виступає за "все як сервіс" (з акронімами EaaS або XaaS або просто aaS), постачальники хмарних обчислень пропонують свої послуги за різними моделями, з яких три стандартні моделі для NIST є: IaaS, PaaS та SaaS. Ці моделі пропонують збільшення абстракції; таким чином вони часто зображуються як шари у стеку: інфраструктура, платформа та

програмне забезпечення як сервіси, але вони не повинні бути пов'язаними. Наприклад, можна встановити SaaS, реалізований на фізичних машинах (без власне фізчиних машин), не використовуючи основні шари PaaS або IaaS, і навпаки, можна запустити програму на IaaS і отримати доступ до неї безпосередньо, не огорнувши її як SaaS.

За даними Міжнародної спеціалізованої цільової групи (IETF), найпопулярнішою моделлю cloud-сервісу є постачальники, що пропонують обчислювальну інфраструктуру – віртуальні машини та інші ресурси – як сервіс для користувачів. IaaS реалізує онлайнові сервіси, які абстрагують користувача від деталей інфраструктури, таких як фізичні обчислювальні ресурси, розташування, розподіл даних, масштабування, безпека, резервне копіювання тощо. Гіпервізор, такий як Xen, Oracle VirtualBox, Oracle VM, KVM, VMware ESX / ESXi або Hyper-V запускає віртуальні машини на фізичній машині. Операційні системи Cloud можуть підтримувати велику кількість віртуальних машин і можливість масштабування сервісів вгору і вниз відповідно до різних вимог клієнтів. Контейнери Linux працюють в ізольованих розділах одного ядра Linux, що працює безпосередньо на фізичному обладнанні. Cgroups та області імен Linux – це основні технології ядра Linux, які використовуються для ізоляції, захисту та керування контейнерами. Контейнералізація пропонує вищу продуктивність, ніж віртуалізація, тому що немає гіпервізора і відповідних накладних втрат. Також, динамічно збільшується ємність контейнерів з обчислювальним навантаженням, що усуває проблему перевищення ресурсів та дозволяє здійснювати тарифікацію на основі використання. Хмари IaaS часто пропонують додаткові ресурси, такі як бібліотека диска з віртуальним комп'ютером, зберігання необроблених блоків, зберігання файлів або об'єктів, мережні екрани, балансувальники завантаження, IP-адреси, віртуальні локальні мережі (VLAN) та пакети програм.

Постачальники PaaS пропонують середовище розробки для розробників застосунків. Провайдер зазвичай розробляє набір інструментів і стандартів для розробки та каналів для розповсюдження та оплати. У моделях PaaS

постачальники хмар передають комп'ютерну платформу, яка зазвичай включає операційну систему, середовище виконання програми, базу даних та веб-сервер. Розробники застосунків можуть розробляти та керувати своїми програмними рішеннями на хмарній платформі без витрат та складності купівлі та керування основними апаратними та програмними рівнями. Деякі системи PaaS, такі як Microsoft Azure та Google App Engine, забезпечують надання обчислювальних ресурсів, які масштабуються автоматично, щоб відповідати вимогам до застосунку, щоб користувач хмари не змушений був розподіляти ресурси вручну.

У моделі SaaS користувачі отримують доступ до прикладного програмного забезпечення та баз даних. Хмарні постачальники керують інфраструктурою та платформами, на яких запущені системи користувача. SaaS іноді називають "програмним забезпеченням на вимогу", і, як правило, цей сервіс надається на платній основі або за допомогою підписки. У моделі SaaS постачальники хмарних мереж встановлюють та експлуатують прикладне програмне забезпечення в хмарі, а користувачі хмарних ІТС отримують доступ до програмного забезпечення від хмарних клієнтів. Користувачі хмар не керують хмарною інфраструктурою та платформою, де працює система. Це усуває необхідність встановлювати та запускати застосунки на власних комп'ютерах користувача, що спрощує обслуговування та підтримку. Cloud застосунки відрізняються від інших застосунків у їх масштабованості, що може бути досягнута завдяки клонуванню їх екземплярів на декілька віртуальних машин під час виконання, щоб задовольнити зміну попиту. Балансувальники навантаження розподіляють запити на множину віртуальних машин. Цей процес є прозорим для користувача хмари, який бачить лише одну точку доступу. Щоб розмістити велику кількість хмарних користувачів, застосунки у хмарі можуть бути багатоклієнтними, а це означає, що будь-яка машина може обслуговувати більше однієї організації хмарних користувачів.

Модель ціноутворення для застосунків SaaS, як правило, становить щомісячну або щорічну плату за користувача, тому ціни стають масштабованими та регульованими, якщо користувачі додаються або видаляються в будь-який

момент. Один недолік SaaS пов’язаний із зберіганням даних користувачів на сервері постачальника хмари. Як результат, може бути неавторизований доступ до даних. З цієї причини користувачі все більше залучають інтелектуальні сторонні системи керування ключами, щоб захистити свої дані.

Надання cloud послуг – це розподіл ресурсів постачальника хмар для клієнтів. Коли постачальник хмар приймає запит від клієнта, він повинен створити відповідну кількість віртуальних компонентів і розподілити ресурси для їх підтримки. Процес ведеться кількома різними способами: попередня підготовка, динамічне забезпечення та самостійне надання користувачам. У цьому контексті термін "надання" просто означає "забезпечити".

Заздалегідь замовляючи послуги, клієнт укладає договори з провайдером, а постачальник готує відповідні ресурси до початку сервісу.

За допомогою динамічного резервування постачальник виділяє більше ресурсів, якщо вони потрібні, і видаляє їх, коли вони не використовуються. Клієнт сплачує плату за використання.

За допомогою самообслуговування користувачів, клієнт купує ресурси у постачальника хмар через веб-інтерфейс, створюючи обліковий запис клієнта та оплачуючи ресурси. Ресурси постачальника доступні для використання клієнтами майже миттєво.

Система надання cloud-послуг в основному визначає, як, що і коли організація надаватиме у вигляді хмарних сервісів. Ці служби можуть бути внутрішніми, публічними або гібридними хмарними продуктами та рішеннями. Існує три різні моделі доставки:

- на вимогу: замовник або програма отримує ресурси під час запуску.
- надання користувачам: користувач / клієнт додає хмарний пристрій самостійно.
- Post-Sales / Advanced Provisioning: клієнт отримує ресурс після реєстрації контракту / послуги.

Коли мова йде про вибір постачальника хмарних послуг, власники бізнесу часто плутаються із запропонованими ними функціями та планами, і часто

проблеми виникають у зв'язку з неправильним провайдером і планом. У процесі надання хмарних сервісів особливу роль займає віртуалізація. По суті, у всіх випадках ресурс насправді емулює або імітує інший ресурс. Нижче наведено приклади віртуалізації.

Віртуальна пам'ять: на диску є набагато більше простору, ніж пам'ять комп'ютера. Тому з віртуальною пам'яттю комп'ютер звільняє цінний простір пам'яті, розміщуючи інформацію, яку він часто не використовує на дисковий простір. ПК мають віртуальну пам'ять, яка є областю дисків, яка використовується як пам'ять. Незважаючи на те, що диски дуже повільні порівняно з пам'яттю, користувач може ніколи не помічати різницю, особливо якщо система правильно керує віртуальною пам'яттю.

Програмне забезпечення – компанії створили програмне забезпечення, яке може наслідувати увесь комп'ютер. Таким чином, один комп'ютер може працювати як насправді 20 комп'ютерів. Результати консолідації серверів можуть бути значними. Наприклад, мігрувавши з центру обробки даних з тисячами серверів на той, який підтримує лише кілька сотень, можна зменшити витрати не тільки на комп'ютери, а й інші операційні витрати.

Віртуалізація має три характеристики, що робить її ідеальною для хмарних обчислень:

- розділення: у віртуалізації багато програм та операційних систем підтримуються в одній фізичній системі, розділяючи доступні ресурси.
- ізоляція: кожна віртуальна машина виділяється з її фізичної системи та інших віртуальних машин. Через таку ізоляцію, якщо один віртуальний екземпляр відмовляє, це не впливає на інші віртуальні машини. Крім того, дані не розподіляються між одним віртуальним контейнером та іншим.
- інкапсуляція: віртуальна машина може бути представлена (і навіть зберігається) як єдиний файл, тому можливо легко ідентифікувати його на основі надаваного ним сервісу. По суті, інкапсульований процес може бути бізнес-сервісом. Ця інкапсульована віртуальна машина може бути

представлена застосунком як повноцінним об'єктом. Тому інкапсуляція може захистити кожну програму, щоб вона не заважала іншій програмі.

Віртуалізація може бути широко застосована до таких компонентів:

- пам’ять;
- мережі;
- сховища даних;
- апаратне забезпечення;
- операційні системи;
- застосунки.

У хмарних обчисленнях еластичність визначається як "ступінь, до якої система здатна адаптуватися за змін робочого навантаження шляхом забезпечення резервування ресурсів і вичерпання ресурсів автономним способом, таким чином, що в кожний конкретний момент часу доступні ресурси відповідають поточному попиту настільки близько, наскільки це можливо". Еластичність є визначальною характеристикою, яка відрізняє хмарні обчислення від раніше запропонованих обчислювальних парадигм, таких як grid обчислення. Динамічна адаптація продуктивності, наприклад, шляхом зміни обчислювальних ресурсів, для задоволення різного робочого навантаження надає еластичні обчислення.

Миттєва еластичність визначається як здатність хмарного сервісу надавати послуги на замовлення, оперативно перемикаючи ресурси, коли попит підвищується або зменшується. Часто це негайна реакція клієнтів на видалення або додавання послуг у реальному часі. Миттєва еластичність також відома як швидка еластичність.

Еластичність хмар пов'язана з різними стратегіями, такими як об'єднання ресурсів, та іншим способами, які постачальники хмарних мереж використовують для надання своїх послуг. Ідея полягає в тому, що сервіс повинен мати можливість швидко збільшувати масштаб або зменшувати його відповідно до потреб окремого замовника. Публічні хмарні системи в значній мірі роблять це завдяки тому, що у будь-який час містять багато активних

клієнтів та підтримують системи, які можна легко перекофігурувати відповідно до змінного попиту.

Фахівці з інформаційних технологій виділяють еластичність хмари та масштабованість хмари. Масштабованість або економія масштабу означають, що система може бути побудована на практиці з невеликими ресурсами. Еластичність, з іншого боку, передбачає динамічну реакцію на волатильність попиту та пропозиції.

Ефективне використання існуючої мережної та IT-інфраструктури може бути досягнуто шляхом надання об'єднаних мереж та IT-ресурсів на вимогу як інфраструктурних послуг, здатних підтримувати складні технологічні процеси, наукові експерименти та спільні групи дослідників та застосунків.

Хмарне обчислення на самообслуговування є формою приватного хмарного сервісу, в якому клієнт зберігає дані та запускає застосунки, не переходячи через зовнішнього постачальника хмарних послуг. Завдяки хмарному самообслуговуванню користувачі отримують доступ до веб-порталу, де вони можуть запитувати або налаштувати сервери та запускати програми.

Переваги включають користувача, який має доступ до сховища, програм та інших сервісів, не покладаючись на свого внутрішнього постачальника, що робить це для них. Оскільки дані зберігаються у віртуалізованій мережі, апаратне забезпечення об'єднується і його використовують різні клієнти, що робить самообслуговування економічним рішенням. Існує також прозорість, що дає споживачам прямий доступ до їх хмари.

Однак для хмарного обчислення самообслуговування вимагає плавного планування. Для роботи постачальник послуг cloud сервісу самообслуговування повинен обробляти запити споживачів без власної участі. З послугами, які по суті є за замовчуванням, добре спланована автоматизація необхідна на кінцевому рівні постачальника послуг. Інтерфейс також повинен бути простим, щоб вмістити користувачів різної обізнаності з хмарними обчисленнями.

6.1.5. Вплив cloud сервісів на процеси проектування і розробки інформаційно-комунікаційних систем

Архітектура хмарних обчислень описує компоненти і підкомпоненти, необхідні для процесу хмарного обчислення. Ці компоненти, як правило, складаються з кінцевої платформи користувача (товстий клієнт, тонкий клієнт, мобільний пристрій), кінцевої платформи системи (сервери, сховища), хмарної доставки та мережі (Інтернет, Intranet, Intercloud). Разом ці складові складають архітектуру хмарних обчислень.

Перед розробкою власної архітектури хмарних систем, особливо якщо слід розглядати архітектуру з декількома хмарами / регіонами, потрібно враховувати кілька факторів, серед яких вартість, складність, швидкість, портативність, безпека [43].

Наведені нижче архітектурні схеми показують прогрес від простих до більш складних базових архітектур [32].

У стандартній трирівневій архітектурі ІТС є щонайменше один виділений сервер на кожному рівні архітектури системи [36].

Нерезервована 3-х рівнева архітектура. Якщо слід лише перевірити комунікацію між рівнями архітектури, можна використати архітектуру без резервування, щоб заощадити на витратах та ресурсах. Оскільки це архітектура побудована без резервної системи, вона, в основному, використовується для базових цілей тестування та розробки. Сервіс у такому випадку надається на кожному із рівнів архітектури на основі використання відповідних серверів. Нерезервована архітектура не рекомендується для реальних інфраструктур.

Будь-яка реальна інфраструктура, яка надається як cloud сервіс, також повинна містити резервні компоненти для маскування відмови та відновлення. Як правило, використовують серверний масив для рівня застосунків, однак можуть бути деякі сценарії, коли застосунок не може бути масштабований у автоматичному режимі. У таких випадках рекомендується побудувати резервовану багаторівневу архітектуру, де реалізована надлишковість на кожному рівні опорної архітектури.

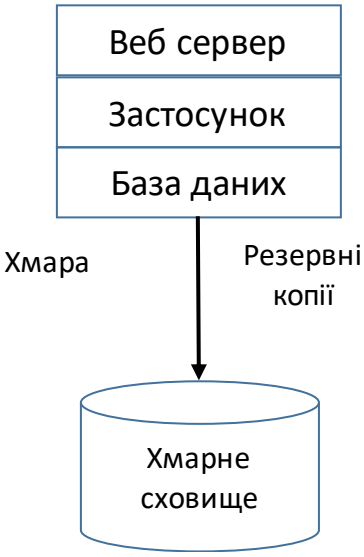


Рис. 6.2. Найпростіша дворівнева архітектура хмарної інформаційно-телекомунікаційної системи

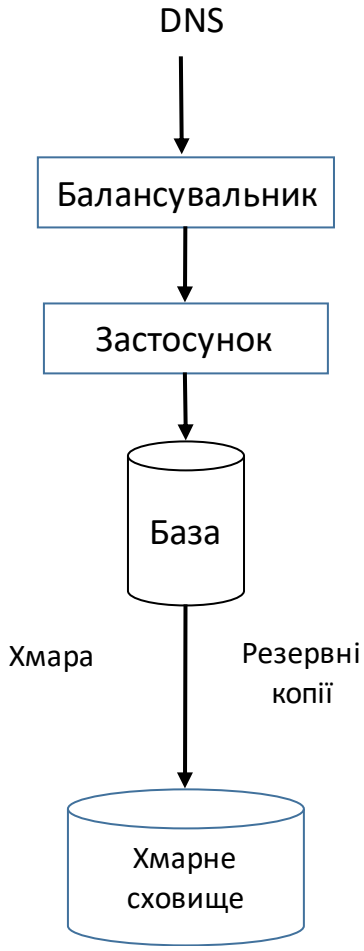


Рис. 6.3. Трирівнева архітектура хмарної інформаційно-телекомунікаційної системи без резервування

У наведеному нижче прикладі існують два сервери балансування навантаження, два сервери застосунків, а також сервери базової та підлеглих баз даних. Надмірна архітектура допоможе захистити застосунок від недоступності компонентів системи [138]. Наявність реплік баз даних у віддалених географічних розташуваннях забезпечить також додаткову катастрофостійкість.

Рисунок також демонструє використання інтерфейсів на рівні бази даних [69]. Якщо база даних велика і вимагає швидшого резервного копіювання, можна розглянути можливість використання набору компонентів хмарного сховища для зберігання даних [228, 229].

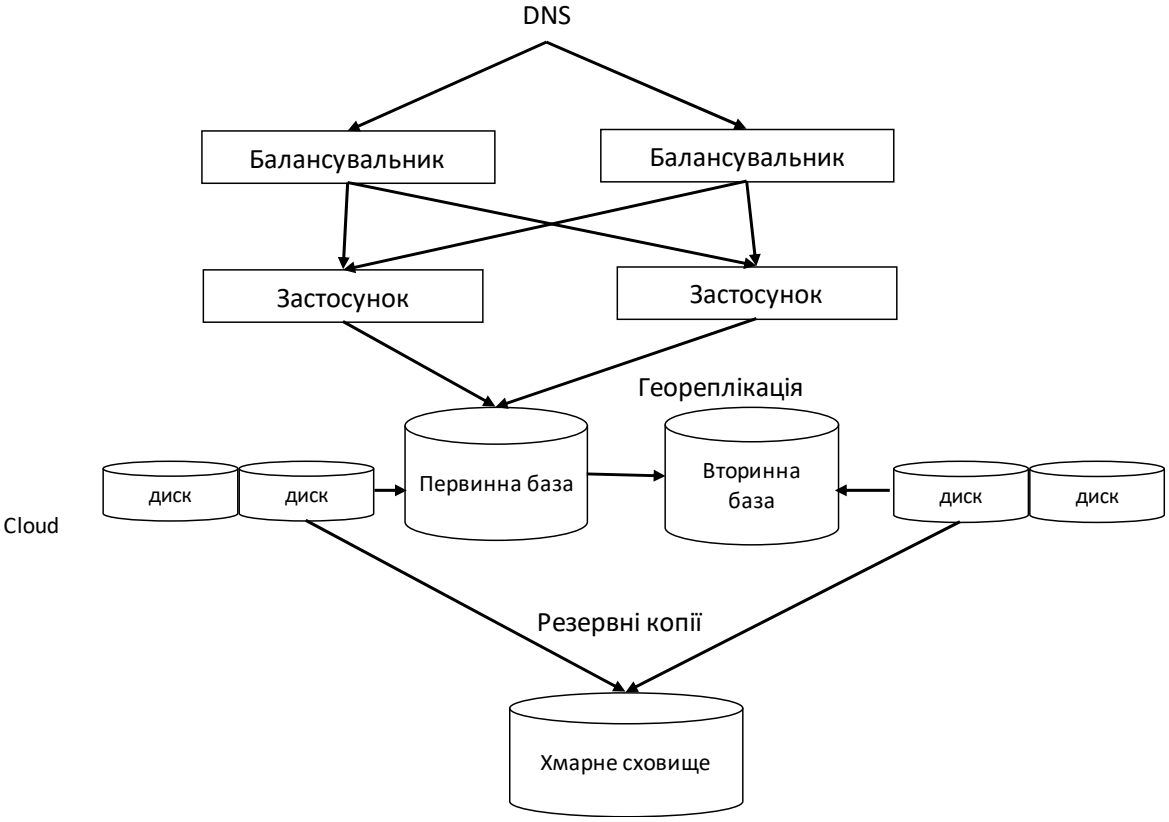


Рис. 6.4. Запропонована трирівнева архітектура хмарної інформаційно-телекомунікаційної системи з дублюванням компонентів та георепліkацією

Незабаром розробники зрозуміли, що для того, щоб повною мірою використовувати цю можливість, їм доведеться змінити архітектуру їх застосунків. Хоча традиційний застосунок може бути розгорнутий в хмарній інфраструктурі як сервіс (IaaS), традиційна трирівнева архітектура не забезпечує гнучкість, необхідну для використання можливостей інфраструктури хмари,

таких як еластичність, самообслуговування та широкий спектр послуг обчислення та зберігання.

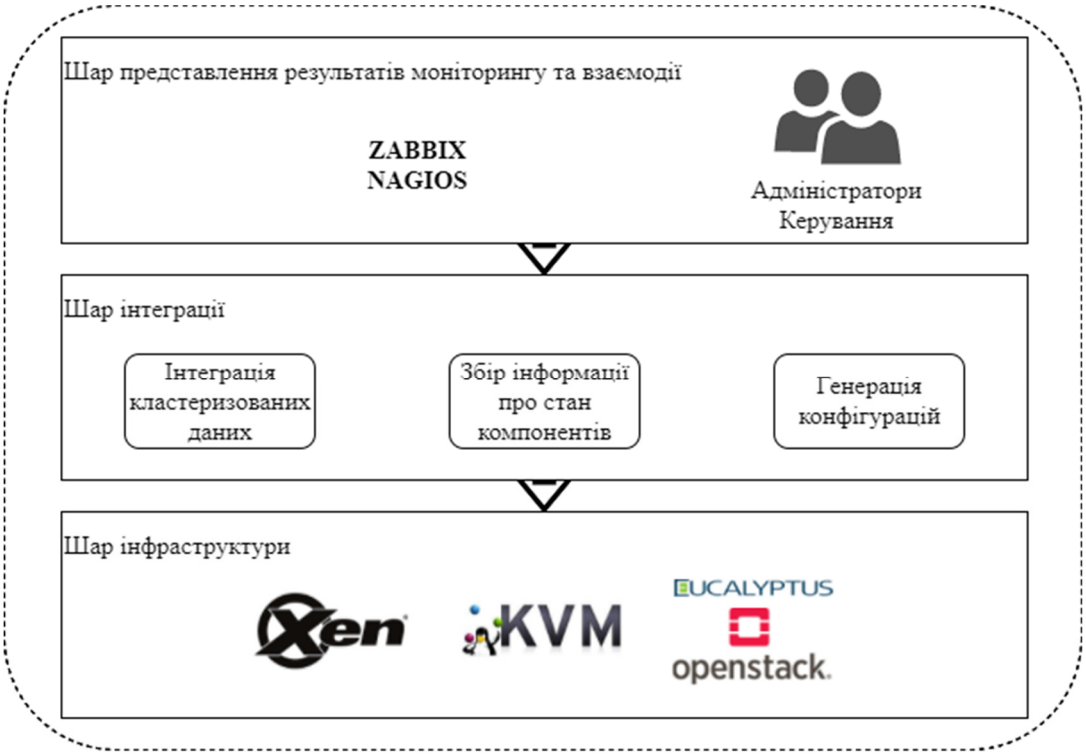


Рис. 6.5. Архітектура системи моніторингу приватного cloud

Розглянемо приклад бази даних, щоб зрозуміти розбіжності між двома архітектурами. Традиційні програми використовують єдину базу даних, яка зберігає всю інформацію застосунку, а потім надає інформацію, що зберігається в ній, для різних його компонентів, коли вона запитується. Коли дані зростають, база даних стає вузьким місцем для більшості застосунків, і, як правило, це призводить до необхідності нарощування апаратних ресурсів сервера БД [101]. Коли аналізуємо архітектуру хмари, то бачимо, що для зберігання даних існує множина різних варіантів (наприклад, об'єкт зберігання, база даних No-SQL, RDB, сховище кешу, сховище даних, сховище архіву даних, сховище Elastic block). Тому замість того, щоб зберігати всі дані в єдиній базі даних, дані можна розділити, виходячи з вимог використання, а потім кожен тип даних може зберігатися в службі хмарного зберігання, яка найкраще відповідає його вимогам. Наприклад, дані, які часто не змінюються, але вимагають ресурсоємних запитів на читання та масштабованість, можуть зберігатися в службі зберігання

об'єктів. Або, якщо застосунок вимагає динамічних зв'язків між об'єктами даних, то можна використовувати базу даних No-SQL. Крім того, якщо в застосунку є кілька великих наборів даних, які вимагають великих обчислень, можна використовувати сервіс Hadoop MapReduce [180].

Щоб використати можливості cloud еластичності, хмарні застосунки повинні мати можливість автоматичного вертикального та горизонтального масштабування, у відповідь на зміни в навантаженні [134]. Крім того, маршрутизація в топології мережі та розташування серверів постійно змінюються. Тому застосунки для хмар розроблені на основі сервісно-орієнтованої архітектури (SOA), де вони виконуються як пов’язані набори сервісів, які не мають строгої прив’язки до інфраструктури, на якій вони працюють. Ці сервіси зареєстровані в сервісному реєстрі при їх запуску, щоб їх могли використати інші сервіси незалежно від їх місцезнаходження. Коли вимоги до навантаження застосунку збільшуються, він здатний динамічно масштабуватися, розгортаючи більше екземплярів сервісів, які зазнають переванаження. Пізніше, коли вимоги до навантаження зменшуються, застосунок вимикає додаткові екземпляри сервісів у цілях економії вартості обчислення та зберігання.

Хмарні застосунки працюють у спільному середовищі, де навантаження іншого орендаря інфраструктури може спричинити зростання тривалості реакції системи. Крім того, оскільки місцезнаходження сервісів не є статичним у сценаріях хмари, зміна локалізації сервісу з однієї мережі в іншу також може викликати зростання затримки. Тому, як правило, хмарні застосунки призначені для контролю затримки. Крім того, збої в інфраструктурі хмари також дуже поширені. Наприклад, мережні перевантаження, затримки запиту, завантаження інтерфейсів вводу-виводу в системах зберігання, аварійні застосунки та обладнання тощо. Популярний шаблон проектування, який використовується для подолання затримок та несправностей, – це черги запитів / відповідей, де запити та відповіді зберігаються в чергах, щоб вони не втрачались. Крім того, затосунки не блокують роботу користувачів, коли реакція відбувається повільно

або коли виникає збій. Скоріше, вони вказують на те, що запит перебуває у процесі оброблення, і користувач продовжує використовувати інші функції програми [218, 219, 227].

У традиційних застосунках розробники вважають, що мережа є відносно безпечною за захищеним центром обробки даних та мережним екраном. Тому вони не шифрують зв'язок застосунку між різними рівнями і з базою даних. У середовищі хмари компоненти додатків розміщуються в різних мережах. Вони знаходяться в умовах багаторазового переміщення, а інфраструктура керується множиною третіх сторін. Тому обов'язковим є шифрування всіх комунікацій в cloud застосунках. В архітектурі хмарних застосунків використовуються три основні принципи для захисту: захист та шифрування даних в режимі транзиту, захист та шифрування даних в режимі очікування та контроль доступу до API.

Моделі розгортання хмарних систем представляють точну копію хмарного середовища і переважно відрізняються власником, розміром та доступом. Вони хактеризують призначення та природу хмари. Більшість організацій готові реалізувати хмару, оскільки це зменшує капітальні витрати та забезпечує контроль над операційними витрати. Щоб дізнатися, яка модель розгортання відповідає вимогам досліджуваного застосунку, необхідно проаналізувати чотири базові моделі розгортання.

Публічна група – це тип веб-хостингу, у якому хмарні сервіси надаються через мережу, відкриту для загального користування. Ця модель є справжнім представленням хмарного хостингу; в ньому провайдер послуг надає послуги та інфраструктуру для різних клієнтів. Клієнти не мають жодної відмінності та контролю над розташуванням інфраструктури. З технічної точки зору може бути незначна різниця між структурою приватних та загальнодоступних хмар, за винятком рівня безпеки, пропонованої для різних послуг, наданих користувачам хмарних хостингових послуг.

Публічна хмара краще підходить для бізнес-цілей, які вимагають керування завантаженням, застосунків, які базуються на SaaS, і керування застосунками, які споживають багато користувачів. Завдяки зменшенню капітальних накладних

витрат та експлуатаційних витрат ця модель є економічно доцільною. Постачальник може надавати послуги безкоштовно або у формі ліцензійної політики, наприклад у якості плати за ліцензію користувача. Вартість ділиться між всіма користувачами, тому загальна хмара приносить додаткові прибутки за рахунок економії на масштабі. Загальнодоступні cloud об'єкти можна використовувати безкоштовно, наприклад – загальнодоступна хмара Google.

Притна хмара також відома як внутрішня хмара. Платформа для хмарних обчислень реалізована в захищеному середовищі із використанням хмарного мережного екрану, який знаходиться під управлінням IT-відділу, який належить до певної корпоративної мережі. Приватна хмара, оскільки доступ дозволено лише авторизованим користувачам, надає організації більший і прямий контроль над своїми даними. Перешкоди, пов'язані з безпекою, можна уникнути у приватній хмарі, але у разі стихійного лиха та внутрішньої крадіжки даних приватна хмара може бути схильною до вразливостей.

Гібридна хмара – це інтегрований тип хмарних обчислень. Вона може бути розташуванням двох або більше хмарних серверів, тобто приватних, загальнодоступних або спільноти хмар, які пов'язані між собою, але залишаються окремими об’єктами. Переваги декількох моделей розгортання доступні в гібридному хмарному хостингу. Це дозволяє користувачеві збільшити пропускну здатність або можливості шляхом агрегації, асиміляції або налаштування за допомогою іншого хмарного пакету / сервісу. У гібридній хмарі ресурси керуються та надаються як власними, так і зовнішніми постачальниками. Це адаптація між двома платформами, в яких відбувається обмін робочими навантаженнями між приватною хмарою та загальною хмарою залежно від потреб.

Ресурси, які не є критичними (розробка та тестові інфраструктури), можуть розміщуватися у загальнодоступній хмарі, яка належить сторонньому постачальнику. Розглянемо у якості прикладу веб-сайт електронної комерції, який розміщений у приватній хмарі, що забезпечує безпеку та масштабованість. Оскільки безпека не є головною проблемою для інформаційної сторінки, вона

розміщується в загальнодоступній хмарі, яка є більш економічною, ніж приватна хмара. Корпоративні клієнти, які більше зосереджують увагу на безпеці та потребують унікальної присутності, можуть реалізувати гібридну хмару як ефективну бізнес-стратегію. Коли виникають проблеми зі збільшенням попиту, додаткові ресурси, необхідні для конкретного застосунку, можуть бути доступними з загальнодоступної хмари. Тут проявляється явище, яке має назву "розрив хмари" і доступне для гібридної хмари.

Організації можуть використовувати гібридну хмарну модель для обробки великих даних. У приватній хмарі організація може зберігати дані про продажі, бізнес та інші конфіденційні дані, і може ініціювати аналітичні запити через загальну хмару, оскільки загальна хмара ефективна для задоволення підвищеного попиту на ресурси. Гібридне хмарне середовище реалізоване за допомогою таких можливостей, як масштабованість, гнучкість та безпека. Якщо вдасться уникнути таких недоліків, як можлива несумісність інтерфейсу застосунків, проблем із підключенням до мережі та підвищених капітальних витрат, гібридна хмара стає оптимальною моделлю розгортання. Тому досліджена у подальшому ІТС у якості моделі розгортання використовує гібридну хмару з хмарними спільнотами. Для цього і реалізовано моделі корпоративного клієнта, яка являє собою приватну хмару, та спільної cloud інфраструктури, яка представляє публічну хмарну систему. Конфігурація системи являє собою поєднання однієї публічної хмари та множини приватних хмар, які утворюють cloud спільноти.

Спільнота Cloud – це тип веб-хостингу, в якому інфраструктура взаємно розподіляється між багатьма організаціями, що належать певній спільноті. Це середовище для декількох орендарів, яке ділиться між кількома організаціями, що належать до певної групи, яка має подібні обчислювальні ризики. Члени спільноти, як правило, поділяють однакову конфіденційність, ефективність та проблеми безпеки. Головний намір цих спільнот – досягти своїх цілей, пов'язаних з бізнесом. Хмара спільноти може бути внутрішньо керованою або нею може керувати сторонний постачальник. Вона може бути розміщена

зовнішньо або внутрішньо по відношенню до інфраструктури організації. Вартість поділяється між певними організаціями всередині спільноти, отже можливо зекономити витрати. Хмара громади підходить для організацій та підприємств, які працюють на спільних інфраструктурах, тендерах або дослідженнях, для яких потрібна централізована хмарна обчислювальна продуктивність для керування, створення та реалізації подібних проектів.

Для забезпечення високої ефективності cloud системи по відношенню до підтримання сталої затримки надання послуг в частині баз даних рекомендовано широке використання кешування. Бази даних в оперативній пам'яті представляють повний або частковий робочий набір даних у системній пам'яті.

Коли всі дані зберігаються в пам'яті, зникає потреба у вирішенні питань, пов'язаних із використанням традиційних обертових дисків. Це означає, наприклад те, що не потрібно зберігати додаткові кеш-копії даних та керувати синхронізацією між ними.

Дані також можуть бути стиснуті та розпаковані у пам'яті легше, що дає змогу заощаджувати місце на еквівалентному екземплярі диска.

Нижче розглянемо компоненти інформаційно-телекомунікаційної системи із застосуванням кешування даних.

Сервер застосунку керує співвідношеннями між запитами користувачів до близького кешу, серверу кешування та до бази даних. Кожен сервер застосунків має власну кеш-пам'ять для останніх запитів. Близький кеш – це перше місце, де шукає інформацію затосунок, потім – сервер кешування і база даних. Сервер кешування встановлюється на машину, відокремлену від вузлів сервера застосунків у кластері. Він доступний для всіх вузлів у кластері. Місцевий кеш існує, головним чином, для окремих машин, де сервер кешування може не бути присутнім. Як і близький кеш, він існує разом з сервером застосунків. Місцевий кеш повинен використовуватися лише для окремих машин або для даних, які не повинні бути доступними для інших вузлів у кластері. Місцевий кеш сервера застосунків не бере участі в синхронізації в кластері.

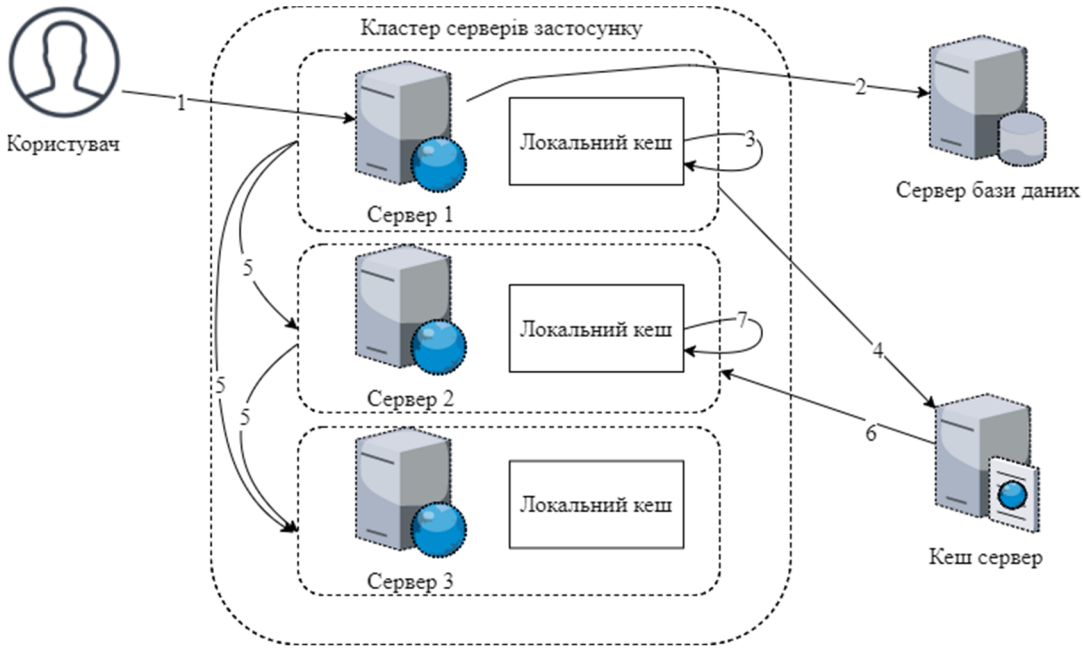
Кластерна система повідомляє про зміни кеш-пам'яті у вузлах сервера застосунків. В результаті, хоча ці дані не повністю реплікуються в вузлах, всі вузли знають, коли вміст їх близького кешу потрібно оновити з серверу кешування або бази даних.

Для типових сценаріїв завантаження контенту дані повертаються з найближчої кеш-пам'яті (якщо дані нещодавно було запитано з поточного вузла сервера застосунків), з кеш-сервера (якщо дані нещодавно було запитано з іншого вузла в кластері) або з бази даних (якщо дані відсутні в кеш-пам'яті).

Дані, отримані з бази даних, поміщаються в кеш так, що подальші завантаження будуть швидшими.

Наведено приклад того, як обробляються зміни в базі даних:

- клієнт вносить зміни, наприклад, оновлення профілю користувача. Їх зміна здійснюється за допомогою сервера 1 кластера через балансувальник навантаження;
- Сервер застосунків записує зміни до бази даних застосунків.
- Сервер застосунків вузла перетворює щойно змінені дані в його найближчий кеш для швидкого пошуку пізніше.
- Сервер застосунків вузла перетворює щойно змінені дані на сервері кеш-пам'яті, де вони будуть знайдені іншими вузлами кластера.
- Сервер 1 повідомляє системі кластеризації про зміну вмісту його близького кешу, проходячи уздовж списку змінених елементів кешу. Система кластеризації збирає звіти про зміни і регулярно надсилає їх у пакетному режимі іншим вузлам у кластері. У близькій кеш-пам'яті на інших вузлах обробляються будь-які записи, що відповідають тим, які містяться в списку змін.
- Коли сервер застосунків 2 отримує запит на дані, які було змінено та які він видалив з його найближчої кеш-пам'яті, він здійснює пошук на сервері кеш-пам'яті.
- Сервер 2 кешує свіжі дані у власній кеш-пам'яті.



1 – запит користувача; 2 – отримання необхідної інформації з БД; 3 – локальне кешування отриманої з БД інформації; 4 – запис отриманої з БД інформації на кешуючий сервер; 5 – кластерна система здійснює синхронізацію локальних кешів; 6 – система кешування оновлює вміст локальних кешів із кешуючого сервера; 7 – аналогічно пункту 3

Рис. 6.6. Підвищення ефективності функціонування розподіленої інформаційно-телекомунікаційної системи шляхом локального кешування інформації з бази даних

6.2. Дослідження доступності cloud сервісів

Сьогодні віртуалізація набирає все більшої популярності за рахунок свої переваг, зокрема можливості запуску віртуальних машина на різного роду серверах динамічно залежно від того, як змінюється інтенсивність вхідного навантаження від користувачів, а також можливість міграції віртуальної машини між серверами як одного, так і багатьох центрів обробки даних. Тому не дивно, що ця технологія лягає в основу практично всіх сервісно-орієнтованих інформаційних систем, які проектуються сьогодні, а також центрів обробки даних [33].

Кожен сервіс можна розбити на дрібніші логічні функціональні компоненти, які можуть виступати як окремі сервіси, що встановлені на тій чи нішій

віртуальній машині [108]. Припустимо, що обсяг обчислювальних ресурсів, потрібен для всіх компонентів одного і того ж додатку є однаковим. В умовах відсутності механізмів вертикального масштабування ресурсів компоненту сервісу та динамчіної зміни інтенсивності навантаження, що надходить від користувачів, може виникнути перевантаження одного з помпонентів і як наслідок блокування роботи усього сервісу.

Доступність характеризує сервіс на основі імовірності блокування, отже щоб підвищити доступність необхідно забзпечити мінімальну імовірінсть блокування всіх компонентів сервісу. Для сервісів, побудованих за принципами сервісно-орієнтованої архітектури та які виконуються в середовищі центрів обробки даних, у робьоті пропонукється використовувати метод визначення ефектоивної доступності сервісу у режимі групового пошуку. Необхідно ввести ряд наступних обмежень та позначень:

- нехай j – це кількість окремих компонентів програмного сервісу. Кожен компонент може обслуговувати тільки один запит за раз, а тому буде недоступним для інших сервісів;
- компонент блокується на період обслуговування запиту;
- один компонент відповідає одній віртуальній машині, як наслідок на одній віртуальній машині не може бути встановлено більше ніж один компонент будь якого сервісу;
- максимальна кількість віртуальних машин становить N, що можуть бути розгорнуті на одному фізчиному сервері;;
- всі сервіси розділені на однакову кількість функціональних компоненті Z.

Враховуючи зазначені обмеження та позначення, доступність сервісів можна оцінити з використанням класичної моделі багатоланкових систем, що відобрадена на Рис. 6.7.

Основними математичними позначеннями, що використовуються у зазначеному методі є [80]:

- середня доступність d_{sr};
- середня недоступність $\overline{d_{sr}}$;

– максимальна доступність d_{max}.

Для того, щоб обчислити ймовірність втрат, необхідно спочатку використовуючимодифіковану формулу Пальма-Якобеуса визначити ефективну доступність d_{ef}. Згадана формула дає змогу визначити зайняття з'єднувальних ліній в неповнодоступному пучку. По іншому зайняття ліній можна визначити використовуючи розподіл Ерланга, який був отриманий в повнодоступному пучку для будь-яких ліній.

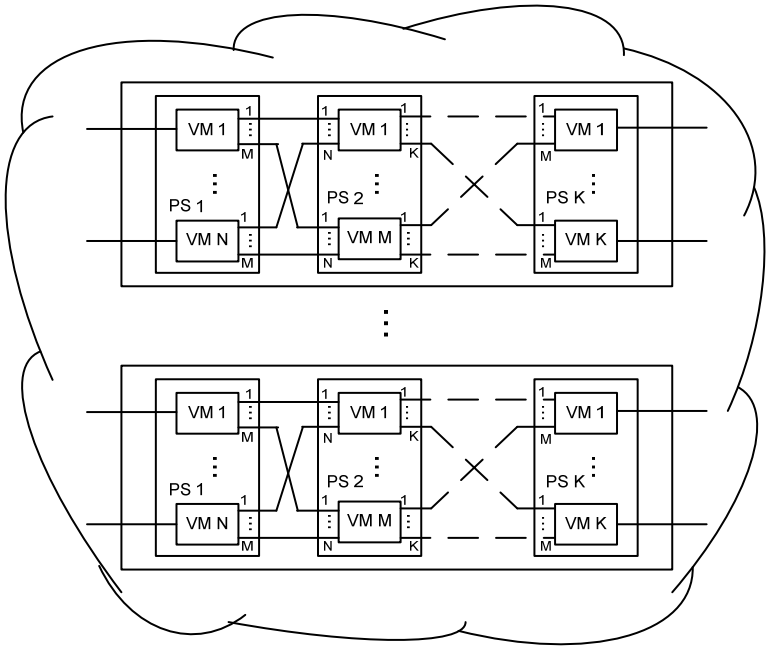


Рис. 6.7. Модель розподіленого сервісу на основі сервісно-орієнтованої архітектури з розбиттям на компоненти

Описаний метод можна використовувати для оцінки доступності складних розподілених додатків у середовищі центрів обробки даних. Для прикладу приймемо наступні умови:

– фізична інфраструктура складається з 5 фізичних серверів;
– кожен фізичний сервер не може містити більше ніж 4 віртуальні машини одночасно.
– кожен фізичний сервер може містити компоненти сервісу тільки одного типу (Рис. 6.7) і формує ланку у зазначеному методів.

Процес обслуговування запитів відбувається наступним чином. Запит обслуговується кожною ланкою по черзі. Після обслуговування певною лакою запит перенаправляється до компоненту в наступнй по порядку ланці, за умови що компонент є доступним. Варто зазначити, що при надходженні запиту відбувається наскрізне резервування усього сервісу, тобто усіх компонетів сервісу в усіх ланках, необхідних для обробки цього запиту. Це означає що ці компоненти не можуть бути використані іншими сервісами для обслуговування запитів. Описаний процес відображений на Рис. 6.8.

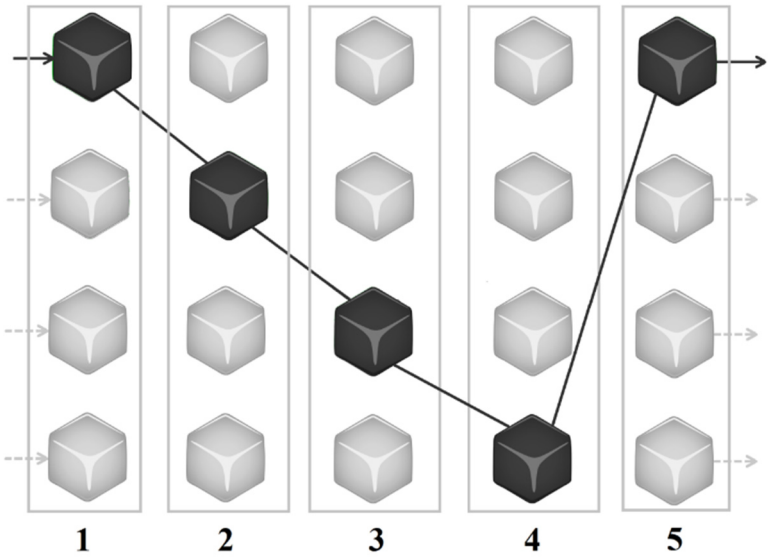


Рис. 6.8. Процес обслуговування запиту послідовністю сервісів

Для подальшого дослідження функціонування методу в середовищі центру обробки даних необхідно ввести наступні позначення:

- m_i – число логічних з'єднань між віртуальними машинами в ланках i та i+1 (у роботі приймається, що $m_i = 4$);
- k_i – число віртуальних машин на сервері *i*;
- s – число фізичних серверів в центрів обробки даних;
- q_r – число лоігчних з'єднань між віртуальною машиною, що містить останній компонент сервісу, та користувачем, що відправив запит (у роботі приймається, що $q_r = 4$);
- v_r – число лоігчних з'єднаннь між вірутальною машиною та користувачем (у роботі приймається, що $v_r = 1$);

– M – кількість усіх виходів (у роботі приймається, що M=4);
– $v_{i,i+1}$ – кількість усіх з'єднань між послідовними компонентами з двох сусідніх ланок.

Для оцінки доступності сервісів використаємо наступні позначення:

– середнья доступність сервісу d_{sr} ;
– середня недоступність;
– максимальна доступність сервісів системи від кількості запитів на сервіс, що генеруються користувачами d_{max};
– ймовірність втрат запитів.

Таким чином, для визначення середньої доступності сервісів використаємо наступні формулу:

$$d_{sr_k} = \left[\frac{m_i - y_k}{m_i}\right]^{s-1}$$

де y_k – навантаження від користувачі, що обслуговується компонентом в ланці k. Зрозуміло, що така система не зможе обслужити навантаження, що перевищує один Ерланг. В іншому випадку, це буде означати, що абсолютно всі виходи компонента будуть зайняті. Оскільки всі компоненти, необхідні для обслуговування в рамках одного сервісу, резервуються при надходженні одного запиту, то при максимальній інтенсивності навантаження, компоненти сервісу будуть постійно зайняті і як наслідок недоступні для ніших серсісів. У роботі приємається, що інтенсивність надходження запитів рівна 100 запитів/год.

Для визначення максимальної доступності необхідно використати настпний вираз:

$$d_{max} = \prod_{i=1}^{s-1} \frac{m_i}{m_{i+1}}.$$

Суть виразу полягає у знаходженні добутку між кількістю зв'язків між віртаульними машинами ланок і та i+1, та віртуальними машинами ланок i+1 та i+2. Кожен сервер містить однакове число компонентів, а тому кількість зв'язків між ланкми і та i+1 буде такою самою як між ланками i+1 та i+n, що в результаті призведе до $d_{max} = 1$.

Для визначення середньої недоступності слід використовувати наступний вираз:

$$\overline{d_{sr}} = d_{max} - d_{sr}$$

Отримане значення відображає середню кількість недоступних компонентів.

Ефективна доступність визначається за домопогою наступного виразу:

$$d_e = d_{1e} + d_{2e}$$

де $d_{1e} = d_{sr} q_r$ відображають середню кількість з'єднаннь між віртуальниими машинами в останній ланці та користувачами. Для визначення другого доданку в попередньому виразі слід використовувати наступний вираз:

$$d_{2e} = \overline{d_{sr}} * q_r * \frac{y_{0r}}{v_r} * \frac{v_{s-1,s} - y_0}{M}$$

де y_{0r} – кількість опрацьованих запитів; y_0 – кількість усіх запитів, що надійшли.

Використовуючи аналітичні та статистичні методи розрахунку, у роботі отримано наступні результати доступності (Таблиця 6.1).

Таблиця 6.1. Результати проведених розрахунків [34]

Опрацьовані запити і-тою ланкою, зап/год	Середня доступність системи, %	Середня недоступність системи, %	Ефективна доступність системи
100	0.422	0.578	20.188
120	0.343	0.657	22.396
140	0.275	0.725	24.31
160	0.216	0.784	25.952
180	0.166	0.834	27.341
200	0.125	0.875	28.5
220	0.091	0.909	29.448
240	0.064	0.936	30.208
260	0.043	0.957	30.799
280	0.027	0.973	31.244
300	0.016	0.984	31.563
320	0.008	0.992	31.776
340	0.00375	0.997	31.905
360	0.001	0.999	31.972

Опрацьовані запити і-тою ланкою, зап/год	Середня доступність системи, %	Середня недоступність системи, %	Ефективна доступність системи
380	0.000125	1	31.996

Результати, отримані вище, дають змогу зробити наступні висновки:

- кількість копій компонентів одного сервісу впливає прямопропорційно на доступність усього сервісу;
- навантаження, що створюється на сервіс дорівнює сумі запитів, що надходять від усіх користувачів протгом визначеного інтревалу часу;
- кількість опрацьованих запитів та з'єднаннь між сусідніми ланками впливає прямопропорційно на ефективну доступність сервісів.

Залежність між середньою доступністю сервісу та опрацьованим навантаженням відображена на Рис. 6.9.

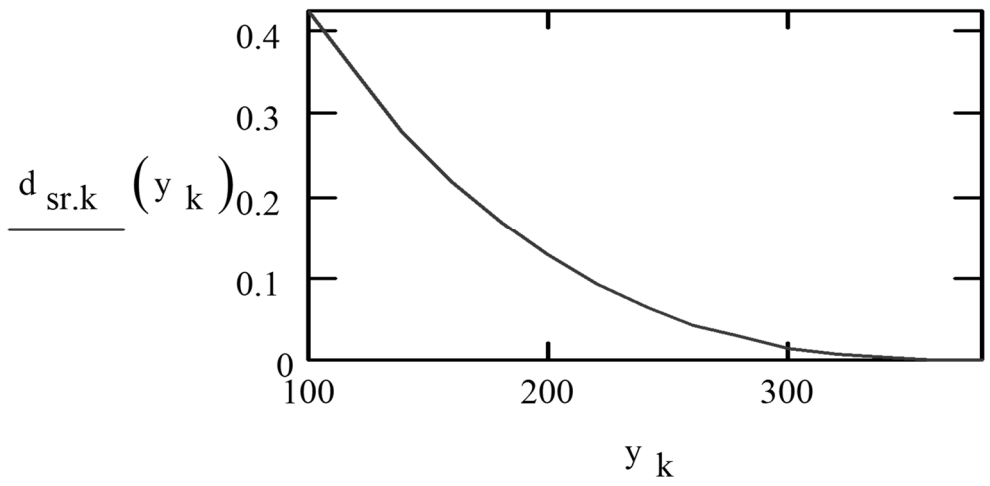


Рис. 6.9. Залежність між середньою доступністю сервісу та опрацьованим навантаженням

Аналізуючи вищенаведений, можна зробити висновок, що інтенсивність навантаження є обернено-пропорнціною до середньої доступності кожної компоненти, що безпосереньо впливає на доступність сервісу в цілому. Зрозуміло, що кількість серверів у таких ситуація дасть змогу підвищити доступність середовища в цілому (Рис. 6.10).

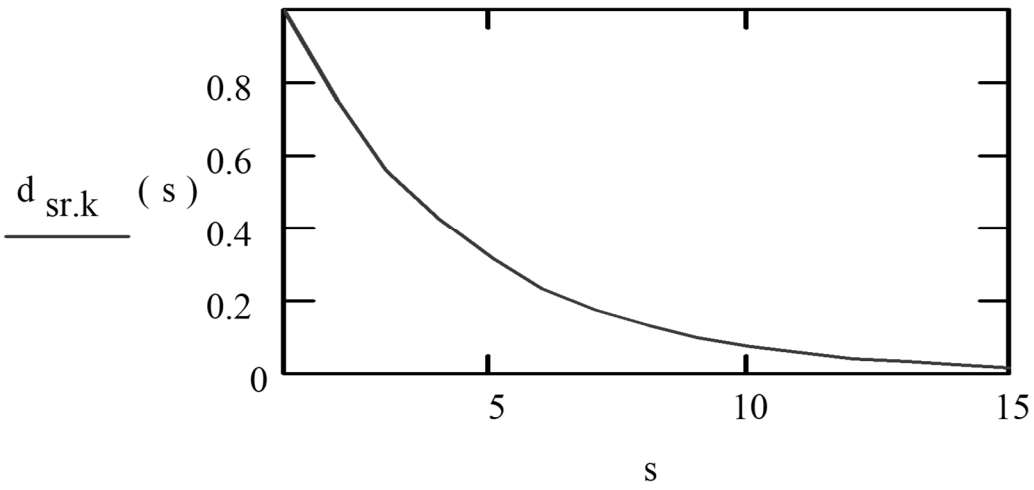


Рис. 6.10. Залежність значень середньої доступності від кількості серверів

Отже, середня доступність окремої компоненти сервісу в будь якій ланці є обернено-пропорціною до кількості ланок та інтенсивності навантаження, що надходить в систему.

Поняття доступність розподілених програмних сервісів відповідає традимційному поняттю доступності в телекомунікаційних системах. Проте на відміну від традиційних систем, розподілені сервіси виконуються у середовищі центрів обробки даних, де окрім власної інформаційної інфраструктури, виконується й інші сервіси та служби, які спільно використовують ресурси серверів та мережних каналів. Кожен сервіс має власну політику безпеки та договір про якість надання послуг тим чи нішим користувачам. Відповідно поняття доступності відповідає тривалості очікування споживачем відповіді від сервісу з урахуванням усіх особливостей його функціонування та умов договору про якість надання послуг. У сервісно-орієнтованих системах, прикладом яких є система описана у роботі, доступність характеризує наявність ресурсів (віртуальних каналів чи обчислювальних потужностей), використання яких дає змогу опрацювати запит користувача протягом часу, що визначений сервісним договором. Це поняття також включає в себе відмовостійкість сервісу, оскільки у такому випадку тривалість очікування відповіді прямує до незкінченності .

Взявши за основу поняття доступності в сервісно-орієнтованих системах, поняття ефективної доступності охоплює всі вільні ресурси, які існують між

двома послідовними компонентами сервісу, що залежить від інтенсивності опрацювання вхідного навантаження (Рис. 6.11).

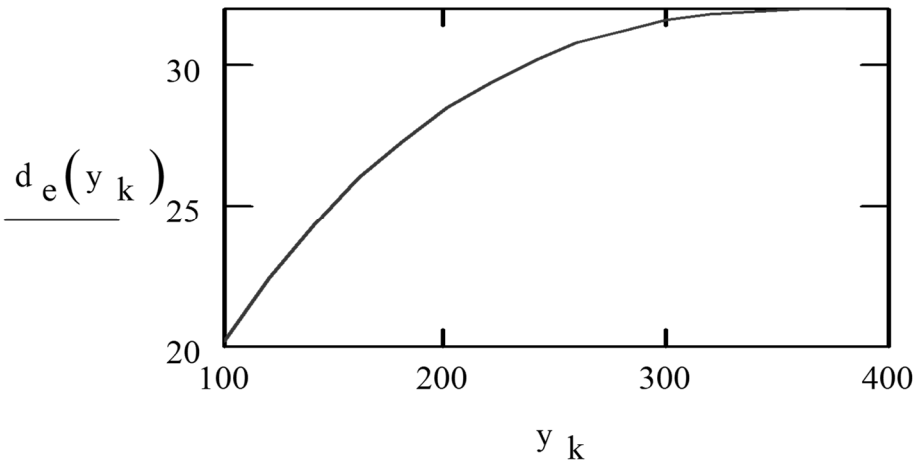


Рис. 6.11. Залежність ефективної доступності від швидкості опрацювання запитів

Залежність на Рис. 6.11 дає змогу зробити висновок, що система реактивно реагує на підвищення інтенсивності вхідного навантаження та збільшує обсяг мережних та обчислювальних ресурсів для того, щоб не зросла тривалість опрацювання запитів та не погіршилася якість обслуговування сервісу.

Зрозуміло, що інтенсивність вхідлного навантаження не є єдиним фактором, що впливає на доступність сервісів. Це також і розмір фізичної інфраструктури а також потужність кожного фізчиного сервера окремо. Саме фізична інфраструктура є вирішальним фактором при забезпеченні високої доступності сервісів, оскільки дає змогу проводити багаторазову реплікацію та гнучке балансування навантаження з урахування завантаження кожного сервера. Це дає змогу не тільки підвищити якість осблугвоування користувачів, але і збільшити кількість сервісів, які можуть бути паралельно розгорнуті на одній і тій же ж фізичні йінфрастуктурі. Чим більша кількість реплікованих компонентів, тим більша кількість логічних з’єднаннь, а отже вища доступність сервісу.

На Рис. 6.12 відображено ефективної доступності системи, що залежить від кількості фізичних серверів (ланок).

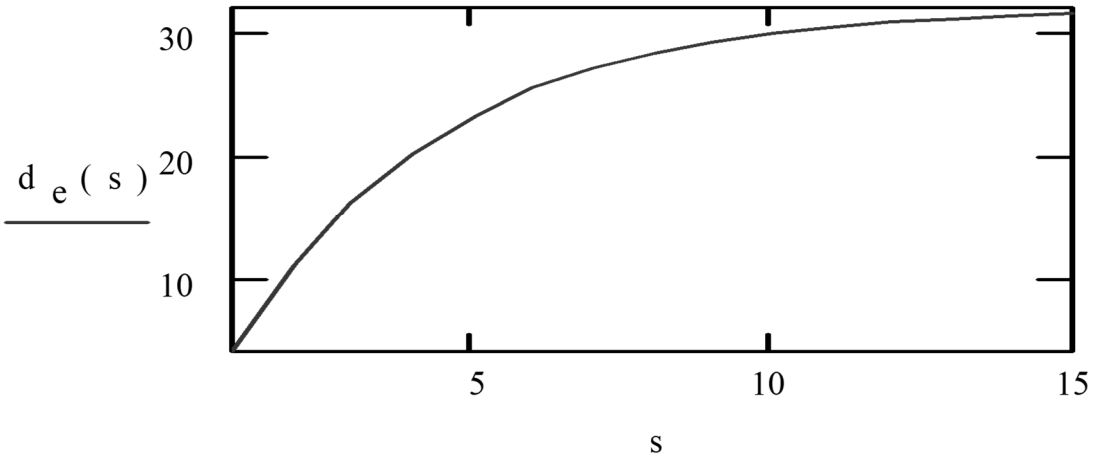


Рис. 6.12. Залежність між ефективною доступністю сервісу і кількістю фізичних серверів у ІКС

Чим більша кількість фізичних серверів у центрі обробки даних тим більша ефективна доступність, оскільки більше число реплікацій одного і того ж сервісу може бути створене для обслуговування зростаючого навантаження.

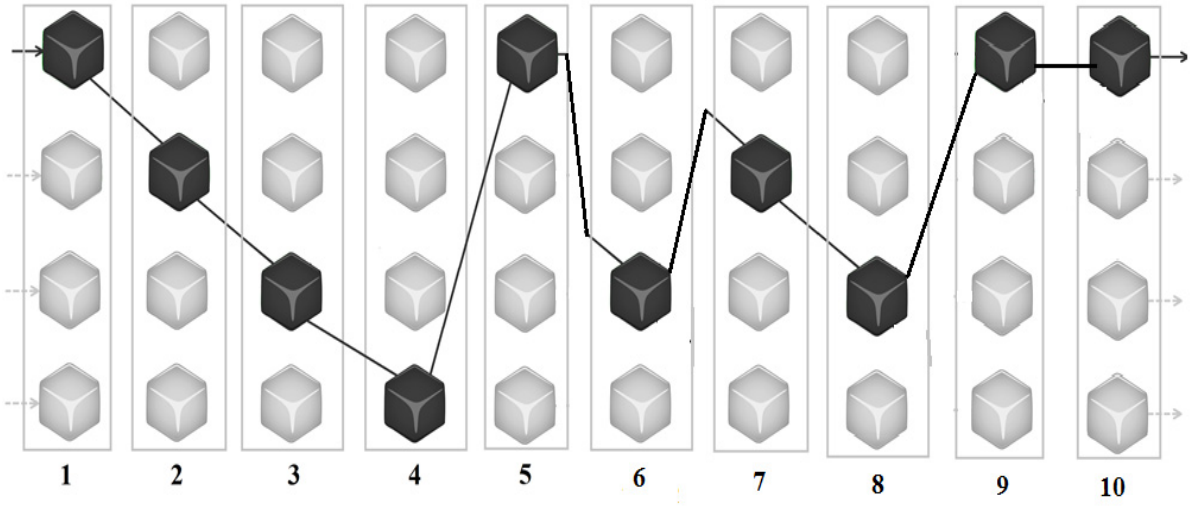


Рис. 6.13 Реплікація сервісів для підвищення достіпності та якості обслуговуання сервісів

6.3. Навантажувальне тестування інформаційно-комунікаційної системи, побудованої на базі cloud сервісу «платформа як сервіс»

6.3.1. Завдання та цілі навантажувального тестування інформаційно-комунікаційних систем

У процесі побудови та надання хмарних застосунків клієнтам постачальнику дуже важливо розуміти, яку кількість одночасних користувачів може підтримувати система за умови її використання з урахуванням різних сценаріїв [266]. Враховуючи, що будь-яка зміна програмного коду або ж інфраструктури впливає на цю кількість, подекуди дуже важко або й неможливо коректно змоделювати поведінку такої системи. Тому отримання оцінки максимальної кількості користувачів проводять експериментальним шляхом на основі навантажувального тестування.

Серед основних завдань навантажувального тестування інформаційно-комунікаційних систем слід виділити такі:

- Встановлення максимальної кількості одночасних користувачів у системі за різних сценаріїв використання cloud застосунку.
- Виявлення вузьких місць у системі.
- Встановлення компонентів системи, які потребують вдосконалення або заміни з метою збільшення ємності системи.
- Виявлення нерегулярних відмов та помилок, які їх спричиняють.
- Визначення прийнятного рівня помилок, за якого система може бути передана у використання клієнтам.

Оскільки навантажувальне тестування за своєю природою є експериментальним процесом, то перед його початком слід спланувати цей експеримент, визначити ключові точки навантажувального тестування, тобто елементи системи, які підлягають перевірці, визначити тривалість, хід, параметри ітерацій та інші характеристики експерименту.

Окремо слід реалізувати програмно розроблений алгоритм експерименту, оскільки йдеться про навантажувальне тестування розподілених програмних систем.

6.3.2. Конфігурування параметрів системи у процесі навантажувального тестування

У роботі здійснено реалізацію системи навантажувального тестування, яка орієнтована на роботу з інформаційно-комунікаційною системою корпоративного рівня, розрахованою на обслуговування не менше 5 тисяч користувачів по всьому світу. Слід відзначити, що комплекс засобів навантажувального тестування є окремим програмним продуктом, який розробляється конкретно для тестування певної інформаційно-комунікаційної системи, оскільки сам набір тестів та умови їх використання безпосередньо залежать від кінцевої системи. Система навантажувального тестування лише дає змогу зімітувати одночасну роботу кінцевих користувачів за певними сценаріями.

Навантажувальне тестування завжди організовують перед передаванням інформаційно-комунікаційної системи у користування клієнтам. Цей процес гарантує, що у реальних умовах експлуатації системи її поведінка буде відповідати очікуваній, а якість сприйняття буде знаходитися у заданих межах незалежно від кількості одночасних користувачів до досягнення певного порогу, який і є ємністю систми. Варто відзначити, що для різних сценаріїв поведінки користувачів ємність системи може різнитися. Тому за ємність системи приймають найменше із отриманих значень.

Навантажувальне тестування дає змогу визначити показники, які можуть бути закладені в угоди про рівень сервісу (SLA), які укладаються між клієнтом та постачальником cloud послуги.

Конфігурування системи в процесі навантажувального тестування слід здійснювати з метою калібрування параметрів системи. Дуже часто таке калібрування дає змогу досягти планованої ємності системи без необхідності

вносити зміни на рівні програмного коду. Проте, це можливо лише за умови незначних відхилень реальної поведінки системи від очікуваної.

Якщо ж ємність системи, отримана в результаті навантажувального тестування, суттєво відрізняється від планованої, то без змін програмного коду неможливо досягти необхідної ємності системи.

6.3.3. Метрики навантажувального тестування

У процесі навантажувального тестування одним із етапів є збір метрик, які характеризують інфраструктуру cloud системи, а також метрик, які характеризують якість сприйняття користувача. До першої групи належать: завантаження апаратних ресурсів віртуальних машин, які забезпечують хостинг застосунків (процесора, оперативної пам’яті тощо), завантаження системних та користувацьких баз даних (за обсягом переданих на вхід/вихід даних, завантаженням процесора, оперативної пам’яті тощо), завантаження інших системних компонентів, наприклад редіс кешу за кількістю активних з’єднань. До другої групи метрик належать тривалість відгуку сервера, тривалість завантаження веб-сторінки, кількість запитів, на які сервер відповів з помилкою, що спричинила відмову в обслуговуванні запиту.

Моніторинг метрик навантажувального тестування, які належать до першої групи, може здійснюватися через інтерфейс користувача, який надається постачальником cloud сервісу.

Моніторинг метрик навантажувального тестування, які належать до другої групи, здійснюється в середовищі запуску програмного продукту, який реалізує функціонал навантажувального тестування.

6.3.4. Ітеративний процес навантажувального тестування

Оскільки зміни у програмному коді впливають на результати навантажувального тестування, то цей процес має ітеративний характер. Ітерацією навантажувального тестування вважаємо кожен цикл виконання цього процесу після необхідного запланованого обсягу змін в програмному коді. На підставі двох послідовних ітерацій процесу навантажувального тестування

формується порівняльний звіт, який в відсотковому співвідношенні дає змогу оцінити покращення / погіршення метрик другої групи.

Паралельно після збору метрик першої групи слід здійснити порівняльний аналіз для отримання повної картини щодо зміни поведінки інформаційно-комунікаційної програмної cloud системи під впливом коливань користувацького навантаження.

На підставі отриманих порівняльних оцінок є можливість прийняти рішення про готовність продукту до передавання у користування реальними клієнтами, чи, навпаки, є потреба у зміні програмного коду з метою забезпечення відповідності планованій ємності системи із заданим рівнем помилок, що спричиняють до відмови у процесі обробки запиту.

Слід зазначити, що програмний продукт, який реалізує сценарії навантажувального тестування, є лише інструментальним засобом, який сам по собі не дає повної об’єктивної оцінки поведінки системи за умови, що він не охоплює всі можливі сценарії поведінки системи.

6.3.5. Результати навантажувального тестування інформаційно-комунікаційних систем

Параметри та характеристики навантажувального тестування можуть бути зведені до таблиці для кращого представлення.

Спільна cloud інфраструктура на платформі Microsoft Azure (в назвах сервіс планів “Standard” сервісний рівень, який надається на сервера, цифра після нього означає кількість серверів, а слова «Medium» або «Large» визначають рівень конфігурації):

- Сервіс план 1 Standard 1 Medium (сервер, на якому розміщено частину системний застосунок)
- Сервіс план 2 Standard 4 Medium (сервер, на якому розміщено клієнтські веб-застосунки)
- Сервіс план 3 Standard 8 Large (сервер, на якому розміщено веб-застосунок безпеки)

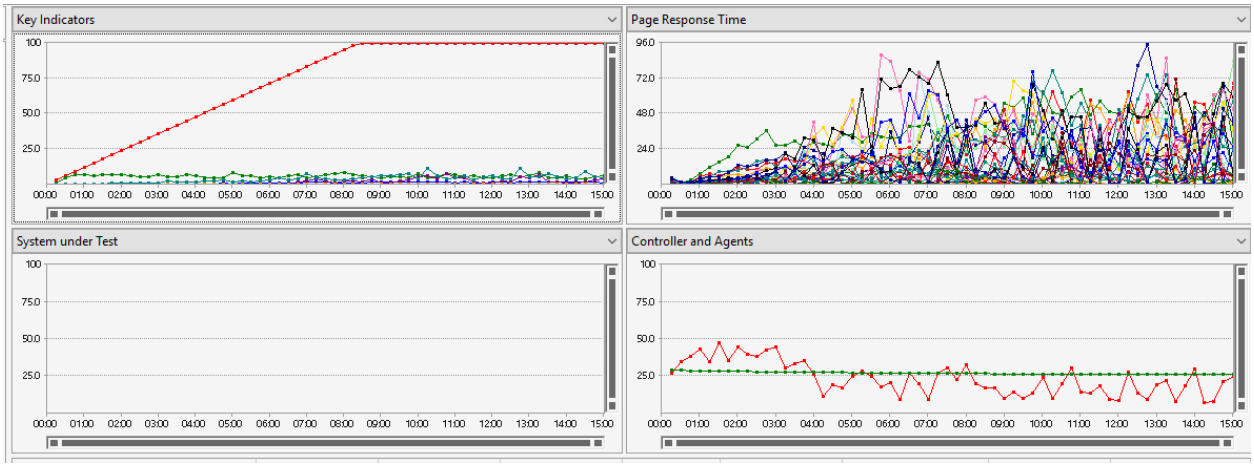
– системна база S0
– база безпеки S2
– база клієнта S3 (MaxPoolSize=1000)
– Редіс кеш C1 Standard

Таблиця 6.2. Параметри процесу навантажувального тестування досліджуваної розподіленої інформаційно-телекомунікаційної системи

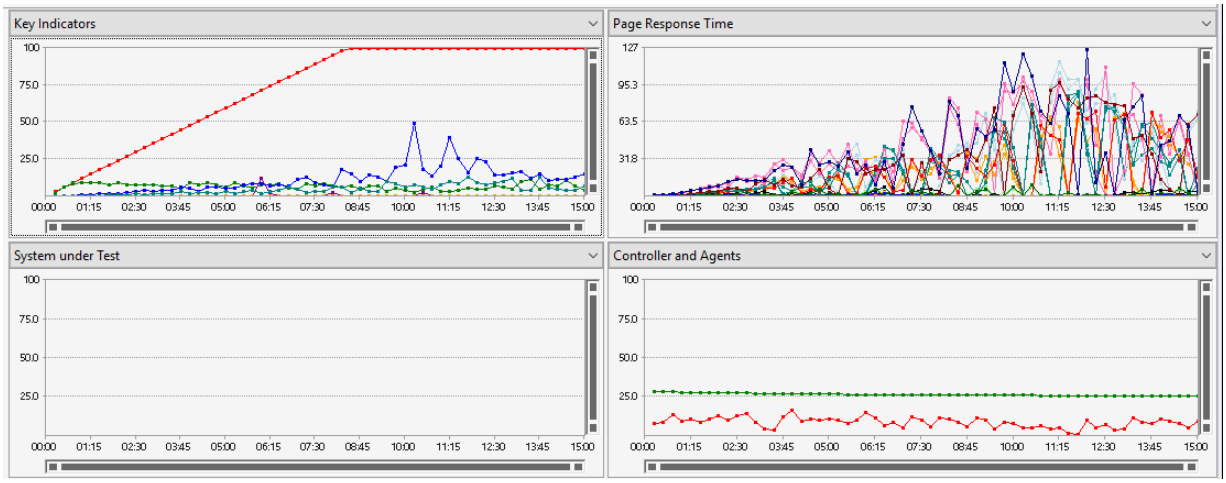
№ з/п	Максимальна кількість користувачів у системі	Тривалість процесу,	Початкова кількість віртуальних користувачів	Крок збільшення, с	Обсяг збільшення користувачів	Холодний старт	Середній час відгуку сервера, с	Максимальний час відгуку сервера, с	Масштабування системи	Збої зафіксовано на кількості .	Статус експерименту
1	5000	15	10	5	10	Ні	3,66	11,7	Реалізовано вручну	5000	Невдача
2	5000	15	10	5	10	Ні	3,71	12,8	Реалізовано вручну	5000	Невдача

Проведено два експерименти, у яких змінювався лише програмний код самої досліджуваної системи. Конфігурація спільної інфраструктури, інфраструктури клієнта та програмної системи навантажувального тестування залишалася незмінною. Отримані результати демонструють, що досліждення поведінки такого класу систем неможливе із використанням аналітичних моделей, чи навіть імітаційних моделей, оскільки немає змоги врахувати природу, складність та зміни програмного коду на всіх рівнях трирівневої архітектури cloud застосунку із резервуванням.

Нижче наведено порівняння результатів двох експериментів.



експеримент 1



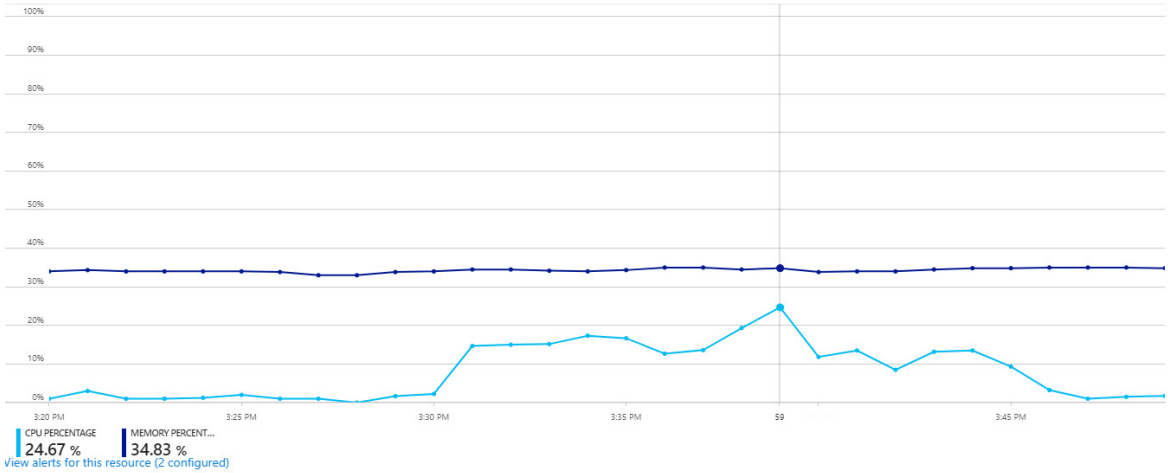
експеримент 2

Рис. 6.14. Результати навантажувального тестування системи, яка реалізована на основі моделей спільної cloud інфраструктури та корпоративного клієнта

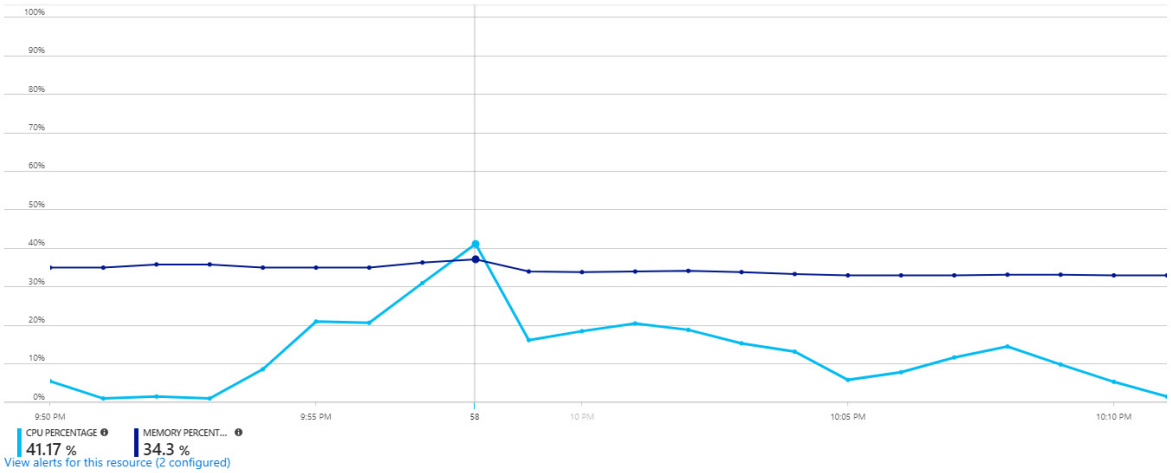
Перша чверть Рис. 6.14 ілюструє динаміку зростання кількості користувачів у системі (у відсотках від максимального значення), друга чверть ілюструє тривалість завантаження сторінок у веб-браузері клієнта при надсилання запитів на різні компоненти інформаційно-комунікаційної системи, третя чверть відображає, що процес тестування завершено, а остання чверть ілюструє завантаження машини, з якої здійснювався запуск процесу навантажувального тестування. Варто відзначити, що загальне задекларованe навантаження створювалося з 5 незалежних машин, оскільки процес навантажувального тестування активно використовує обчислювальні ресурси у процесі формування

значної кількості запитів на різні компоненти інформаційно-телекомунікаційної системи.

Нижче наведено рисунки, які демонструють реакцію інфраструктури досліджуваної інформаційно-телекомунікаційної системи на процес навантажувального тестування. З представлених графіків легко бачити періоди часу, в які система зазнавала зростання навантаження. На Рис. 6.15 наведено ступінь використання ресурсів процесора та оперативної пам'яті Сервіс плану 1. Бачимо, що завантаження знаходиться в допустимих межах (менше 70%) навіть за максимальної кількості користувачів, які одночасно використовують систему.



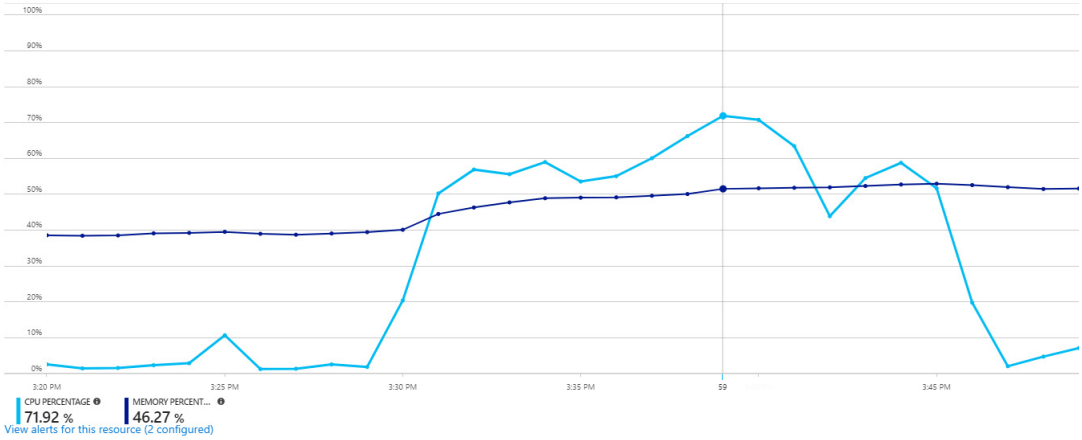
експеримент 1



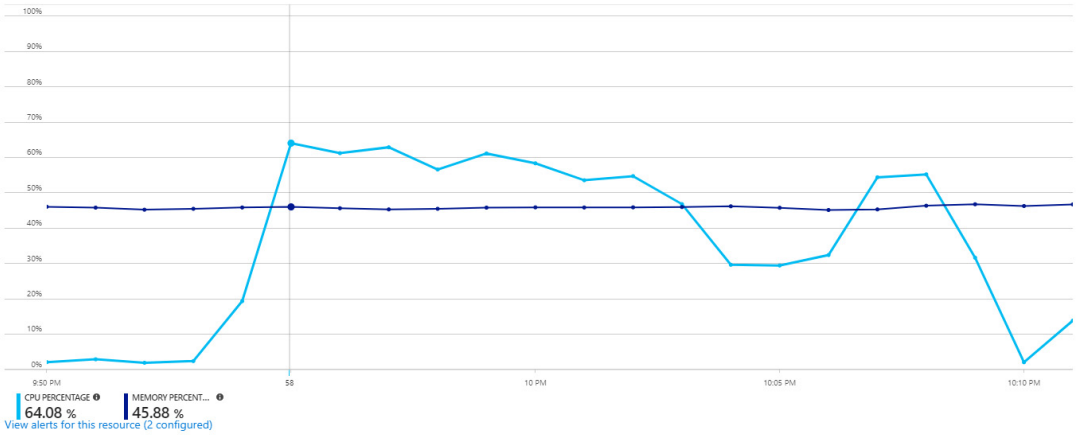
експеримент 2

Рис. 6.15. Завантаження ресурсів Сервіс плану 1

Наступні графіки відображають поведінку Сервіс плану 2 (Рис. 6.16), який відповідає за автентифікацію користувачів в системі. Бачимо, що попри значний обсяг ресурсів, виділених для цього сервісу, їх завантаження перевищує 70 %, що має розглядатися як необхідність вдосконалення системи.



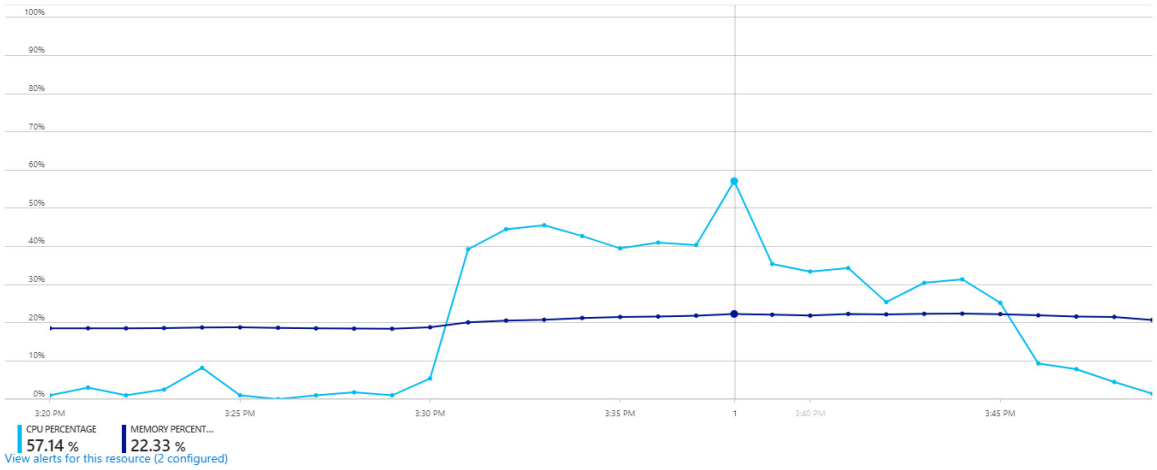
експеримент 1



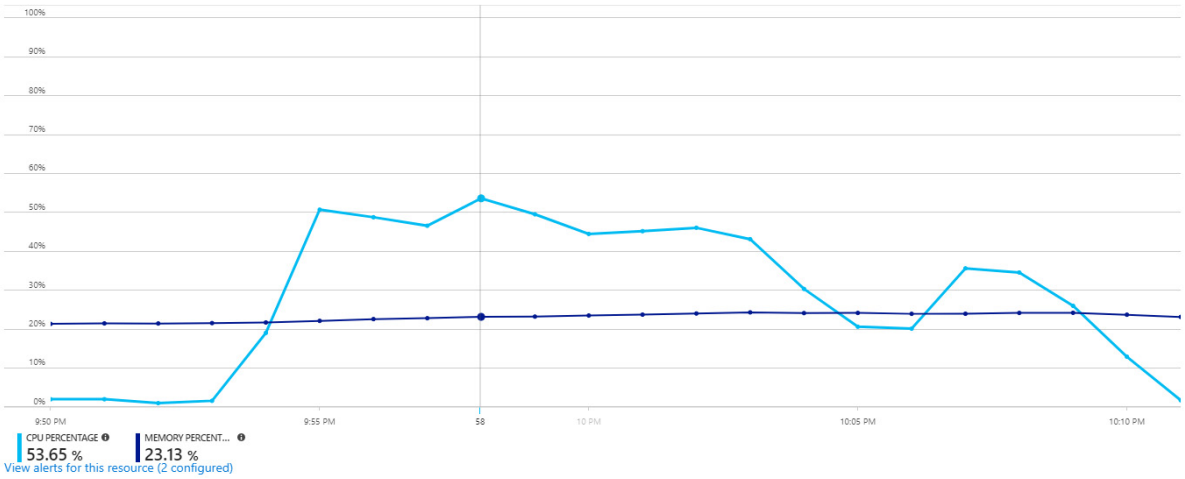
експеримент 2

Рис. 6.16. Завантаження ресурсів Сервіс плану 2

Наступні графіки відображають реакцію Сервіс плану 3 на процес навантажувального тестування. Варто відзначити, що для цього компоненту попередньо підібрано таким чином, щоб було дотримано допустимих показників завантаження ресурсів системи. На Рис. 6.17 добре видно, що, хоча максимальне значення завантаження відрізняється і не суттєво, значним чином змінюється поведінка системи під навантаженням, що зумовлено змінами у програмному коді.



експеримент 1



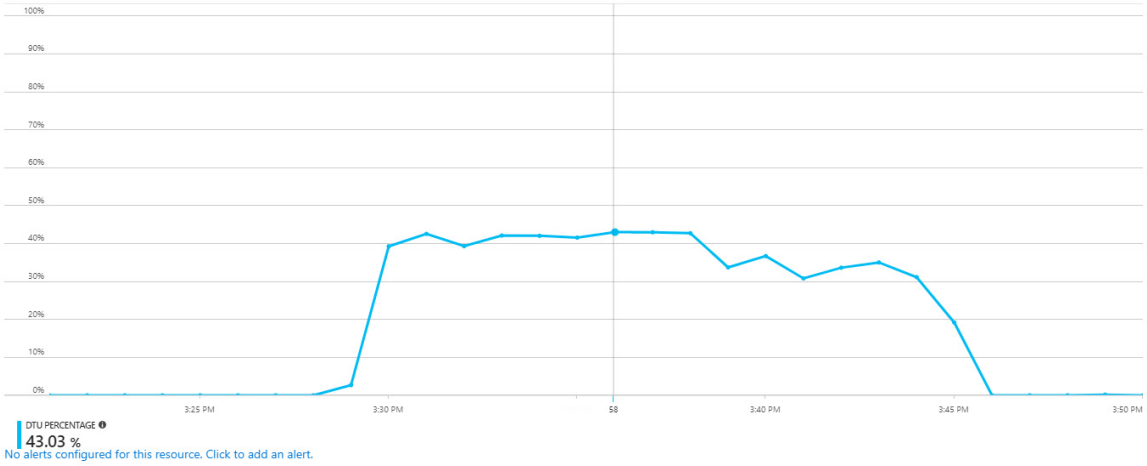
експеримент 2

Рис. 6.17. Завантаження ресурсів Сервіс плану 2

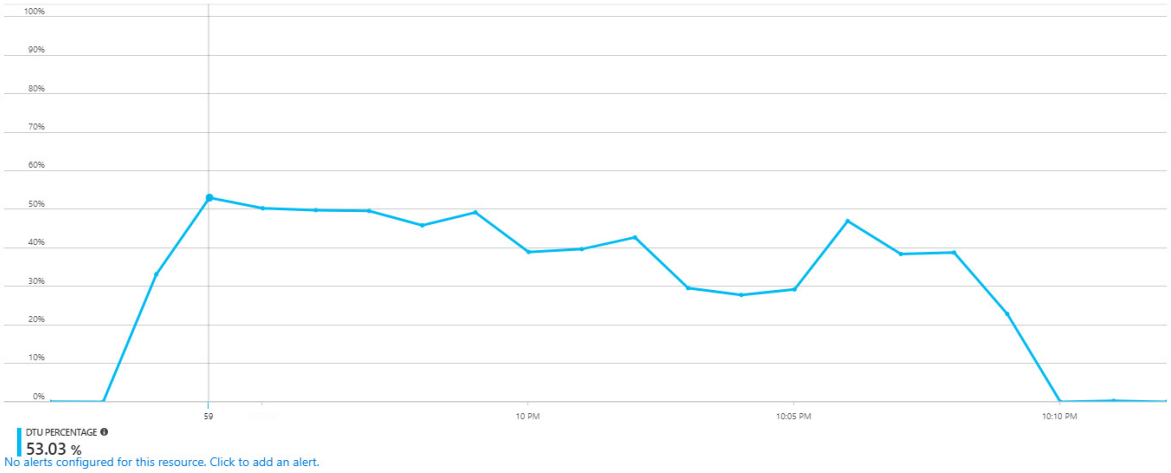
Реакцію системної бази не представлено, оскільки реалізовані сценарії навантажувального тестування не здійснювали фактично ніякого навантаження на цю базу у зв’язку з тим, що при дослідженні прийнято, що всі користувачі належать до одного корпоративного клієнта, а звертання до цієї бази відбувається тільки у випадку частої зміни клієнтів, користувачі яких створюють навантаження. Відповідно до моделі спільної інфраструктури, описаної у четвертому розділі, ця база теж не є її компонентом.

Реакцію бази даних безпеки представлено на Рис. 6.18. Варто відзначити, що величина, яка відображена на графіку, є агрегованим показником, який називається переданими одиницями даних (Data Transfer Unit, DTU) і враховує

використання процесора, оперативної пам’яті та інтерфейсів входу/виходу сервера бази даних. Ресурси баз даних не підлягають адаптивному масштабуванню, тому запас по DTU повинен бути не меншим 50%.



експеримент 1

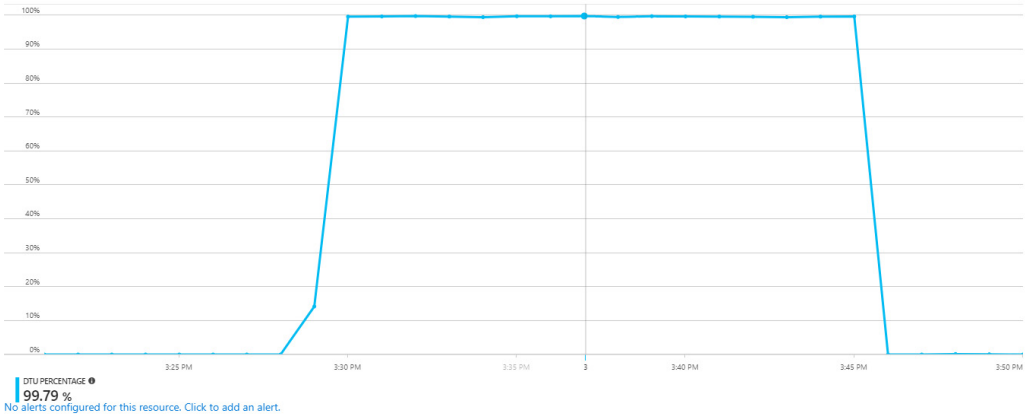


експеримент 2

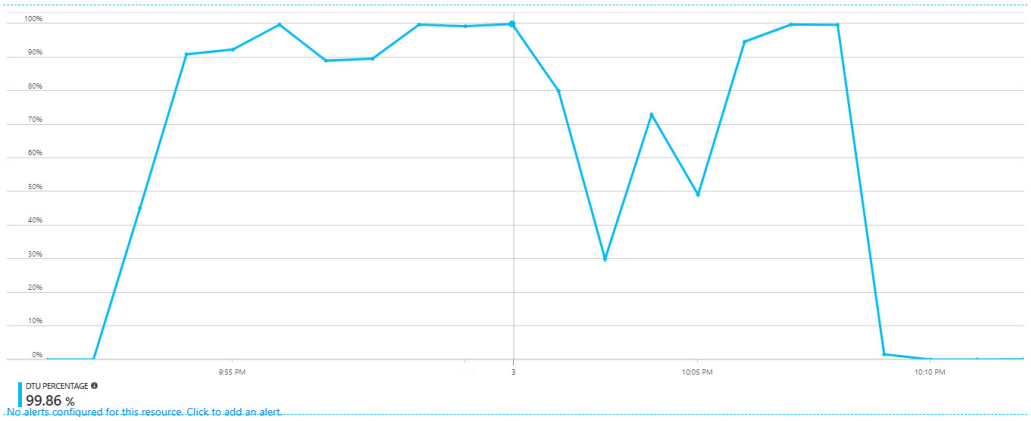
Рис. 6.18. Завантаження бази даних безпеки у процесі навантажувального тестування

На Рис. 6.19 представлено використання клієнтської бази. Результати свідчать, що у програмному коді, який відповідає за взаємодію із цією базою даних, наявні суттєві дефекти, оскільки завантаження цієї бази даних протягом майже всього періоду проведення експерименту було близьким до 100%. Це означає, що вузьке місце системи, яке обмежує її подальше масштабування, виявлено, що є однією із цілей процесу навантажувального тестування. Можна

також спостерігати зміну поведінки інфраструктури зі зміною програмного коду досліджуваної системи.



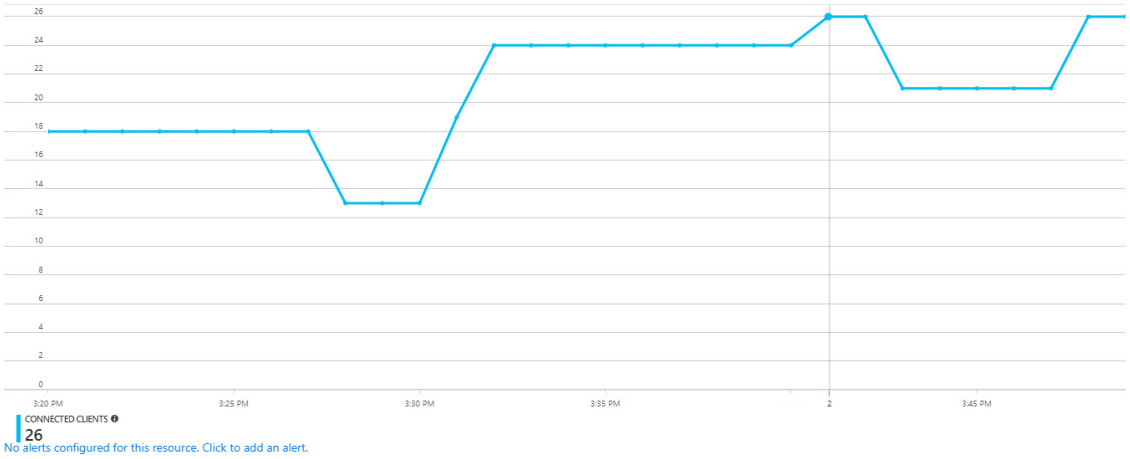
експеримент 1



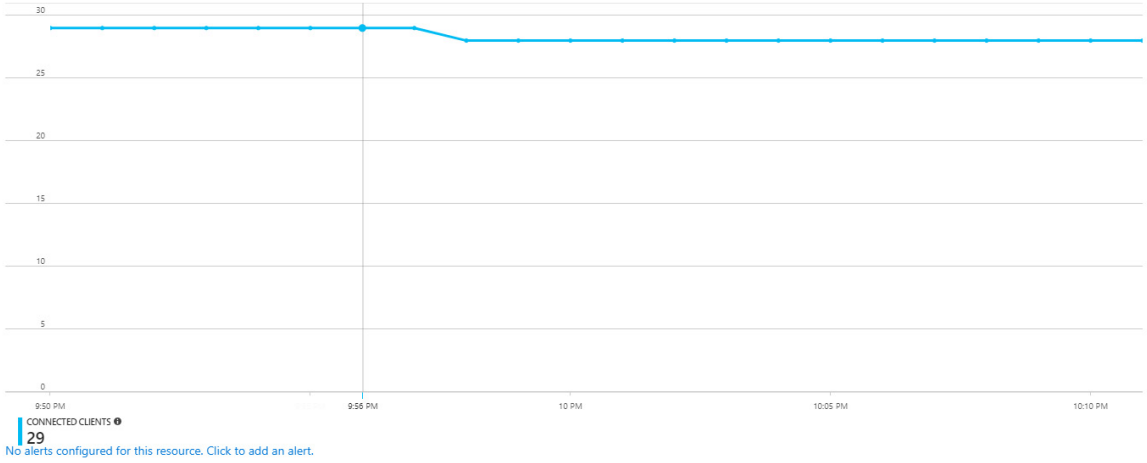
експеримент 2

Рис. 6.19. Завантаження бази даних корпоративного клієнта у процесі навантажувального тестування

На Рис. 6.20 представлено реакцію редіс кешу на процес навантажувального тестування. У зв’язку з особливістю поведінки програмно-апаратної системи кількість одночасних з’єднань до цього компонента не перевищує 30, однак обсяг трафіку, який проходить через цей компонент дуже суттєвий, оскільки він призначений для кешування токенів доступу до системи після того, як користувач пройшов процес автентифікації.



експеримент 1



експеримент 2

Рис. 6.20. Завантаження редіс кешу у процесі навантажувального тестування

6.3.6. Шляхи вдосконалення процесу навантажувального тестування інформаційно-комунікаційних систем

Вдосконалення процесу навантажувального тестування пов’язане із розширенням набору охоплених сценаріїв використання програмної інформаційно-комунікаційної системи.

Також процес навантажувального тестування може бути покращений шляхом ретельного планування ітеративного процесу, параметрів користувацького навантаження тощо.

Підставою для оцінки покращення/погіршення процесу навантажувального тестування є наведені нижче рисунки, які дають змогу оцінити середню

тривалість виконання тестових сценаріїв та кількість охоплених тестових сценаріїв (Рис. 6.21), а також середню тривалість завантаження сторінки [144, 145]. Варто зазначити, що на обох рисунках наведено найбільш відмінні результати, хоча фактична їх кількість є на декілька порядків більшою.

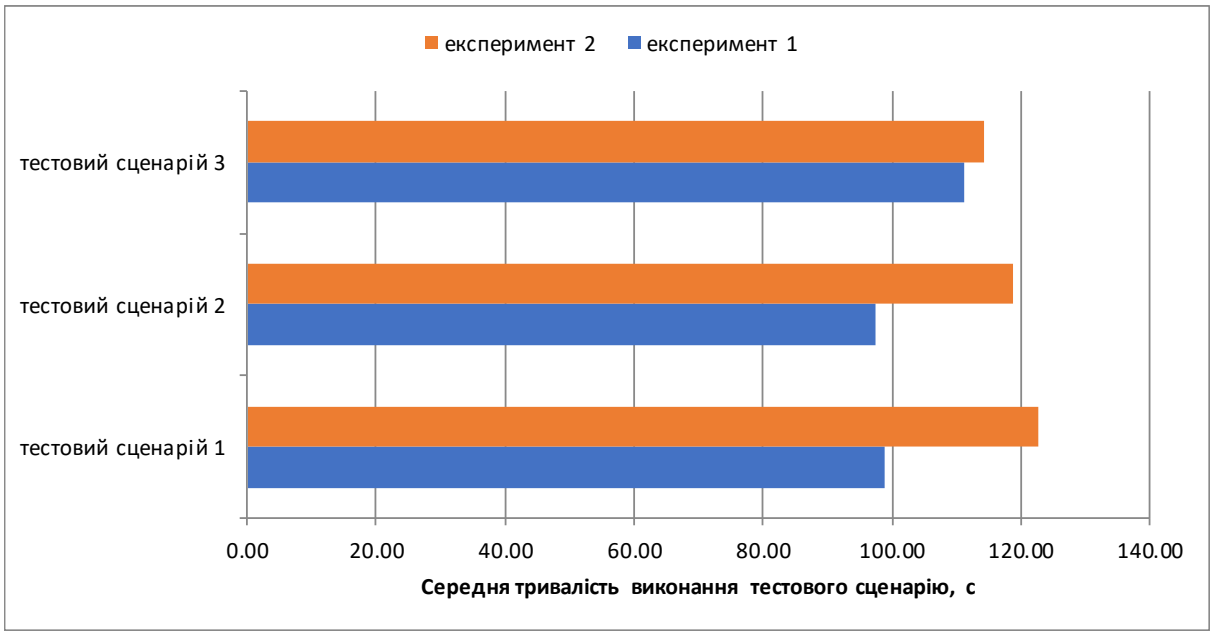


Рис. 6.21. Порівняння тривалості виконання тестових сценаріїв у двох експериментах

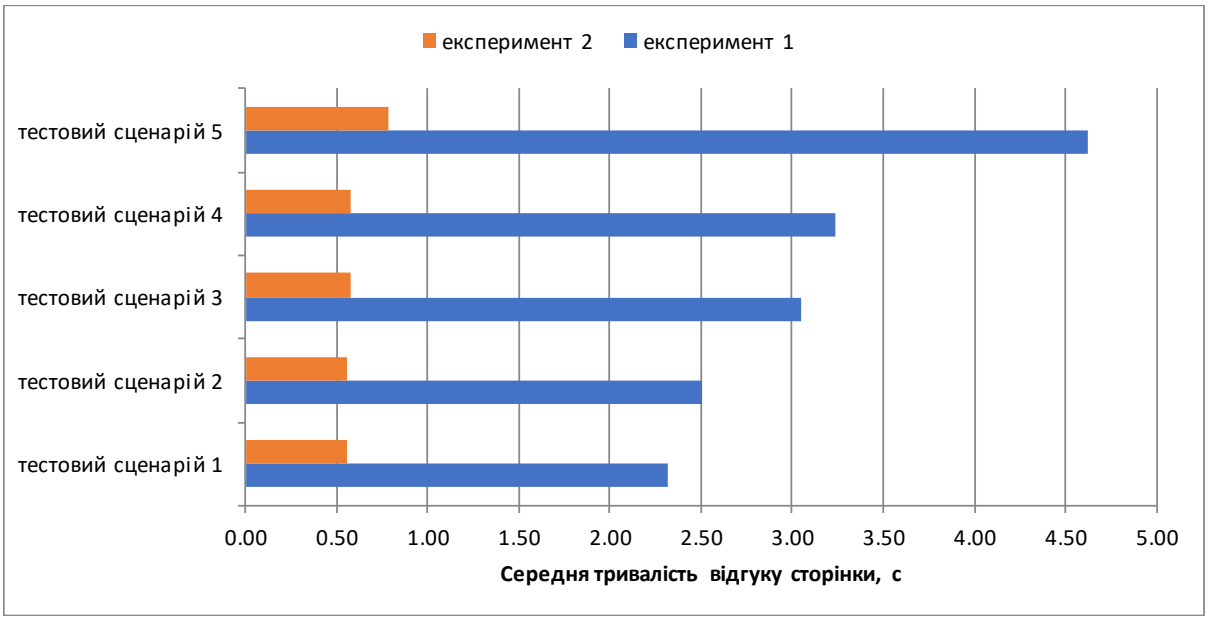


Рис. 6.22. Порівняння тривалості компонентів досліджуваної системи у двох експериментах

6.4. Висновки до шостого розділу

1. Представлено методи розгортання cloud інфраструктур на основі принципу PaaS. Представлено властивості cloud сервісів основних всесвітньовідомих їх провайдерів та властивості інформаційно-телекомунікаційних інфраструктур, побудованих на їх основі. Подано характеристики надання послуг в інформаційно-текомунікаційних системах, побудованих на основі cloud сервісів. Запропоновано удосконалену трирівневу архітектуру інформаційно-телекомунікаційних систем із резервуванням, побудованих із застосуванням cloud сервісів. Представлено концепцію локального кешування вмісту баз даних із застосуванням принципу стано-незалежних серверів.

2. Проведено дослідження доступності cloud сервісів у інформаційно-телекомунікаційних системах із багатокомпонентною структурою cloud інфраструктури із застосуванням методу оцінки доступності багатоланкових структур. Отримано оцінки середньої та ефективної доступностей та їх залежності від обслуженого навантаження та кількості фізичних серверів у системі. Дослідження проведено на основі адаптованої до застосування методу оцінки доступності багатоланкових структур моделі cloud інфраструктури. Отримані результати характеризують поведінку cloud системи за умови незмінності програмного коду cloud сервісів, які формують програмну частину системи.

3. Здійснено навантажувальне тестування досліджуваної розподіленої гетерогенної інформаційно-телекомунікаційної системи, яка базується на cloud інфраструктурі. У процесі навантажувального тестування використано розроблений програмний сервіс системи навантажувального тестування, запропоновані моделі корпоративного клієнта та спільної cloud інфраструктури. За результати навантажувального тестування виявлено вузьке місце у досліджуваній системі, а також отримано оцінки використання інфраструктурних компонентів системи і метрики якості обслуговування та якості сприйняття послуг користувачами досліджуваної інформаційно-

телекомунікаційної системи. Показано, що оновлення програмного коду системи робить неможливим розроблення та використання навіть імітаційних універсальних моделей системи у цілому, оскільки навіть незначні зміни програмного коду, який логічно поєднує компоненти системи, призводить до зміни поведінки системи у цілому, що не може бути враховано у аналітичній, так і у імітаційній моделі [264]. У зв'язку з цим запропонована методика дослідження розподілених гетерогенних інформаційно-телекомунікаційних систем на базі cloud інфраструктури є важливим інструментом у процесі їх проектування та експлуатації для дозволяє у цілому вирішити проблему гармонізації рівнів запропонованої концептуальної структури інформаційно-телекомунікаційних систем із покращенням значень метрик якості обслуговування та оптимальним розподілом ресурсів таких систем. Це свідчить про досягнення мети дисертації.

ВИСНОВКИ

У дисертаційній роботі розв’язано актуальну наукову проблему розроблення методів та моделей надання послуг у гетерогенних розподілених інформаційно-телекомунікаційних системах для підвищення якості обслуговування та покращення структурно-функціональних параметрів і характеристик цих систем з метою узгодження взаємодії сервісного шару та шару передавання даних.

Основні результати роботи такі:

1. Проаналізовано вплив системної архітектури розподілених гетерогенних інформаційно-телекомунікаційних систем на значення їх метрик якості, цілі розроблення архітектур та здійснено порівняння архітектурних шаблонів таких систем. Встановлено, що архітектура системи впливає на метрики якості надання послуг. Аналіз процесів комунікації з урахуванням рівня застосунків є основою для розв’язування завдань розроблення моделей компонентів інформаційно-телекомунікаційних систем та методів надання послуг в таких системах.

2. Запропоновано математичну модель пересилання запиту у процесі надання послуг у гетерогенних сервісно-орієнтованих системах із урахуванням вимог користувачів. Сформульовано завдання розподілу ресурсів таких систем між сервісними потоками як завдання нелінійної оптимізації. Його розв’язок базується на використанні методу розкладання на основі агрегації функцій корисності. Завдання розподілу ресурсів системи розв’язано для різнотипних послуг з урахуванням миттєвих значень функцій корисності кожної послуги. На основі територіально-залежної моделі проведено моделювання поведінки абонентів у інформаційно-телекомунікаційній системі з урахуванням їх активності та переміщення з мінімізацією кількості точок присутності послуг.

3. Запропонований метод балансування навантаження з багатоетапним перемиканням точок присутності послуг збільшує доступність інформаційних ресурсів у періоди пікових навантажень на 14%. Ефективна конфігурація

телекомунікаційної підсистеми надання послуг здійснена шляхом розрахунку необхідних параметрів для максимізації продуктивності інфраструктури та забезпечення процесів передавання сервісних потоків з мінімальними затримками, що значно підвищує показники якості надання послуг розподіленої гетерогенної інформаційно-телекомунікаційної системи та унеможливлює відмову в обслуговуванні, коли кількість активних користувачів та їх вимоги різко зростають. Завдання забезпечення потоків даних достатнім рівнем сервісу розв’язано шляхом оптимального поєднання фізичної і логічної мережних структур.

4. Проведено дослідження конфігурацій мережних топологій на основі теорії графів із їх інтегральним оцінюванням за методом аналізу ієрархій. Отримані результати дають змогу оцінити ефективність вибору шляхів у досліджуваному варіанті структури та порівняти її з еквівалентними конфігураціями. У процесі вибору конфігурації топології телекомунікаційної мережі врахування інтегральної оцінки забезпечує підвищення ефективності функціонування протоколів динамічної маршрутизації. Завдання маршрутизації інформаційних потоків сформульоване і розв’язане із використанням методів алгебраїчної топології. Проведено дослідження телекомунікаційного каналу з підтримкою множини класів обслуговування. Проведено дослідження процесу обслуговування аудіо- та відеопотоків за трьох типів обслуговування і встановлено умови, за яких використання кожного із досліджених типів обслуговування є оптимальним. Запропоновано розв’язок завдання розподілу ресурсів між мережними сервісними потоками за умови рівноправної конкуренції за доступні фізичні ресурси для оцінки максимальної кількості клієнтів, які можуть використовувати послугу з різними рівнями якості обслуговування. Завдання сформульовано в термінах теорії графів і розв’язано методом лінійного програмування для заданого графа мережі. Результати свідчать, що у випадку формування логічної структури системи за протоколом RIP потоки в рівній конкуренції використовують всі доступні мережеві ресурси. У випадку OSPF використовуються не всі доступні фізичні ресурси, адже він

оминає канали з мінімальною пропускною здатністю. Це дає змогу виявити надлишковість у фізичній структурі та зменшити вартість мережі, або ж підвищити її продуктивність через введення k-шляхової маршрутизації.

5. Розроблено нові моделі корпоративного клієнта інформаційно-комунікаційної системи надання послуг та спільної cloud-інфраструктури. Ці моделі відрізняється від відомих тим, що реалізовані у вигляді програмного коду, який внаслідок ініціалізації його параметрами, дає змогу утворити як завгодно багато об'єктів, що представляють cloud-інфраструктуру. Вони використані у процесі навантажувального тестування як елементи інформаційно-телекомунікаційної системи надання послуг. Перевагою цього класу моделей є масштабованість, тобто додавання нових елементів інфраструктури або ж видалення існуючих реалізується зміною у програмному коді і повторною ініціалізацією об'єкта. Модель корпоративного клієнта базується на принципі функцій корисності, який сформульовано у другому розділі роботи. Модель спільної cloud-інфраструктури дає змогу представити спільну мультиклієнтську інфраструктуру у вигляді ініціалізованого параметрами програмного об'єкта, який поєднано із клієнтським програмним об'єктом. Разом із логікою надання послуг ці дві моделі формують розподілену гетерогенну інформаційно-телекомунікаційну систему надання послуг, поведінка якої під впливом користувацького навантаження досліджена у шостому розділі роботи.

6. Розроблено та досліджено систему інформаційної безпеки ІТС як IaaS-послугу розподіленої гетерогенної інформаційно-телекомунікаційної системи. Проведено моделювання атакуючих впливів на інформаційні ресурси до та після застосування заходів захисту. Результати показують зниження ефекту від впливу атак на інформаційно-телекомунікаційну систему до 90%. Для підтвердження адекватності результатів проведено дослідження процесу реальної мережної атаки. Встановлено, що запропонована система функціонує втричі ефективніше. Досліджено процес відновлення інфраструктури після катастрофи Оцінено тривалість післяаварійного відновлення в випадку ручного та автоматичного

відновлення на основі концепції IaaC. Значення виграшу залежить від складності хмарної системи і в умовах виконаного дослідження відрізняється на порядок.

7. Запропоновано реалізацію системи безперервної інтеграції інформаційних сервісів для забезпечення високої якості та оновлення версій сервісу за допомогою системи керування версіями, статичного аналізу якості програмного коду, автоматизованого тестування та утворення виконуваних файлів. Запропоновано удосконалений метод повного впорядкованого групового розсилання, який забезпечує безперервність процесу синхронізації екземплярів бази даних за умови втрати повідомлення процесу-відправника шляхом обмеження часу очікування на відповідь від усіх процесів-адресатів. Це забезпечує зниження імовірності відмови сервісу. Для підтвердження адекватності розробленої моделі досліджено процес обслуговування інформаційних потоків апаратним маршрутизатором та програмним компонентом гетерогенної розподіленої інформаційно-телекомунікаційної системи. Максимальне відхилення затримки пакетів між програмним та апаратним компонентами знаходиться в межах 0,5 %. Доведено, що балансування навантаження за допомогою інтегрованої системи керування розгортанням інформаційних сервісів дає змогу зменшити тривалість обслуговування запитів у середньому до 3 разів. Для дослідження балансування потоків IPTV розроблено дворівневу марківську модель такого потоку на основі як просторової, так і часової кореляції в кодових послідовностях MPEG. Модель описує динаміку процесу зміни кадрів і груп зображень в потоці IPTV та дає змогу оцінити характер потоку у процесі балансування навантаження за методом Round Robin DNS. Для дослідження затримок обслуговування запитів у розподіленій системі з балансуванням навантаження розроблено модель пересилання запитів та виведено співвідношення для оцінки затримки у каналах зі змінною пропускною здатністю. Отримані результати дають змогу стверджувати, що зростання пропускної здатності каналу у 10 разів приводить до зниження затримки обслуговування запиту у середньому у 2,5 рази, що пояснюється обмеженнями методу балансування навантаження.

8. Запропоновано удосконалену трирівневу архітектуру інформаційно-телекомунікаційних систем із резервуванням, побудованих із застосуванням cloud-сервісів та геореплікації. Представлено концепцію локального кешування вмісту баз даних із застосуванням принципу стано-незалежних серверів. Проведено дослідження доступності cloud-сервісів у інформаційно-телекомунікаційних системах із багатокомпонентною структурою cloud-інфраструктури. Отримано оцінки середньої та ефективної доступностей та їх залежності від обслуженого навантаження, а також кількості фізичних серверів у системі. Отримані результати характеризують поведінку cloud-системи за умови незмінності програмного коду cloud-сервісів, які формують програмну частину системи.

9. Проведено навантажувальне тестування досліджуваної розподіленої гетерогенної інформаційно-телекомунікаційної системи, яка базується на cloud-інфраструктурі. У процесі навантажувального тестування використано розроблений програмний сервіс системи навантажувального тестування, запропоновані моделі корпоративного клієнта та спільної cloud-інфраструктури. За результатами навантажувального тестування виявлено вузьке місце у досліджуваній системі, а також отримано оцінки використання інфраструктурних компонентів системи і метрики якості обслуговування та якості сприйняття послуг користувачами досліджуваної інформаційно-телекомунікаційної системи. Показано, що оновлення програмного коду системи робить неможливим розроблення та використання універсальних моделей системи, оскільки навіть незначні зміни програмного коду, який логічно поєднує компоненти системи, призводить до зміни її поведінки, що не може бути враховано ні у аналітичній, ні у імітаційній моделях. У зв’язку з цим, запропонована методика дослідження розподілених гетерогенних інформаційно-телекомунікаційних систем на базі cloud-інфраструктури є важливим інструментом у процесі їх проектування та експлуатації і дає змогу у цілому вирішити проблему гармонізації рівнів запропонованої концептуальної структури інформаційно-телекомунікаційних систем із покращенням значень

метрик якості обслуговування та оптимальним розподілом ресурсів цих систем. Це свідчить про досягнення мети дисертації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] “Documentation - Terraform by HashiCorp.” [Online]. Available: https://www.terraform.io/docs/index.html. [Accessed: 10-Jan-2018].

[2] V. Petroff, “Self-Similar Network Traffic : From Chaos and Fractals to Forecasting and QoS,” pp. 1–9.

[3] S. Valsamidis, S. Kontogiannis, A. Karakos, and A. Efraimidis, “An adaptive load balancing algorithm for cluster-based web systems.”

[4] Y. Li, A. H. Chan, and M. Ieee, “A Mechanism of Applying Human Intelligence to Future Generation Network.”

[5] Н. А. Земской, О. Г. Иванова, А. В. Лагутин, and В. М. Тютюнник, *Фрактальный анализ и процессы в компьютерных сетях*.

[6] A. Ulvan and R. Bestak, “Handover Scenario and Procedure in LTE-based Femtocell Networks.”

[7] О. Н. Щукина, “Точные и приближенные методы анализа и расчета вероятностей блокировок соединений в мультисервисных сетях.”

[8] K. Lee, “Efficient Distributed Recovery Algorithm using Non-FIFO Channel in Distributed Network,” pp. 166–171.

[9] D. F. O. R. Reliability and O. F. C. Solutions, “Design for reliability of cloud solutions,” pp. 271–295.

[10] A. Auyoung, L. Grit, J. Wiener, and J. Wilkes, “Service contracts and aggregate utility functions Service provider.”

[11] B. Buhyl, P. Huskov, O. Lavriv, R. Bak, and A. Luntovskyy, “Maximization of Service Flows Rates as a Solution of Network Capacity Allocation Problem,” *Internet Things Eng. Appl.*, vol. 3, no. 1, pp. 1–10, May 2018.

[12] O. Lavriv, M. Klymash, G. Grynkevych, O. Tkachenko, and V. Vasylenko, “Method of cloud system disaster recovery based on ‘Infrastructure as a code’ concept,” in *14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 1139–1142.

[13] R. Bak, O. Lavriv, and B. Koval, “Load balancing based on multi-hop handover for wireless cellular networks,” in *2017 IEEE 1st Ukraine Conference on Electrical and Computer Engineering, UKRCON 2017 - Proceedings*, 2017, pp. 1103–1106.

[14] O. Lavriv, Z. Kharkhalis, and B. Strykhaliuk, “Concept of intrusion detection system with implementation of topological sorting,” in *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings*, 2017, pp. 170–172.

[15] P. Huskov, O. Lavriv, and M. Klymash, “The concept of services assurance in heterogeneous service-oriented systems,” in *2016 3rd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2016 - Proceedings*, 2017, pp. 24–26.

[16] O. Lavriv, B. Buhyl, P. Huskov, and R. Bak, “Heterogeneous network capacity distribution among service flows,” in *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings*, 2017, pp. 173–175.

[17] O. Lavriv, B. Buhyl, M. Klymash, and G. Grynkevych, “Services continuous integration based on modern free infrastructure,” in *2017 2nd International Conference on Advanced Information and Communication Technologies (AICT)*, 2017, pp. 150–153.

[18] C. Baham, R. Hirschheim, A. A. Calderon, and V. Kisekka, “An Agile Methodology for the Disaster Recovery of Information Systems Under Catastrophic Scenarios,” *J. Manag. Inf. Syst.*, vol. 34, no. 3, pp. 633–663, Jul. 2017.

[19] P. Yatskiv, O. Lavriv, Z. Kharkhalis, V. Mosorov, and M. Niedźwiedziński, “Security. Provisions for Ensuring Telecommunication Networks’ Protection from Malicious Influences,” *Przedsiębiorczość i Zarządzanie*, vol. 17, no. 11, pp. 111–125, 2016.

[20] Р. П. Кришталь, О. А. Лаврів, and З. М. Хархаліс, “Удосконалення алгоритму повного групового розсилання у системах оброблення даних з розподіленою архітектурою,” *Вісник Національного університету “Львівська*

політехніка", серія "Радіоелектроніка та телекомунікації," no. 849, pp. 241–247, 2016.

[21] Р. І. Бак, П. О. Гуськов, and О. А. Лаврів, "Імітаційна макромодель поведінки абонентів у мережі коміркового зв'язку," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* no. 849, pp. 274–284, 2016.

[22] О. А. Лаврів, З. М. Хархаліс, and В. Я. Мосоров, "Моделювання DDoS атаки на веб-сервер," in *10-а Міжнародна науково-технічна конференція "Проблеми телекомунікацій – 2016,"* 2016, pp. 348–351.

[23] О. А. Лаврів, Б. А. Бугиль, П. О. Гуськов, Р. Р. Баса, "Розподіл ємності телекомунікаційної мережі із максимізацією інтенсивності інформаційних потоків," in *"Фізико-технологічні проблеми передавання, оброблення та зберігання інформації в інфокомунікаційних системах,"* 2016, С. 134–135.

[24] M. Seliuchenko, O. Lavriv, O. Panchenko, and V. Pashkevych, "Enhanced multi-commodity flow model for QoS-aware routing in SDN," in *2016 IEEE International Scientific Conference "Radio Electronics and Info Communications", UkrMiCo 2016*, 2016, pp. 1–3.

[25] I. Demydov, O. Lavriv, Z. Kharkhalis, and M. M. El-Hatri, "Concept of the migrating firewall to scalable cloud networks," in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET 2016*, 2016, pp. 643–645.

[26] A. Masiuk, M. Beshley, O. Lavriv, and Y. Deschynskiy, "Common radio resource management model for heterogeneous cellular networks," in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET 2016*, 2016, pp. 661–663.

[27] O. V Lemeshko, O. S. Yeremenko, and A. M. Hailan, "QoS solution of traffic management based on the dynamic tensor model in the coordinate system of interpolar paths and internal node pairs," *2016 IEEE Int. Sci. Conf. "Radio Electron. Info Commun. UkrMiCo 2016 - Conf. Proc.*, 2016.

[28] O. Lemeshko, O. Nevzorova, and T. Vavenko, “Hierarchical coordination method of inter-area routing in telecommunication network,” *2016 IEEE Int. Sci. Conf. "Radio Electron. Info Commun. UkrMiCo 2016 - Conf. Proc.*, 2016.

[29] V. R. Senchenko, “Intelligent Modeling System based on Cloud-technology,” pp. 1–4, 2016.

[30] V. Boyun, “Directions of development of intelligent real time video systems,” *2016 Int. Conf. Radio Electron. Info Commun.*, pp. 1–7, 2016.

[31] D. Titov and A. Doroshenko, “Automated Design of a Parallel Distributed System for Streaming Data Processing,” 2016.

[32] T. Sergii and S. Tech, “Optimization of Data Access in Tiered Storage,” no. 2, pp. 1–4, 2016.

[33] I. Demydov and O. Shpur, “The Method of Improving the Availability at Cloud Service Systems,” pp. 2–4, 2016.

[34] OWASP INJECTION, “SQL Injection Prevention Cheat Sheet,” *2016*, p. 1, 2016.

[35] O. Rolik, V. Kolesnik, and D. Halushko, “Decomposition-Compensation Method of Service Level Management in Corporate IT Infrastructures with the use of Adaptive Genetic Algorithm,” 2016.

[36] H. Maziku and S. Shetty, “Towards a Network-Aware VM Migration : Evaluating the Cost of VM Migration in Cloud Data Centers,” no. CloudNet 2014, pp. 114–119, 2016.

[37] M. Klymash, O. Lavriv, T. Maksymyuk, and M. Beshley, “State of the art and further development of information and communication systems,” in *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, 2016, pp. 1–6.

[38] O. Shpur, M. Klymash, V. Seliuchenko, B. Strykhaliuk, and O. Lavriv, “Improving the Quality of Composite Services Through Improvement of Cloud Infrastructure Management,” *Int. J. Comput. Sci. Inf. Secur.*, vol. 13, no. 9, pp. 36–44, 2015.

[39] M. Beshley, M. Seliuchenko, O. Lavriv, V. Chervenets, H. Kholiavka, and V. Klymash, “Increasing the efficiency of real-time content delivery by improving the technology of priority assignment and processing of IP traffic,” *Smart Comput. Rev.*, vol. 5, no. 2, pp. 1–13, 2015.

[40] М. І. Бешлей, М. О. Селюченко, О. А. Лаврів, А. Р. Масюк, and Г. В. Холявка, “Оцінка адекватності функціонування програмного маршрутизатора у процесі обслуговування мультимедійного трафіку,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* no. 818, pp. 162–173, 2015.

[41] O. Lavriv, I. Kahalo, and R. Kolodiy, “Application of NoSQL approach in data-centered network architectures: big data case,” in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, pp. 103–105.

[42] M. Al-Shuraifi, H. Al-Zayadi, O. Lavriv, and M. Klymash, “Improving QoS in MAX C/I Scheduling Using Resource Allocation Type 1 of LTE,” in *13 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2015, pp. 12–14.

[43] О. А. Лаврів, Б. М. Стрихалюк, О. М. Шпур, and Т. А. Максимюк, “Дистанційний моніторинг параметрів місця вчинення злочину з використанням мобільних технологій,” in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, pp. 67–69.

[44] O. Krasko, M. Klymash, and O. Lavriv, “Data flows transmission models in converged optical access networks,” in *2015 2nd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2015 - Conference Proceedings*, 2015, pp. 65–68.

[45] N. Sahebjamnia, S. A. Torabi, and S. A. Mansouri, “Integrated business continuity and disaster recovery planning: Towards organizational resilience,” *Eur. J. Oper. Res.*, vol. 242, no. 1, pp. 261–273, Apr. 2015.

[46] V. Chang, “Towards a Big Data system disaster recovery in a Private Cloud,” *Ad Hoc Networks*, vol. 35, pp. 65–82, Dec. 2015.

[47] J. M. Ruiz-Aviles, M. Toril, S. Luna-Ramirez, V. Buenestado, and M. A. Regueira, “Analysis of Limitations of Mobility Load Balancing in a Live LTE System,” *IEEE Wirel. Commun. Lett.*, vol. 4, no. 4, pp. 417–420, 2015.

[48] Z. Wang, R. Schoenen, H. Yanikomeroglu, and M. St-hilaire, “Load Balancing in Cellular Networks with A Spatial Traffic Shaping Approach,” pp. 4241–4246, 2015.

[49] Z. Gao, C. Chen, Y. Li, B. Wen, L. Huang, and Y. Zhao, “A mobility load balancing algorithm based on handover optimization in LTE network,” *10th Int. Conf. Comput. Sci. Educ. ICCSE 2015*, no. Iccse, pp. 611–614, 2015.

[50] Zhangpeng Huang, Jing Liu, Qiang Shen, Jin Wu, and Xiaoying Gan, “A threshold-based multi-traffic load balance mechanism in LTE-A networks,” *2015 IEEE Wirel. Commun. Netw. Conf.*, pp. 1273–1278, 2015.

[51] F. Zhou, L. Feng, P. Yu, and W. Li, “A load balancing method in downlink LTE network based on load vector minimization,” *Proc. 2015 IFIP/IEEE Int. Symp. Integr. Netw. Manag. IM 2015*, pp. 525–530, 2015.

[52] R. Castro-Hernandez, Diego and Paranjape, “A Distributed Load Balancing Algorithm for LTE / LTE-A Heterogeneous Networks,” *Wirel. Commun. Netw. Conf. Work.*, pp. 380–385, 2015.

[53] F. Zhou, L. Feng, P. Yu, and W. Li, “Energy-efficiency driven load balancing strategy in LTE-WiFi interworking heterogeneous networks,” *2015 IEEE Wirel. Commun. Netw. Conf. Work. WCNCW 2015*, pp. 276–281, 2015.

[54] S. Ben Rejeb, S. Tabbane, and N. Nasser, “An adaptive auto-tuning scheme based mobility in 4G and beyond networks,” *Proc. 2015 Int. Conf. Electr. Inf. Technol. ICEIT 2015*, pp. 329–334, 2015.

[55] P. Kreuger, O. Gornerup, D. Gillblad, T. Lundborg, D. Corcoran, and A. Ermedahl, “Autonomous load balancing of heterogeneous networks,” *IEEE Veh. Technol. Conf.*, vol. 2015, 2015.

[56] A. Gomes and T. Braun, “Load balancing in LTE mobile networks with Information-Centric Networking,” *2015 IEEE Int. Conf. Commun. Work. ICCW 2015*, pp. 1845–1851, 2015.

[57] D. R. Purohith, A. Hegde, and K. M. Sivalingam, “Network architecture supporting seamless flow mobility between LTE and WiFi networks,” *Proc. WoWMoM 2015 A World Wirel. Mob. Multimed. Networks*, 2015.

[58] A. Alfayly, A. Alfayly, I. Mkwawa, and L. Sun, “QoE-based performance evaluation of scheduling algorithms over LTE QoE-based Performance Evaluation of Scheduling Algorithms over LTE,” no. October, pp. 1362–1366, 2015.

[59] U. Bayram, D. Divine, P. Zhou, and E. W. D. Rozier, “Improving Reliability with Dynamic Syndrome Allocation in Intelligent Software Defined Data Centers,” *Proc. Int. Conf. Dependable Syst. Networks*, vol. 2015–Septe, pp. 219–230, 2015.

[60] A. Blenk, A. Basta, and W. Kellerer, “HyperFlex: An SDN virtualization architecture with flexible hypervisor function allocation,” *Proc. 2015 IFIP/IEEE Int. Symp. Integr. Netw. Manag. IM 2015*, pp. 397–405, 2015.

[61] S. Peng *et al.*, “Multi-Tenant Software-Defined Hybrid Optical Switched Data Centre,” *J. Light. Technol.*, vol. 33, no. 15, pp. 3224–3233, 2015.

[62] A. Darabseh, M. Al-Ayyoub, Y. Jararweh, E. Benkhelifa, M. Vouk, and A. Rindos, “SDDC: A Software Defined Datacenter Experimental Framework,” *Proc. - 2015 Int. Conf. Futur. Internet Things Cloud, FiCloud 2015 2015 Int. Conf. Open Big Data, OBD 2015*, pp. 189–194, 2015.

[63] A. Fressancourt and M. Gagnaire, “A SDN-based network architecture for cloud resiliency,” *2015 12th Annu. IEEE Consum. Commun. Netw. Conf. CCNC 2015*, pp. 479–484, 2015.

[64] W. Li, H. Qi, K. Li, and I. Stojmenovic, “Joint optimization of bandwidth for provider and delay for user in software defined data centers,” *IEEE Trans. Cloud Comput.*, vol. PP, no. 99, pp. 1–14, 2015.

[65] M. Canini, P. Kuznetsov, D. Levin, and S. Schmid, “A distributed and robust SDN control plane for transactional network updates,” *Proc. - IEEE INFOCOM*, vol. 26, pp. 190–198, 2015.

[66] M. Zhukova, “Development of the protected telecommunication systems,” 2015.

[67] S. Maity, E. Al-Shaer, P. Bera, and S. K. Ghosh, “Formal integrated network security analysis tool: formal query-based network security configuration analysis,” *IET Networks*, vol. 4, no. 2, pp. 137–147, 2015.

[68] J. Costa-Requena *et al.*, “SDN and NFV integration in generalized mobile network architecture,” *2015 Eur. Conf. Networks Commun. EuCNC 2015*, pp. 154–158, 2015.

[69] E. Kassela, C. Boumpouka, I. Konstantinou, and N. Koziris, “Automated workload-aware elasticity of NoSQL clusters in the cloud,” *Proc. - 2014 IEEE Int. Conf. Big Data, IEEE Big Data 2014*, pp. 195–200, 2015.

[70] J. Bhogal and I. C71hoksi, “Handling Big Data Using NoSQL,” *Proc. - IEEE 29th Int. Conf. Adv. Inf. Netw. Appl. Work. WAINA 2015*, pp. 393–398, 2015.

[71] J. Kim and J.-G. Lee, “Community detection in multi-layer graphs: A survey,” *SIGMOD Rec.*, vol. 44, no. 3, pp. 37–48, 2015.

[72] M. M. Klymash, H. A. Al-Zayadi, and O. A. Lavriv, “Improving Thoughput Using Channel Quality Indicator in LTE Technology,” *Збірник наукових праць Донецького національного технічного університету, серія “Обчислювальна техніка та автоматизація,”* no. 1(26), pp. 134–144, 2014.

[73] H. A. Al-Zayadi, M. M. Klymash, and O. A. Lavriv, “Mobility Affected on Channel Estimation Using Different Modulation in LTE,” *Проблеми телекомунікацій*, no. 2(14), pp. 30–41, 2014.

[74] М. М. Климаш, М. І. Бешлей, Б. М. Стрихалюк, and О. А. Лаврів, “Підвищення якості обслуговування в конвергентних мобільних системах на основі платформи UMA-A,” *Проблеми телекомунікацій*, no. 1(13), pp. 3–19, 2014.

[75] М. І. Бешлей, В. П. Ткачук, Б. А. Бугиль, and О. А. Лаврів, “Модель системи динамічного управління пропускною здатністю каналу інтегрованої WI-FI/GSM мережі,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* no. 796, pp. 83–96, 2014.

[76] H. A. Al-Zayadi, O. Lavriv, and M. Klymash, “QoE Estimation on the Basis of LTE Service Architecture,” in *12th International Conference Modern Problems of*

Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University, 2014, pp. 590–592.

[77] М. І. Бешлей, О. А. Лаврів, and Г. В. Холявка, “Дослідження методів побудови конвергентної мережі оператора мобільного зв’язку для надання послуг quad play,” in *Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій*, 2014, pp. 76–77.

[78] О. В. Красько and О. А. Лаврів, “Адаптація смуги пропускання в мережах NG-SDH/WDM,” in *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, pp. 201–204.

[79] M. Seliuchenko, A. Kovalchuk, O. Lavriv, and M. Klymash, “Enhancing Reliability of Transport Software-Defined Networks Using Flow Table Mutual Reservation Method,” in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 573–575.

[80] І. В. Демидов, О. А. Лаврів, О. М. Шпур, and М. О. Селюченко, “Доступність композитних застосувань у сервісно-орієнтованих системах,” in *Вимірювальна та обчислювальна техніка в технологічних процесах*, 2014, pp. 119–122.

[81] Б. М. Стрихалюк, О. А. Лаврів, and З. М. Хархаліс, “Проблематика розгортання SDP для операторів мобільного зв’язку на основі віртуалізованих мереж дата-центрів,” in *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, pp. 221–224.

[82] М. М. Климаш, М. О. Селюченко, О. А. Лаврів, “Забезпечення відмовостійкості багаторівневої ієрархії управління у програмно-керованих мережах,” in *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, С. 225–228.

[83] M. A. Khoshkholghi, A. Abdullah, R. Latip, S. Subramaniam, and M. Othman, “Disaster Recovery in Cloud Computing: A Survey,” *Comput. Inf. Sci.*, vol. 7, no. 4, pp. 39–54, Sep. 2014.

[84] H. Al-Zayadi, O. Lavriv, M. Klymash, and A.-S. Mushtaq, “Increase throughput by expectation channel quality indicator,” in *2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology*, 2014, pp. 120–121.

[85] M. Sheng, C. Yang, Y. Zhang, and J. Li, “Zone-based load balancing in LTE self-optimizing networks: A game-theoretic approach,” *IEEE Trans. Veh. Technol.*, vol. 63, no. 6, pp. 2916–2925, 2014.

[86] S. Betgé-Brezetz, G. B. Kamga, A. El Amrani Joutei, and O. Maalmi, “Control of sensitive traffic in the cloud based on OpenFlow,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 266–268, 2014.

[87] A. Faggiani, E. Gregori, A. Improta, L. Lenzini, V. Luconi, and L. Sani, “A study on traceroute potentiality in revealing the Internet AS-level topology,” *2014 IFIP Netw. Conf. IFIP Netw. 2014*, pp. 1–9, 2014.

[88] S. Maity and A. Chaudhuri, “Optimal negotiation of SLA in federated cloud using multiobjective genetic algorithms,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 269–271, 2014.

[89] J. Soares, M. Dias, J. Carapinha, B. Parreira, and S. Sargento, “Cloud4NFV: A platform for Virtual Network Functions,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 288–293, 2014.

[90] S. Sardellitti, G. Scutari, and S. Barbarossa, “Joint Optimization of Radio and Computational Resources for Multicell Mobile-Edge Computing,” *Ieee Trans. Signal Inf. Process. Over Networks*, vol. 1, no. 2, pp. 1–13, 2014.

[91] G. Lee, H. Ko, S. Pack, and C. H. Kang, “Efficient delta synchronization algorithm in mobile cloud networks,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 455–460, 2014.

[92] R. Cziva, D. Stapleton, F. P. Tso, and D. P. Pezaros, “SDN-based Virtual Machine management for Cloud Data Centers,” *2014 IEEE 3rd Int. Conf. Cloud Netw.*, pp. 388–394, 2014.

[93] K. Blaiech, S. Hamadi, A. Mseddi, and O. Cherkaoui, “Data plane acceleration for virtual switching in data centers: NP-based approach,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 108–113, 2014.

[94] M. Rana, S. Bilgaiyan, and U. Kar, “A Study on Load Balancing in Cloud Computing Environment Using Evolutionary and Swarm Based Algorithms,” pp. 245–250, 2014.

[95] S. K. Dhurandher, M. S. Obaidat, I. Woungang, P. Agarwal, A. Gupta, and P. Gupta, “A cluster-based load balancing algorithm in cloud computing,” *Commun. (ICC), 2014 IEEE Int. Conf.*, pp. 2921–2925, 2014.

[96] G. Portaluri and S. Giordano, “A Power Efficient Genetic Algorithm for Resource Allocation in Cloud Computing Data Centers,” pp. 58–63, 2014.

[97] N. U. H. Shirazi, S. Simpson, A. K. Marnerides, M. Watson, A. Mauthe, and D. Hutchison, “Assessing the impact of intra-cloud live migration on anomaly detection,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, no. Ci, pp. 52–57, 2014.

[98] K. Ravindran, “QoS Management by Competitive Agent-based Negotiation in Distributed Cloud Services,” pp. 191–196, 2014.

[99] M. Vondra and Z. Becvar, “QoS-ensuring distribution of computation load among cloud-enabled small cells,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 197–203, 2014.

[100] W. Xie, J. Zhu, C. Huang, M. Luo, and W. Chou, “Dynamic resource pooling and trading mechanism in flexible-grid optical network virtualization,” *2014 IEEE 3rd Int. Conf. Cloud Networking, CloudNet 2014*, pp. 167–172, 2014.

[101] D. Ehwzhhq, W. K. H. Yluwxdo, P. Ri, and W. K. H. Gdwd, “Load balancing data center,” vol. 0, pp. 141–144, 2014.

[102] J. B. Tomas-Gabarron and Y. Ghamri-Doudane, “Distributed load balancing by two-hop relaying in LTE-Advanced networks,” *2014 11th Annu. IEEE Int. Conf. Sensing, Commun. Networking, SECON 2014*, pp. 337–345, 2014.

[103] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, “Heterogeneity in mobile cloud computing: Taxonomy and open challenges,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 369–392, 2014.

[104] J. Spiess, Y. T. Joens, R. Dragnea, and P. Spencer, “Using Big Data to Improve Customer Experience and Business Performance,” *Bell Labs Tech. J.*, vol. 18, no. 4, pp. 3–17, 2014.

[105] Y. Song, Y. Han, and Y. Choi, “A QoE-aware joint resource allocation algorithm for uplink carrier aggregation in LTE-Advanced systems,” *2014 IEEE Int. Conf. Commun. ICC 2014*, pp. 1065–1069, 2014.

[106] M. Silic, G. Delac, I. Krka, and S. Srbljic, “Scalable and accurate prediction of availability of atomic web services,” *IEEE Trans. Serv. Comput.*, vol. 7, no. 2, pp. 252–264, 2014.

[107] G. Zou, Q. Lu, Y. Chen, R. Huang, Y. Xu, and Y. Xiang, “QoS-aware dynamic composition of web services using numerical temporal planning,” *IEEE Trans. Serv. Comput.*, vol. 7, no. 1, pp. 1–14, 2014.

[108] S. Deng, L. Huang, W. Tan, and Z. Wu, “Top-k automatic service composition: A parallel method for large-scale service sets,” *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 891–905, 2014.

[109] N. Feamster, J. Rexford, and E. Zegura, “The Road to SDN: An Intellectual History of Programmable Networks,” *ACM Sigcomm Comput. Commun.*, vol. 44, no. 2, pp. 87–98, 2014.

[110] M. A. McCarthy, L. M. Herger, and S. M. Khan, “A compliance aware software defined infrastructure,” *Proc. - 2014 IEEE Int. Conf. Serv. Comput. SCC 2014*, pp. 560–567, 2014.

[111] A. Krishnamurthy, S. P. Chandrabose, and A. Gember-jacobson, “Pratyaastha : An Efficient Elastic Distributed SDN Control Plane,” *HotSDN 2014*, pp. 133–138, 2014.

[112] X. Xiang, C. Lin, and X. Chen, “Energy-efficient link selection and transmission scheduling in mobile cloud computing,” *IEEE Wirel. Commun. Lett.*, vol. 3, no. 2, pp. 153–156, 2014.

[113] A. Anjum *et al.*, *A Rule-Based Approach to Activity Recognition*, vol. 6, no. 1. 2014.

[114] Y. Liu, J. K. Muppala, and M. Veeraraghavan, “A survey of data center network architectures,” p. 22pp, 2014.

[115] R. Alshahrani and H. Peyravi, “Modeling and simulation of data center networks,” *Proc. 2nd ACM SIGSIM/PADS Conf. Princ. Adv. Discret. Simul. - SIGSIM-PADS ’14*, pp. 75–82, 2014.

[116] J. Liao, L. Li, H. Chen, and X. Liu, “Adaptive Replica Synchronization for Distributed File Systems,” *IEEE Syst. J.*, vol. 9, no. 3, pp. 865–877, 2014.

[117] J. Costa-Requena, “SDN integration in LTE mobile backhaul networks,” *Int. Conf. Inf. Netw.*, pp. 264–269, 2014.

[118] D. Li *et al.*, “Reliable multicast in data center networks,” *IEEE Trans. Comput.*, vol. 63, no. 8, pp. 2011–2024, 2014.

[119] J. Cao, K. Li, and I. Stojmenovic, “Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers,” *IEEE Trans. Comput.*, vol. 63, no. 1, pp. 45–58, 2014.

[120] V. N. Gudivada, D. Rao, and V. V. Raghavan, “NoSQL Systems for Big Data Management,” *2014 IEEE World Congr. Serv.*, pp. 190–197, 2014.

[121] M. Scavuzzo, E. Di Nitto, and S. Ceri, “Interoperable data migration between NoSQL columnar databases,” *Proc. - IEEE Int. Enterp. Distrib. Object Comput. Work. EDOCW*, pp. 154–162, 2014.

[122] C. P. Tsai and H. C. Hsiao, “Streaming in NoSQL,” *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, vol. 2015–April, pp. 867–872, 2014.

[123] M. Van Der Pol, G. Currie, S. Kromm, and M. Ryan, “Specification of the utility function in discrete choice experiments,” *Value Heal.*, vol. 17, no. 2, pp. 297–301, 2014.

[124] Y. Hayali, O. Lavriv, B. Buhyl, R. Bak, and M. Klymash, “Models and mechanisms for traffic balancing of IPTV VoD service,” *Int. J. Serv. Econ. Manag.*, vol. 5, no. 4, pp. 291–300, 2013.

[125] I. Demydov, O. Lavriv, Y. Dobush, M. Klymash, and B. Buhyl, “Analysis of service workflows distribution and service delivery platform parameters,” *Int. J. Serv. Econ. Manag.*, vol. 5, no. 4, pp. 280–290, 2013.

[126] І. М. Кузь, З. М. Хархаліс, П. О. Гуськов, and О. А. Лаврів, “Дослідження методів частотного планування коміркових мереж на основі технологій LTE та GSM,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* no. 766, pp. 82–87, 2013.

[127] O. Yaremko, B. Stryhalyuk, T. Maksymyuk, O. Lavriv, and D. Kozhurov, “The optimal power control method in multiuser cellular networks,” *An Int. Q. J. Econ. Technol. New Technol. Model. Process.*, vol. 2, no. 1, pp. 63–67, 2013.

[128] Б. А. Бугиль, О. А. Лаврів, М. І. Бешлей, and В. В. Червенець, “Методи оптимізації фізичної та логічної структур телекомунікаційних мереж,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* no. 766, pp. 76–81, 2013.

[129] Y. Hayali, M. Klymash, and O. Lavriv, “Analysis of IPTV VoD Traffic Balancing Mechanisms,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 140–142.

[130] М. М. Климаш, О. А. Лаврів, and М. М. Бешлей, “Моделювання та прогнозування параметрів і структур мультисервісної мережі з диференційованим обслуговуванням послуг,” in *VI Міжнародний науково-технічний симпозіум «Нові технології в телекомунікаціях»*, 2013, pp. 28–31.

[131] З. М. Хархаліс and О. А. Лаврів, “Порівняльний аналіз методів побудови мультиоператорних мереж мобільного зв’язку,” in *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, pp. 249–252.

[132] M. Klymash, M. Beshley, and O. Lavriv, “Model of Network Resources Management on the Basis of Services Priorities Association,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 164–166.

[133] Д. В. Кожуров, О. А. Лаврів, and А. Р. Масюк, “Модель обміну керуючою інформацією в сервісно-орієнтованій Cloud-мережі,” in *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, pp. 105–108.

[134] M. O. Seliuchenko ; O. A. Lavriv ; M. M. Klymash, “Analysis of load balancing methods for improving of utilization efficiency of hardware resources of distributed systems,” in *Microwave and Telecommunication Technology (CriMiCo), 2013 23rd International Crimean Conference*, 2013, pp. 515–516.

[135] N. Zia and A. Mitschele-Thiel, “Self-organized neighborhood mobility load balancing for LTE networks,” *IFIP Wirel. Days*, pp. 1–6, 2013.

[136] X. Lin, J. G. Andrews, R. Ratasuk, B. Mondal, and A. Ghosh, “Carrier Aggregation in Heterogeneous Cellular Networks,” pp. 5199–5203, 2013.

[137] Y. Li, H. Wang, M. Liu, B. Zhang, and H. Mao, “Software defined networking for distributed mobility management,” *2013 IEEE Globecom Work. GC Wkshps 2013*, pp. 885–889, 2013.

[138] J. Bhatia, T. Patel, H. Trivedi, and V. Majmudar, “HTV dynamic load balancing algorithm for virtual machine instances in cloud,” *Proc. - 2012 Int. Symp. Cloud Serv. Comput. ISCOS 2012*, pp. 15–20, 2013.

[139] P. Gupta, M. Goyal, and P. Kumar, “Trust and reliability based load balancing algorithm for cloud IaaS,” *Adv. Comput. Conf. ...*, pp. 65–69, 2013.

[140] B. Prabavathy, K. Priya, and C. Babu, “A load balancing algorithm for private cloud storage,” *Comput. Commun. Netw. Technol. (ICCCNT),2013 Fourth Int. Conf.*, pp. 1–6, 2013.

[141] Y. Mei, L. Liu, S. Member, and X. Pu, “Performance Analysis of Network I / O Workloads in Virtualized Data Centers,” *IEEE Trans. Serv. Comput.*, vol. 6, no. 1, pp. 48–63, 2013.

[142] P. M. Mohan, D. M. Divakaran, and M. Gurusamy, “Performance study of TCP flows with QoS-supported OpenFlow in data center networks,” *IEEE Int. Conf. Networks, ICON*, pp. 1–6, 2013.

[143] J. Yao, “On-demand optimal cloud service provisioning composition across multi-cloud,” *Proc. - 2013 Int. Conf. Comput. Inf. Sci. ICCIS 2013*, pp. 1566–1569, 2013.

[144] M. Shehada, B. Fu, S. Thakolsri, and W. Kellerer, “QoE-based resource reservation for unperceivable video quality fluctuation during Handover in LTE,” *2013 IEEE 10th Consum. Commun. Netw. Conf. CCNC 2013*, pp. 171–177, 2013.

[145] D. J. Vergados, A. Sgora, A. Michalas, D. D. Vergados, J. P. Laulajainen, and Y. Jiang, “A QoE-driven adaptation scheme for video content delivery in LTE networks,” *Proc. Int. Symp. Wirel. Commun. Syst.*, vol. 9, pp. 773–777, 2013.

[146] T. Specification, “QoS aspects for popular services in mobile networks ; Part 5 : Definition of typical measurement profiles,” vol. 1, pp. 1–22, 2013.

[147] A. Gelberger, N. Yemini, and R. Giladi, “Performance analysis of Software-Defined Networking (SDN),” *Proc. - IEEE Comput. Soc. Annu. Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst. MASCOTS*, pp. 389–393, 2013.

[148] S. Huang and J. Griffioen, “Network hypervisors: Managing the emerging SDN chaos,” *Proc. - Int. Conf. Comput. Commun. Networks, ICCCN*, 2013.

[149] A. Dixit, F. Hao, and S. Mukherjee, “Towards an elastic distributed sdn controller,” *Proc. ...*, pp. 7–12, 2013.

[150] S. Schmid and J. Suomela, “Exploiting locality in distributed SDN control,” *Proc. Second ACM SIGCOMM Work. Hot Top. Softw. Defin. Netw. - HotSDN ’13*, p. 121, 2013.

[151] G. Monteleone and P. Paglierani, “Session Border Controller virtualization towards ‘service- defined’ networks based on NFV and SDN,” *SDN4FNS 2013 - 2013 Work. Softw. Defin. Networks Futur. Networks Serv.*, 2013.

[152] A. Basta, W. Kellerer, M. Hoffmann, K. Hoffmann, and E. D. Schmidt, “A virtual SDN-enabled LTE EPC architecture: A case study for S-/P-gateways

functions," *SDN4FNS 2013 - 2013 Work. Softw. Defin. Networks Futur. Networks Serv.*, 2013.

[153] W. Jiang, D. Ma, and Y. Zhao, "Strategy-based fault handling mechanism for composite service," *Proc. 2013 3rd Int. Conf. Intell. Syst. Des. Eng. Appl. ISDEA 2013*, pp. 1297–1301, 2013.

[154] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 215–228, 2013.

[155] M. F. Bari *et al.*, "Data Center Network Virtualization: A Survey," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 2, pp. 909–928, 2013.

[156] N. Bitar, S. Gringeri, and J. X. Tiejun, "Technologies and protocols for data center and cloud networking," *IEEE Commun. Mag.*, vol. 51, no. 9, pp. 24–31, 2013.

[157] C. J. M. Tauro, N. Ganesan, A. A. Easo, and S. Mathew, "Convergent replicated data structures that tolerate eventual consistency in NoSQL databases," *Proc. - 2013 3rd Int. Conf. Adv. Comput. Commun. ICACC 2013*, pp. 70–75, 2013.

[158] T. Yigit, M. A. Cakar, and A. S. Yuksel, "The experience of NoSQL database in telecommunication enterprise," in *AICT 2013 - 7th International Conference on Application of Information and Communication Technologies, Conference Proceedings*, 2013.

[159] Intel, "Getting Started with Big Data," no. February, 2013.

[160] P. Großkopf, "The Role of Interoperability for eServices," no. 040, 2013.

[161] B. Oselio, A. Kulesza, and A. H. Iii, "Multi-layer graph analytics for dynamic social networks," *arXiv Prepr. arXiv1309.5124*, vol. 8, no. 4, pp. 1–9, 2013.

[162] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds," *2013 IEEE Glob. Conf. Signal Inf. Process. Glob. 2013 - Proc.*, vol. 62, no. 4, pp. 993–996, 2013.

[163] R. Hoettger, B. Igel, and E. Kamsties, "A novel partitioning and tracing approach for distributed systems based on vector clocks," *Proc. 2013 IEEE 7th Int.*

Conf. Intell. Data Acquis. Adv. Comput. Syst. IDAACS 2013, vol. 2, no. September, pp. 670–675, 2013.

[164] M. M. Klymash, I. V. Demydov, O. A. Lavriv, and Y. D. Dobush, "Analysis of service workflows distribution and service delivery platform parameters," *Системи обробки інформації*, no. 6(104), pp. 103–107, 2012.

[165] М. М. Климаш, Б. А. Бугиль, О. А. Лаврів, and Т. В. Корецький, "Інтегральна оцінка ефективності вибору маршрутів передавання потоків даних для різних конфігурацій мережевих топологій," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* no. 738, pp. 95–100, 2012.

[166] О. А. Лаврів, М. І. Бешлей, and Б. А. Бугиль, "Дослідження якості надання послуг абонентам мобільного зв'язку в IMS-мережі," *Збірник наукових праць Інститут проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України*, no. 65, pp. 132–140, 2012.

[167] О. А. Лаврів, М. І. Бешлей, М. М. Гнатчук, and А. В. Поліщук, "Модель системи управління ресурсами мультисервісних мереж в умовах самоподібності трафіку," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* no. 738, pp. 165–173, 2012.

[168] M. Klymash, Y. Hayali, and O. Lavriv, "Models and mechanisms of IPTV VoD Traffic Balancing," *Comput. Probl. Electr. Eng.*, vol. 2, no. 2, pp. 41–45, 2012.

[169] Б. А. Бугиль, М. М. Климаш, О. А. Лаврів, and І. В. Демидов, "Підвищення ефективності розподілу ресурсів телекомунікаційної мережі шляхом зміни маршрутів передавання даних," *Проблеми телекомунікацій*, no. 4(9), pp. 32–44, 2012.

[170] М. М. Климаш, Б. А. Бугиль, and О. А. Лаврів, "Дослідження ефективності топологій телекомунікаційних мереж на основі їх інтегральної оцінки за методом аналізу ієрархій," in *Сучасні інформаційно-комунікаційні технології COMINFO'2012*, 2012, pp. 46–48.

[171] M. Klymash, O. Lavriv, and B. Buhil, “Service-oriented Resource Planning for Multiservice IP Networks,” *Comput. Probl. Electr. Eng.*, vol. 2, no. 1, pp. 59–63, 2012.

[172] О. А. Лаврів, Б. А. Бугиль, and І. Д. Орлевич, “Архітектура і програмна концепція телекомунікаційної платформи SDN – OPENFLOW,” in *Науково-методична конференція “Сучасні проблеми телекомунікацій і підготовка фахівців в галузі телекомунікацій – 2012,”* 2012, pp. 13–15.

[173] О. А. Лаврів, “Сервісно-орієнтоване планування ресурсів мультисервісної телекомунікаційної мережі,” *Збірник наукових праць Інститут проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України*, no. 62, pp. 118–126, 2012.

[174] T. Warabino, S. Kaneko, S. Nanba, and Y. Kishi, “Advanced load balancing in LTE/LTE-A cellular network,” *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC*, pp. 530–535, 2012.

[175] M. Z. Shakir and M. S. Alouini, “On the area spectral efficiency improvement of heterogeneous network by exploiting the integration of macro-femto cellular networks,” *IEEE Int. Conf. Commun.*, pp. 5695–5700, 2012.

[176] J. Yao and J. He, “Load balancing strategy of cloud computing based on artificial bee algorithm,” pp. 185–189, 2012.

[177] H. Ren, Y. Lan, and C. Yin, “The load balancing algorithm in cloud computing environment,” *Proc. 2012 2nd Int. Conf. Comput. Sci. Netw. Technol.*, pp. 925–928, 2012.

[178] D. Taniar, “High Performance Database Processing,” *2012 IEEE 26th Int. Conf. Adv. Inf. Netw. Appl.*, vol. 117, pp. 5–6, 2012.

[179] F. Ma, F. Liu, and Z. Liu, “Distributed load balancing allocation of virtual machine in cloud data center,” *ICSESS 2012 - Proc. 2012 IEEE 3rd Int. Conf. Softw. Eng. Serv. Sci.*, pp. 20–23, 2012.

[180] R. Birke, L. Y. Chen, and E. Smirni, “Data centers in the cloud: A large scale performance study,” *Proc. - 2012 IEEE 5th Int. Conf. Cloud Comput. CLOUD 2012*, pp. 336–343, 2012.

[181] L. Liu, F. Yang, R. Wang, Z. Shi, A. Stidwell, and D. Gu, “Analysis of handover performance improvement in cloud-RAN architecture,” *2012 7th Int. ICST Conf. Commun. Netw. China, CHINACOM 2012 - Proc.*, pp. 850–855, 2012.

[182] L. Globa and T. Kot, “Three- and five-level architectures of Internet based information systems,” in *2010 International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET)*, 2010.

[183] D. Zissis and D. Lekkas, “Addressing cloud computing security issues,” *Futur. Gener. Comput. Syst.*, vol. 28, no. 3, pp. 583–592, 2012.

[184] S. Abolfazli, Z. Sanaei, M. Shiraz, and A. Gani, “MOMCC: Market-oriented architecture for Mobile Cloud Computing based on Service Oriented Architecture,” *Proc. 1st IEEE Int. Conf. Commun. China Work.*, pp. 8–13, 2012.

[185] K. SAITO1l and Y. KAKISHIMA, “Field Experiments on Throughput Performance of Carrier Aggregation with Asymmetric Bandwidth in LTE Advanced,” vol. 7, pp. 1–8, 2012.

[186] M. Han and J. P. Hong, “Balancing QoE and fairness of layered video multicast in LTE networks,” *Adv. Commun. Technol. (ICACT), 2012 14th Int. Conf.*, pp. 387–392, 2012.

[187] Y. He, C. Wang, H. Long, and K. Zheng, “PNN-based QoE measuring model for video applications over LTE system,” *2012 7th Int. ICST Conf. Commun. Netw. China, CHINACOM 2012 - Proc.*, pp. 58–62, 2012.

[188] C. Liang, “Software Defined Network Support for Real Distributed Systems,” pp. 1–10, 2012.

[189] D. Levin, A. Wundsam, B. Heller, N. Handigol, and A. Feldmann, “Logically centralized? State distribution trade-offs in software defined networks,” *HotSDN*, pp. 1–6, 2012.

[190] Z. Liu, S. Li, J. He, D. Xie, and Z. Deng, “Complex Network Security Analysis Based on Attack Graph Model,” *2012 Second Int. Conf. Instrumentation, Meas. Comput. Commun. Control*, pp. 183–186, 2012.

[191] F. Xhafa, “Data Replication and Synchronization in P2P Collaborative Systems,” *2012 IEEE 26th Int. Conf. Adv. Inf. Netw. Appl.*, pp. 7–7, 2012.

[192] J. Cao *et al.*, “Datacast: a scalable and efficient reliable group data delivery service for data centers,” *Proc. 8th Int. Conf. Emerg. Netw. Exp. Technol. - Conex. ’12*, vol. 31, no. 12, p. 37, 2012.

[193] D. Jayathilake, C. Sooriaarachchi, T. Gunawardena, B. Kulasuriya, and T. Dayaratne, “A study into the capabilities of NoSQL databases in handling a highly heterogeneous tree,” *ICIAFS 2012 - Proc. 2012 IEEE 6th Int. Conf. Inf. Autom. Sustain.*, pp. 106–111, 2012.

[194] E. Overview, “Mining Big Data in the Enterprise for Better Business Intelligence,” *Intel IT Best Pract. Bus. Intell.*, no. July, p. 7, 2012.

[195] Teradata, *Big Data Analytics*. 2012.

[196] D. Guide, “Amazon Simple Workflow Service,” 2012.

[197] D. Karastoyanova, D. Dentsas, D. Schumm, M. Sonntag, L. Sun, and K. Vukojevic, “Service-based integration of human users in workflow-driven scientific experiments,” *2012 IEEE 8th Int. Conf. E-Science, e-Science 2012*, 2012.

[198] X. Fei *et al.*, “Experiences Using Cloud Computing for a Scientific Workflow Application,” *Procedia Comput. Sci.*, vol. 10, no. 974, pp. 15–24, 2012.

[199] E. Bauer and R. Adams, “Reliability Analysis of Virtualization,” *Reliab. Availab. Cloud Comput.*, p. 352, 2012.

[200] E. Bauer and R. Adams, “Recommendations for Architecting a Reliable System,” *Reliab. Availab. Cloud Comput.*, pp. 209–243, 2012.

[201] М. М. Климаш, О. А. Лаврів, І. О. Кагало, Б. В. Коваль, and Т. А. Максимюк, “Покращення параметрів радіоінтерфейсу LTE/HSOPA,” *Комп’ютерні технології друкарства*, no. 26, pp. 130–137, 2011.

[202] M. Wallace and L. Webber, *The disaster recovery handbook : a step-by-step plan to ensure business continuity and protect vital operations, facilities, and assets*, 2nd edition. AMACOM, 2011.

[203] P. Pedroso, J. Perelló, M. Klinkowski, D. Careglio, S. Spadaro, and J. Solé-Pareta, “A GMPLS/OBS network architecture enabling QoS-aware end-to-end burst transport,” *2011 IEEE 12th Int. Conf. High Perform. Switch. Routing, HPSR 2011*, pp. 64–69, 2011.

[204] I. Katib and D. Medhi, “A network protection design model and a study of three-layer networks with IP/MPLS, OTN, and DWDM,” *2011 8th Int. Work. Des. Reliab. Commun. Networks*, no. 1, pp. 17–24, 2011.

[205] F. B. F. Shaikh and S. Haider, “Security threats in cloud computing,” *2011 Int. Conf. Internet Technol. Secur. Trans.*, no. December, pp. 214–219, 2011.

[206] M. A. F. Gutierrez and N. Ventura, “Mobile Cloud Computing Based on Service Oriented Architecture : Embracing Network As a Service for 3 Rd Party Application Service Providers,” *Network*, 2011.

[207] ETSI, “ETSI TS 102 250-2 V2.2.1 : Speech and multimedia Transmission Quality (STQ); QoS aspects for popular services in mobile networks; Part 2: Definition of Quality of Service parameters and their computation,” vol. 1, pp. 1–252, 2011.

[208] ETSI, “ETSI TS 102 250-1 V2.2.1 (2011-04) Speech and multimedia Transmission Quality (STQ); QoS aspects for popular services in mobile networks; Part 1: Assessment of Quality of Service,” vol. 1, p. 28, 2011.

[209] T. Specification, “Speech and multimedia Transmission Quality (STQ); QoS aspects for popular services in mobile networks ; Part 4 : Requirements for Quality of Service,” vol. 1, pp. 1–45, 2011.

[210] A. Zaballos, A. Vallejo, and J. Selga, “Heterogeneous communication architecture for the smart grid,” *IEEE Netw.*, vol. 25, no. 5, pp. 30–37, 2011.

[211] M. Pańka and P. Bała, “Data centered collaboration in a mobile environment,” *2011 Fed. Conf. Comput. Sci. Inf. Syst. FedCSIS 2011*, pp. 723–728, 2011.

[212] A. Cuzzocrea, I.-Y. Song, and K. C. Davis, “Analytics over large-scale multidimensional data: the big data revolution!,” *... 14th Int. Work. Data ...*, pp. 101–104, 2011.

[213] H. Herodotou *et al.*, “Starfish: {A} Self-tuning System for Big Data Analytics,” *{CIDR} 2011, Fifth Bienn. Conf. Innov. Data Syst. Res. Asilomar, CA, USA, January 9-12, 2011, Online Proc.*, pp. 261–272, 2011.

[214] R. Sion and J. Tatemura, “Runtime web-service workflow optimization,” *Lect. Notes Bus. Inf. Process.*, vol. 74 LNBIP, pp. 112–131, 2011.

[215] X. Song, W. Dou, and J. Chen, “A workflow framework for intelligent service composition,” *Futur. Gener. Comput. Syst.*, vol. 27, no. 5, pp. 627–636, 2011.

[216] D. Zhang, P. Coddington, and A. Wendelborn, “Web services workflow with result data forwarding as resources,” *Futur. Gener. Comput. Syst.*, vol. 27, no. 6, pp. 694–702, 2011.

[217] J. Zhang, W. Tan, A. John, I. Foster, and R. Madduri, “Recommend-as-you-go: A novel approach supporting services-oriented scientific workflow reuse,” *Proc. - 2011 IEEE Int. Conf. Serv. Comput. SCC 2011*, no. July, pp. 48–55, 2011.

[218] R. Buyya and K. Sukumar, “Platforms for Building and Deploying Applications for Cloud Computing,” *arXiv Prepr. arXiv1104.4379*, p. 6, 2011.

[219] S. Subashini and V. Kavitha, “A survey on security issues in service delivery models of cloud computing,” *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 1–11, 2011.

[220] N. Simoni and Y. Wu, “QoS signaling for Service Delivery in NGN / NGS Context,” no. c, pp. 22–29, 2011.

[221] M. Khan and U. Toseef, “User utility function as quality of experience (QoE),” *ICN 2011, Tenth Int. Conf. Networks*, pp. 99–104, 2011.

[222] H. Zhang, X. Wen, B. Wang, W. Zheng, and Y. Sun, “A Novel Handover Mechanism Between Femtocell and Macrocell for LTE Based Networks,” *2010 Second Int. Conf. Commun. Softw. Networks*, pp. 228–231, 2010.

[223] B. Zhang, J. Gao, and J. Ai, “Cloud Loading Balance algorithm,” *2nd Int. Conf. Inf. Sci. Eng. ICISE2010 - Proc.*, pp. 5001–5004, 2010.

[224] K. Ye, D. Huang, X. Jiang, H. Chen, and S. Wu, “Virtual machine based energy-efficient data center architecture for cloud computing: A performance perspective,” *Proc. - 2010 IEEE/ACM Int. Conf. Green Comput. Commun. GreenCom 2010, 2010 IEEE/ACM Int. Conf. Cyber, Phys. Soc. Comput. CPSCom 2010*, pp. 171–178, 2010.

[225] Y. Ye *et al.*, “A framework for QoS and power management in a service cloud environment with mobile devices,” *Proc. - 5th IEEE Int. Symp. Serv. Syst. Eng. SOSE 2010*, pp. 236–243, 2010.

[226] P. Li and Y. Fang, “The Capacity of Heterogeneous Wireless Networks,” 2010.

[227] H. Packard, “Управление ресурсами в сервис-ориентированных системах типа « приложение как сервис »,” vol. 1, no. 21, pp. 156–160, 2010.

[228] R. Al-Ekram and R. Holt, “Multi-consistency data replication,” *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, pp. 568–577, 2010.

[229] D. Lin and A. Squicciarini, “Data protection models for service provisioning in the cloud,” *Proceeding 15th ACM Symp. Access Control Model. Technol. SACMAT 10*, p. 183, 2010.

[230] B. Callaway, “An Autonomic Service Delivery Platform for Service-Oriented Network Environments Introduction & Motivation,” *Perform. Res.*, vol. 3, no. 2, pp. 2044–2048, 2010.

[231] N. Kryvinska, C. Strauss, B. Collini-Nocker, and P. Zinterhof, “A scenario of service-oriented principles adaptation to the telecom providers service delivery platform,” *Proc. - 5th Int. Conf. Softw. Eng. Adv. ICSEA 2010*, pp. 265–271, 2010.

[232] A. Wong-Jiru, J. Colombi, L. Suzuki, and R. Mills, “Graph Theoretical Analysis of Network Centric Operations Using Multi-Layer Models,” *Proc. 5th Int. Conf. Inf. Warf. Secur.*, vol. 2020, 2010.

[233] Y. Wei and M. B. Blake, “Service-Oriented Computing and Cloud Computing: Challenges and Opportunities,” *IEEE Internet Comput.*, vol. 14, no. 6, pp. 72–75, 2010.

[234] S. H. Chin, T. Suh, and H. C. Yu, *Adaptive service scheduling for workflow applications in Service-Oriented Grid*, vol. 52, no. 3. 2010.

[235] W. Tan, J. Zhang, and I. Foster, “Network Analysis of Scientific Workflows: A Gateway to Reuse,” *Computer (Long. Beach. Calif).*, vol. 43, no. 9, pp. 54–61, 2010.

[236] S. Kounev, F. Brosig, N. Huber, and R. Reussner, “Towards self-aware performance and resource management in modern service-oriented systems,” *Proc. - 2010 IEEE 7th Int. Conf. Serv. Comput. SCC 2010*, pp. 621–624, 2010.

[237] M. Wäljas, “Cross-platform service user experience: a field study and an initial framework,” *... Devices Serv.*, pp. 219–228, 2010.

[238] M. Tang and L. Ai, “A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition,” *IEEE Congr. Evol. Comput.*, no. July, pp. 1–8, 2010.

[239] Y. C. Zhou, L. Xue, X. P. Liu, X. N. Wang, X. X. Liang, and C. H. Sun, “Service storm: A self-service telecommunication service delivery platform with platform-as-a-service technology,” *Proc. - 2010 6th World Congr. Serv. Serv. 2010*, pp. 8–15, 2010.

[240] L. Rodero-Merino *et al.*, “From infrastructure delivery to service management in clouds,” *Futur. Gener. Comput. Syst.*, vol. 26, no. 8, pp. 1226–1240, 2010.

[241] J. Craveur, “Management of triple/quadruple play services from a telecom perspective,” pp. 15–52, 2010.

[242] C.-W. Hang and M. Singh, “From Quality to Utility: Adaptive Service Selection Framework,” *Serv. Comput.*, vol. 6470, pp. 456–470, 2010.

[243] B. Phillips, *Disaster recovery*. CRC Press, 2009.

[244] А. Сегайер, “Численное исследование алгоритмов расчета вероятности потерь для многопотоковых моделей пакетных сетей,” pp. 161–182, 2009.

[245] H. Holma and A. Toskala, “LTE for UMTS-OFDMA and SC-FDMA based radio access,” p. 450, 2009.

[246] Y. Zhao and W. Huang, “Adaptive Distributed Load Balancing Algorithm Based on Live Migration of Virtual Machines in Cloud,” *2009 Fifth Int. Jt. Conf. INC, IMS IDC*, pp. 170–175, 2009.

[247] A. Grzech and P. Swiątek, “Modeling and Optimization of Complex Services in Service-Based Systems.,” *Cybern. {&} Syst.*, vol. 40, no. 8, pp. 706–723, 2009.

[248] A. Grzech and P. Swiątek, “Modeling and Optimization of Complex Services in Service-Based Systems.,” *Cybern. {&} Syst.*, vol. 40, no. 8, pp. 706–723, 2009.

[249] N. Blum, T. Magedanz, F. Schreiner, and S. Wahle, “From IMS management to SOA based NGN management,” *J. Netw. Syst. Manag.*, vol. 17, no. 1–2, pp. 33–52, 2009.

[250] C. Yoon, H. Lee, and W. Lyu, “Design of Layered Service Delivery Platform for Enabling I-centric Convergence Service,” pp. 243–247, 2009.

[251] C. Rudolph, N. Kuntze, and Z. Velikova, “Secure Web Service Workflow Execution,” *Electron. Notes Theor. Comput. Sci.*, vol. 236, no. C, pp. 33–46, 2009.

[252] C. Yoon, H. Lee, and W. Lyu, “Design of Layered Service Delivery Platform for Enabling I-centric Convergence Service,” pp. 243–247, 2009.

[253] S. M. Ross, “Elements of Probability,” *Introd. to Probab. Stat. Eng. Sci.*, pp. 55–88, 2009.

[254] T. Hahn and S. Wan, “An Ant Colony Optimization Implementation for Dynamic Routing and Wavelength Assignment in Optical Networks,” vol. 6, no. 6, pp. 578–589, 2008.

[255] X. Jin, W. Tu, and S. H. G. Chan, “Traceroute-based topology inference without network coordinate estimation,” *IEEE Int. Conf. Commun.*, pp. 1615–1619, 2008.

[256] S. Processing, “QoS aspects for popular services in GSM and 3G networks; Part 2 : Definition of Quality of Service parameters,” vol. 2, pp. 1–196, 2008.

[257] K. Lau and C. Tran, “Composite web services,” *Emerg. Web Serv. Technol. Vol. II*, 2008.

[258] K. Lakshminarayanan, D. Adkins, A. Perrig, and I. Stoica, “Securing user-controlled routing infrastructures,” *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 549–561, 2008.

[259] N. Dreihann-holenia, “Service-Oriented Architecture Enabling Flexible , Customer Centric Virtual Logistic Enterprise Table of Content,” 2008.

[260] D. X. D. Xu, Y. L. Y. Li, M. C. M. Chiang, and a. Calderbank, “Elastic service availability: utility framework and optimal provisioning,” *IEEE J. Sel. Areas Commun.*, vol. 26, no. 6, pp. 55–65, 2008.

[261] G. O’Driscoll, *Next Generation IPTV Services and Technologies*. 2007.

[262] Б.Ю. Жураковський, “Розробка структури програмного забезпечення при об’єктному підході,” *Системи управління, навігації та зв’язку*, no. 4(32), pp. 165–170, 2014.

[263] C. J. Pavlovski, “Service delivery platforms in practice [IP Multimedia Systems (IMS) Infrastructure and Services],” *Commun. Mag. IEEE*, vol. 45, no. March, pp. 114–121, 2007.

[264] І. С. Катеринчук and В. М. Періг, “Удосконалення аналітичних залежностей показників якості функціонування телекомунікаційних систем з урахуванням умов невизначеності,” *Збірник наукових праць НАДПСУ*, no. 57, pp. 75–79, 2014.

[265] H. Ohnishi, Y. Yamato, M. Kaneko, T. Moriya, M. Hirano, and H. Sunaga, “Service delivery platform for telecom-enterprise-Internet combined services,” *GLOBECOM - IEEE Glob. Telecommun. Conf.*, pp. 108–112, 2007.

[266] С. В. Толюпа, Ю. П. Недайбіда, and Ю. В. Котова, “Современные проблемы создания сложных ин-формационно-управляющих систем реального времени в условиях конфликта,” Сучасні інформаційні технології у сфері безпеки та оборони, no. 2(20), pp. 100–106, 2014.

[267] A. R. Calderbank, F. Ieee, and J. C. Doyle, “Layering as Optimization Decomposition : A Mathematical Theory of Network Architectures,” vol. 95, no. 1, 2007.

[268] M. Fowler, “Continuous Integration,” 2006. [Online]. Available: http://www.dccia.ua.es/dccia/inf/asignaturas/MADS/2013-14/lecturas/10_Fowler_Continuous_Integration.pdf. [Accessed: 25-Jun-2017].

[269] С. В. Грудинин and Д. Ю. Черников, “Технология конвергенции сетей фиксированной и мобильной связи,” pp. 2241–2249, 2006.

[270] M. Fiedler, S. Chevul, O. Radtke, K. Tutschku, and A. Binzenhäfer, “The Network Utility Function : A Practicable Concept for Assessing Network Impact on Distributed Services,” *Telecommun. Syst.*, pp. 1–13, 2005.

[271] Б. А. Бугиль, “Підвищення ефективності методів динамічної маршрутизації в телекомунікаційних мережах,” 2013.

[272] А. В. Лемешко, “Тензорная модель многопутевой маршрутизации агрегированных потоков с резервированием сетевых ресурсов , представленная в пространстве с кривизной,” *Праці Ундірт*, vol. 4, no. 40, pp. 12–18, 2004.

[273] T. A. Weiss and F. K. Jondral, “Spectrum pooling: An innovative strategy for the enhancement of spectrum efficiency,” *IEEE Commun. Mag.*, vol. 42, no. 3, pp. 8–14, 2004.

[274] І. В. Демидов, “Синтез телекомунікаційних сервісних платформ національного масштабу,” 2017.

[275] H. Kopetz and A. Hanzlik, “Integration of Internal and External Clock Synchronization by the Combination of Clock-State and Clock-Rate Correction in Fault-Tolerant Distributed Systems,” *Real-Time Syst.*, no. Rtss, 2004.

[276] Y. Liang, “Real-Time VBR Video Traffic Prediction for Dynamic Bandwidth Allocation,” *IEEE Trans. Syst.*, vol. 34, no. 1, pp. 32–47, 2004.

[277] W. E. Walsh, G. Tesauro, J. O. Kephart, and R. Das, “Utility functions in autonomic systems,” *Int. Conf. Auton. Comput. 2004. Proceedings.*, pp. 70–77, 2004.

[278] I. Cisco System, *Internetworking Technologies Handbook, Fourth Edition*. 2003.

[279] A. Jain, A. Karandikar, and R. Verma, “An adaptive prediction based approach for congestion estimation in active queue management (APACE),” *GLOBECOM ’03. IEEE Glob. Telecommun. Conf. (IEEE Cat. No.03CH37489)*, vol. 7, pp. 4153–4157, 2003.

[280] L. Du, J. Bigham, and L. Cuthbert, “Towards Intelligent Geographic Load Balancing for Mobile Cellular Networks,” *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 33, no. 4, pp. 480–491, 2003.

[281] G. Connolly, A. Sachenko, and G. Markowsky, “Distributed traceroute approach to geographically locating IP devices,” *Proc. 2nd IEEE Int. Work. Intell. Data Acquis. Adv. Comput. Syst. Technol. Appl. IDAACS 2003*, no. September, pp. 128–131, 2003.

[282] H. W. Park, “Hyperlink network analysis: A new method for the study of social structure on the web,” *Connections*, vol. 25, no. 1, pp. 49–61, 2003.

[283] О. М. Шпур, “Підвищення якості надання композитних сервісів у мережах із сервісно-орієнтованою архітектурою,” 2017.

[284] J. Zhou, “A study of dynamic routing and wavelength assignment with imprecise network state information,” *Proceedings. Int. Conf. Parallel Process. Work.*, pp. 207–213, 2002.

[285] М. І. Бешлей, “Підвищення якості обслуговування пріоритетного трафіку реального часу у вузлах мультисервісної мережі,” 2015.

[286] G. Candea and A. Fox, “A utility-centered approach to building dependable infrastructure services,” *Proc. 10th Work. ACM SIGOPS Eur. Work. beyond PC - EW’10*, no. September, pp. 213–218, 2002.

[287] D. M. Karsten, “QoS Signalling and Charging in a Multi-service Internet using RSVP,” *Control*, 2000.

[288] C. Qiao, H. Wu, and O. Tonguz, “Load balancing via relay in next generation wireless systems,” *Mob. Ad Hoc Netw. …*, vol. 2, pp. 149–150, 2000.

[289] A. Patel and C. Williamson, “Statistical Multiplexing of Self-Similar Traffic: Theoretical and Simulation Results,” pp. 1–14, 1997.

[290] Б. М. Стрихалюк, “Структурний та функціональний синтез гетерогенних сервісно-орієнтованих телекомунікаційних мереж,” 2015.

[291] A. Olson and K. G. Shin, “Probabilistic clock synchronization in large distributed systems,” *IEEE Trans. Comput.*, vol. 43, no. 9, pp. 1106–1112, 1994.

[292] D. P. Bertsekas, “Auction algorithm for network flow problems: A tutorial introduction,” *Comput. Optim. Appl.*, vol. 1, no. 1, pp. 7–66, 1992.

[293] A. M. Gibbons, “Algorithmic graph theory,” *Networks*, vol. 13, no. 2, pp. 304–305, 1985.

ДОДАТОК А. СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ ТА ВІДОМОСТІ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

[1] B. Buhyl, P. Huskov, O. Lavriv, R. Bak, A. Luntovskyy, “Maximization of Service Flows Rates as a Solution of Network Capacity Allocation Problem,” *Internet Things Eng. Appl.*, vol. 3, no. 1, pp. 1–10, May 2018. (іноземне наукове періодичне видання з напряму, ISSN 2371-8609(Print))

[2] P. Yatskiv, O. Lavriv, Z. Kharkhalis, V. Mosorov, and M. Niedźwiedziński, “Security. Provisions for Ensuring Telecommunication Networks’ Protection from Malicious Influences,” *Przedsiębiorczość i Zarządzanie*, vol. 17, no. 11, pp. 111–125, 2016. (іноземне наукове періодичне видання з напряму, ISSN 1733-2486)

[3] Р. П. Кришталь, О. А. Лаврів, З. М. Хархаліс, “Удосконалення алгоритму повного групового розсилання у системах оброблення даних з розподіленою архітектурою,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 849, С. 241–247, 2016. (наукове фахове видання, МНБ ІСІ, ISSN 0321-0499)

[4] Р. І. Бак, П. О. Гуськов, О. А. Лаврів, “Імітаційна макромодель поведінки абонентів у мережі коміркового зв’язку,” *Вісник Національного університету “Львівська політехніка”, серія “Радіоелектроніка та телекомунікації,”* №. 849, С. 274–284, 2016. (наукове фахове видання, МНБ ІСІ, ISSN 0321-0499)

[5] M. Beshley, M. Seliuchenko, O. Lavriv, V. Chervenets, H. Kholiavka, M. Klymash, “Increasing the efficiency of real-time content delivery by improving the technology of priority assignment and processing of IP traffic,” *Smart Comput. Rev.*, vol. 5, no. 2, pp. 1–13, 2015. (електронне іноземне наукове періодичне видання з напряму ISSN 2234-4624)

[6] O. Shpur, M. Klymash, M. Seliuchenko, B. Strykhaliuk, O. Lavriv, “Improving the Quality of Composite Services Through Improvement of Cloud

Infrastructure Management," *Int. J. Comput. Sci. Inf. Secur.*, vol. 13, no. 9, pp. 36–44, 2015. (іноземне наукове періодичне видання за напряму ISSN 1947-5500)

[7] М. І. Бешлей, М. О. Селюченко, О. А. Лаврів, А. Р. Масюк, Г.В. Холявка, "Оцінка адекватності функціонування програмного маршрутизатора у процесі обслуговування мультимедійного трафіку," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 818, С. 162–173, 2015. (наукове фахове видання, МНБ ICI, ISSN 0321-0499)

[8] M. M. Klymash, H. A. Al-Zayadi, O. A. Lavriv, "Improving Thoughput Using Channel Quality Indicator in LTE Technology," *Збірник наукових праць Донецького національного технічного університету, серія "Обчислювальна техніка та автоматизація,"* №. 1(26), С. 134–144, 2014. (наукове фахове видання ISSN 2075-4272)

[9] М. М. Климаш, М. І. Бешлей, Б. М. Стрихалюк, О. А. Лаврів, Г.В. Холявка,"Підвищення якості обслуговування в конвергентних мобільних системах на основі платформи UMA-A," *Проблеми телекомунікацій*, №. 1(13), С. 3–19, 2014. (електронне наукове фахове видання, індексується МНБ РИНЦ ISSN 2220-6922)

[10] М. І. Бешлей, В. П. Ткачук, Б. А. Бугиль, О. А. Лаврів, "Модель системи динамічного управління пропускною здатністю каналу інтегрованої WI-FI/GSM мережі," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 796, С. 83–96, 2014. (наукове фахове видання, МНБ ICI, ISSN 0321-0499)

[11] H. A. Al-Zayadi, M. M. Klymash, O. A. Lavriv, M. Al-Shuraifi, "Mobility Affected on Channel Estimation Using Different Modulation in LTE," *Проблеми телекомунікацій*, no. 2(14), pp. 30–41, 2014. (електронне наукове фахове видання, індексується МНБ РИНЦ ISSN 2220-6922)

[12] O. Yaremko, B. Stryhalyuk, T. Maksymyuk, O. Lavriv, D. Kozhurov, "The optimal power control method in multiuser cellular networks," *An Int. Q. J. Econ.*

Technol. New Technol. Model. Process., vol. 2, no. 1, pp. 63–67, 2013. (іноземне наукове періодичне видання за напрямом ISSN 2084–5715)

[13] Б. А. Бугиль, О. А. Лаврів, М. І. Бешлей, В. В. Червенець, "Методи оптимізації фізичної та логічної структур телекомунікаційних мереж," *Вісник* Національного *університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 766, С. 76–81, 2013. (наукове фахове видання, МНБ ІСІ, ISSN 0321-0499)

[14] І. М. Кузь, З. М. Хархаліс, П. О. Гуськов, О. А. Лаврів, "Дослідження методів частотного планування коміркових мереж на основі технологій LTE та GSM," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 766, С. 82–87, 2013. (наукове фахове видання, МНБ ІСІ, ISSN 0321-0499)

[15] Y. Hayali, O. Lavriv, B. Buhyl, R. Bak, M. Klymash, "Models and mechanisms for traffic balancing of IPTV VoD service," *Int. J. Serv. Econ. Manag.*, vol. 5, no. 4, pp. 291–300, 2013. (іноземне наукове періодичне видання з напряму ISSN print: 1753-0822)

[16] М. М. Климаш, Б. А. Бугиль, О. А. Лаврів, Т. В. Корецький, "Інтегральна оцінка ефективності вибору маршрутів передавання потоків даних для різних конфігурацій мережевих топологій," *Вісник Національного університету "*Львівська *політехніка", серія "Радіоелектроніка та телекомунікації,"* №. 738, С. 95–99, 2012. (наукове фахове видання, ISSN 0321-0499)

[17] О. А. Лаврів, М. І. Бешлей, М. М. Гнатчук, А. В. Поліщук, "Модель системи управління ресурсами мультисервісних мереж в умовах самоподібності трафіку," *Вісник Національного університету "Львівська політехніка", серія "Радіоелектроніка та* телекомунікації*,"* №. 738, С. 165–172, 2012. (наукове фахове видання, ISSN 0321-0499)

[18] M. M. Klymash, I. V. Demydov, O. A. Lavriv, Y. D. Dobush, "Analysis of service workflows distribution and service delivery platform parameters," *Системи*

обробки інформації, №. 6(104), С. 103–107, 2012. (наукове фахове видання, індексується МНБ РИНЦ, ISSN 1681-7710)

[19] О. А. Лаврів, М. І. Бешлей, Б. А. Бугиль, "Дослідження якості надання послуг абонентам мобільного зв'язку в IMS-мережі," *Збірник наукових праць Інститут проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України*, № 65, С. 132–140, 2012. (наукове фахове видання, ISSN 2309-7655)

[20] Б. А. Бугиль, М. М. Климаш, О. А. Лаврів, І. В. Демидов, "Підвищення ефективності розподілу ресурсів телекомунікаційної мережі шляхом зміни маршрутів передавання даних," *Проблеми телекомунікацій*, №. 4(9), С. 32–44, 2012. (електронне наукове фахове видання, індексується МНБ РИНЦ ISSN 2220-6922)

[21] M. Klymash, Y. Hayali, O. Lavriv, "Models and mechanisms of IPTV VoD Traffic Balancing," *Comput. Probl. Electr. Eng.*, vol. 2, no. 2, pp. 41–45, 2012. (наукове видання з напряму, МНБ ІСІ, ISSN 2224-0977)

[22] О.А. Лаврів, "Сервісно-орієнтоване планування ресурсів мультисервісної телекомунікаційної мережі," *Збірник наукових праць Інститут проблем* моделювання *в енергетиці ім. Г.Є. Пухова НАН України*, №. 62, С. 118–126, 2012. (наукове фахове видання, ISSN 2309-7655)

[23] M. Klymash, O. Lavriv, B. Buhil, "Service-oriented Resource Planning for Multiservice IP Networks," *Comput. Probl. Electr. Eng.*, vol. 2, no. 1, pp. 59–63, 2012. (наукове видання з напряму, МНБ ІСІ, ISSN 2224-0977)

[24] М. Климаш, А. Ложковський, О. Лаврів, Р. Бак, "Графо-аналітичний підхід до порівняння ефективності систем безпровідного зв'язку," *Комп'ютерні технології друкарства*, №. 27, С. 189–193, 2012. (наукове фахове видання ISSN 2411-9210)

[25] М. М. Климаш, О. А. Лаврів, І. О. Кагало, Б. В. Коваль, Т. А. Максимюк, "Покращення параметрів радіоінтерфейсу LTE/HSOPA," *Комп'ютерні технології* друкарства, №. 26, С. 130–137, 2011. (наукове фахове видання ISSN 2411-9210)

Наукові праці, які засвідчують апробацію матеріалів дисертації:

[26] Y. Hayali, M. Klymash, O. Lavriv, “Analysis of IPTV VoD Traffic Balancing Mechanisms,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 140–142. (заочна участь, доповідь співавтора)

[27] М. М. Климаш, О. А. Лаврів, М. Бешлей, “Моделювання та прогнозування параметрів і структур мультисервісної мережі з диференційованим обслуговуванням послуг,” в *VI Міжнародний науково-технічний симпозіум «Нові технології в телекомунікаціях»*, 2013, С. 28–31. (заочна участь, доповідь співавтора)

[28] М. М. Климаш, Б. А. Бугиль, О. А. Лаврів, “Дослідження ефективності топологій телекомунікаційних мереж на основі їх інтегральної оцінки за методом аналізу ієрархій,” в *Сучасні інформаційно-комунікаційні технології COMINFO’2012*, 2012, С. 46–48. (заочна участь, доповідь співавтора)

[29] О. А. Лаврів, Б. А. Бугиль, І. Д. Орлевич, “Архітектура і програмна концепція телекомунікаційної платформи SDN – OPENFLOW,” в *Науково-методична конференція “Сучасні проблеми телекомунікацій і підготовка фахівців в галузі телекомунікацій – 2012,”* 2012, С. 13–15. (очна участь із доповіддю)

[30] H. A. Al-Zayadi, O. Lavriv, M. Klymash, “QoE Estimation on the Basis of LTE Service Architecture,” in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 590–592. (очна участь із доповіддю)

[31] М. І. Бешлей, О. А. Лаврів, Г. В. Холявка, “Дослідження методів побудови конвергентної мережі оператора мобільного зв’язку для надання послуг quad play,” в *Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій*, 2014, С. 76–77. (заочна участь, доповідь співавтора)

[32] О. В. Красько та О. А. Лаврів, “Адаптація смуги пропускання в мережах NG-SDH/WDM,” у Всеукраїнська науково-практична конференція “Сучасні

проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,” 2014, С. 201–204. (очна участь із доповіддю)

[33] M. Seliuchenko, A. Kovalchuk, O. Lavriv, and M. Klymash, “Enhancing Reliability of Transport Software-Defined Networks Using Flow Table Mutual Reservation Method,” in *12th International Conference Modern Problems of Radio Engineering, Telecommunications and Computer Science Dedicated to the 170th Anniversary of Lviv Polytechnic National University*, 2014, pp. 573–575. (очна участь із доповіддю)

[34] І. В. Демидов, О. А. Лаврів, О. М. Шпур, and М. О. Селюченко, “Доступність композитних застосувань у сервісно-орієнтованих системах,” в *Вимірювальна та обчислювальна техніка в технологічних процесах*, 2014, С. 119–122. (заочна участь, доповідь співавтора)

[35] З. М. Хархаліс and О. А. Лаврів, “Порівняльний аналіз методів побудови мультиоператорних мереж мобільного зв’язку,” *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, С. 249–252. (очна участь із доповіддю)

[36] M. Klymash, M. Beshley, and O. Lavriv, “Model of Network Resources Management on the Basis of Services Priorities Association,” in *12 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2013, pp. 164–166. (заочна участь, доповідь співавтора)

[37] O. Lavriv, I. Kahalo, and R. Kolodiy, “Application of NoSQL approach in data-centered network architectures: big data case,” in *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, pp. 103–105. (очна участь із доповіддю)

[38] О. А. Лаврів, З. М. Хархаліс, and В. Я. Мосоров, “Моделювання DDoS атаки на веб-сервер,” в *10-а Міжнародна науково-технічна конференціїя "Проблеми телекомунікацій – 2016"*, 2016, С. 348–351. (заочна участь, доповідь співавтора)

[39] M. Al-Shuraifi, H. Al-Zayadi, O. Lavriv, and M. Klymash, “Improving QoS in MAX C/I Scheduling Using Resource Allocation Type 1 of LTE,” in *13 International Conference The Experience of Designing and Application of CAD Systems in Microelectronics*, 2015, pp. 12–14. (заочна участь, доповідь співавтора)

[40] Д. В. Кожуров, О. А. Лаврів, and А. Р. Масюк, “Модель обміну керуючою інформацією в сервісно-орієнтованій Cloud-мережі,” в *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2013,”* 2013, С. 105–108. (очна участь із доповіддю)

[41] О. А. Лаврів, Б. А. Бугиль, П. О. Гуськов, Р. Р. Баса, “Розподіл ємності телекомунікаційної мережі із максимізацією інтенсивності інформаційних потоків,” в *“Фізико-технологічні проблеми передавання, оброблення та зберігання інформації в інфокомунікаційних системах,”* 2016, С. 134–135. (заочна участь)

[42] О. А. Лаврів, Б. М. Стрихалюк, О. М. Шпур, and Т. А. Максимюк, “Дистанційний моніторинг параметрів місця вчинення злочину з використанням мобільних технологій,” в *2015 1st International Conference on Advanced Information and Communication Technologies-2015 (AICT’2015)*, 2015, С. 67–69. (очна участь із доповіддю)

[43] Б. М. Стрихалюк, О. А. Лаврів, З. М. Хархаліс, “Проблематика розгортання SDP для операторів мобільного зв’язку на основі віртуалізованих мереж дата-центрів,” в *Всеукраїнська науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, С. 221–224. (очна участь із доповіддю)

[44] М. М. Климаш, М. О. Селюченко, О. А. Лаврів, “Забезпечення відмовостійкості багаторівневої ієрархії управління у програмно-керованих мережах,” в Всеукраїнська *науково-практична конференція “Сучасні проблеми телекомунікацій та підготовка фахівців в галузі телекомунікацій – 2014,”* 2014, С. 225–228. (очна участь із доповіддю)

[45] O. Lavriv, M. Klymash, G. Grynkevych, O. Tkachenko, V. Vasylenko, “Method of cloud system disaster recovery based on ‘Infrastructure as a code’ concept,” in *14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 1139–1142. (очна участь із доповіддю)

[46] R. Bak, O. Lavriv, B. Koval, “Load balancing based on multi-hop handover for wireless cellular networks,” in *2017 IEEE 1st Ukraine Conference on Electrical and Computer Engineering, UKRCON 2017 - Proceedings*, 2017, pp. 1103–1106. (заочна участь, доповідь співавтора)

[47] O. Krasko, M. Klymash, O. Lavriv, “Data flows transmission models in converged optical access networks,” in 2015 2nd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, PIC S and T 2015 - Conference Proceedings, 2015, pp. 65–68. (заочна участь, доповідь співавтора)

[48] O. Lavriv, Z. Kharkhalis, B. Strykhaliuk, “Concept of intrusion detection system with implementation of topological sorting,” in *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings*, 2017, pp. 170–172. (заочна участь, доповідь співавтора)

[49] M. Seliuchenko, O. Lavriv, O. Panchenko, V. Pashkevych, “Enhanced multi-commodity flow model for QoS-aware routing in SDN,” in *2016 IEEE International Scientific Conference “Radio Electronics and Info Communications”, UkrMiCo 2016 - Conference Proceedings*, 2016, pp. 1–3. (заочна участь, доповідь співавтора)

[50] I. Demydov, O. Lavriv, Z. Kharkhalis, M. M. El-Hatri, “Concept of the migrating firewall to scalable cloud networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of the 13th International Conference on TCSET 2016*, 2016, pp. 643–645. (очна участь, доповідь співавтора)

[51] P. Huskov, O. Lavriv, M. Klymash, “The concept of services assurance in heterogeneous service-oriented systems,” in 2016 3rd International Scientific-Practical

Conference Problems of Infocommunications Science and Technology, PIC S and T 2016 - Proceedings, 2017, pp. 24–26. (заочна участь, доповідь співавтора)

[52] O. Lavriv, B. Buhyl, P. Huskov, R. Bak, “Heterogeneous network capacity distribution among service flows,” in 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2017 - Proceedings, 2017, pp. 173–175. (заочна участь, доповідь співавтора)

[53] A. Masiuk, M. Beshley, O. Lavriv, Y. Deschynskiy, “Common radio resource management model for heterogeneous cellular networks,” in *Modern Problems of Radio Engineering, Telecommunications and Computer Science, Proceedings of* the *13th International Conference on TCSET 2016*, 2016, pp. 661–663. (очна участь, доповідь співавтора)

[54] O. Lavriv, B. Buhyl, M. Klymash, G. Grynkevych, “Services continuous integration based on modern free infrastructure,” in *2017 2nd International Conference on Advanced* Information *and Communication Technologies (AICT)*, 2017, pp. 150–153. (очна участь із доповіддю)

[55] M. O. Seliuchenko, O. A. Lavriv, M. M. Klymash, “Analysis of load balancing methods for improving of utilization efficiency of hardware resources of distributed systems,” in *Microwave and Telecommunication Technology (CriMiCo), 2013 23rd International Crimean Conference*, 2013, pp. 515–516. (заочна участь, доповідь співавтора)

[56] H. Al-Zayadi, O. Lavriv, M. Klymash, A.-S. Mushtaq, “Increase throughput by expectation channel quality indicator,” in *2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology*, 2014, pp. 120–121. (заочна участь, доповідь співавтора)

[57] M. Klymash, O. Lavriv, T. Maksymyuk, M. Beshley, “State of the art and further development of information and communication systems,” in *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, 2016, pp. 1–6. (заочна участь, доповідь співавтора)

ДОДАТОК Б. АКТИ ПРО ВИКОРИСТАННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОЇ РОБОТИ

"ЗАТВЕРДЖУЮ"
Проректор з науково-педагогічної роботи
Національного університету
"Львівська політехніка"

_____ доц. Давидчак О.Р.
_____ 2018 р.

АКТ

про використання результатів дисертаційної роботи Лавріва О.А. **"Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах"** у навчальному процесі кафедри телекомунікацій

Даний акт складений комісією у складі:

- д.т.н. Стрихалюк Б.М., заступник директора Інституту телекомунікацій, радіоелектроніки та електронної техніки;
- к.т.н., доц. Колодій Р.С., заступник декана базової вищої освіти Інституту телекомунікацій, радіоелектроніки та електронної техніки;
- д.т.н., проф. Климаш М.М., завідувач кафедри телекомунікацій

про те, що в навчальному процесі кафедри телекомунікацій використано результати дисертаційної роботи Лавріва О.А. "Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах", а саме:

- розроблено новий розділ лекційного курсу «Розподілені сервісні системи та Cloud-технології», що стосується архітектур та комунікаційних шаблонів розподілених інформаційно-телекомунікаційних систем для студентів-магістрів спеціальності 172 – телекомунікації та радіотехніка (спеціалізація «Інформаційні мережі зв'язку»);
- розроблено новий розділ лекційного курсу «Розподілені сервісні системи та Cloud-технології», що стосується особливостей побудови та поведінки cloud-систем під впливом зростання навантаження для студентів-магістрів спеціальності 172 – телекомунікації та радіотехніка (спеціалізація «Інформаційні мережі зв'язку»).

Члени комісії: _____ Стрихалюк Б.М.
_____ Колодій Р.С.
_____ Климаш М.М.

"ЗАТВЕРДЖУЮ"
Директор
ТзОВ "Телекомунікаційна компанія"
_____Пентак І.М.
"12" 04 2018 р.

АКТ

про використання результатів дисертаційної роботи докторанта
Національного університету "Львівська політехніка"
Лавріва Ореста Андрійовича
на тему "**Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах**"

Даний акт складений про те, що у ТзОВ "Телекомунікаційна компанія" використані такі результати дисертаційної роботи Лавріва О.А:

- метод передавання інформації у процесі надання інформаційно-комунікаційних послуг з урахуванням функцій корисності та параметрів процесу пересилання запиту на інформаційно-комунікаційну послугу, який забезпечив підвищення ефективності використання ресурсів інформаційно-телекомунікаційної системи компанії;
- підтверджено можливість використання програмних компонентів інформаційно-телекомунікаційної системи для обслуговування інформаційних потоків з адаптацією якості обслуговування.

Результати експериментальних досліджень, виконаних на виробничих потужностях ТзОВ "Телекомунікаційна компанія", відповідають результатам досліджень, що представлені у дисертаційній роботі, а похибка не перевищує 5%.

Головний інженер І.М. Рубаха

"ЗАТВЕРДЖУЮ"
Заступник директора
ТОВ «Інформконсалт»
_____ Матіїшин Л.З.
"03" 04 2018 р.

АКТ

про використання результатів дисертаційної роботи Лавріва О.А.
"Методи та моделі надання послуг в гетерогенних розподілених
інформаційно-телекомунікаційних системах"

Даний акт складений про те, що у ТОВ «Інформконсалт» використані такі результати дисертаційної роботи Лавріва О.А.:

- комплексна методика об'єктно-орієнтованого навантажувального тестування для оцінки основних показників використання інфраструктури та якості сприйняття послуг, які надаються користувачам під впливом зростання абонентського навантаження.
- територіально-залежна модель надання інформаційно-телекомунікаційних послуг, яка дала знизити втрати користувацьких сесій.
- алгоритм автоматичного відновлення cloud інфраструктури на основі концепції IaaC, який дав змогу суттєво скоротити тривалість відновлення після явищ катастрофічного характеру.

Отримані дані після застосування вказаних результатів співпадають з чисельними показниками, вказаними у дисертаційній роботі. Середньоквадратичне відхилення результатів не перевищує 3%.

Консультант в галузі електроніки
та телекомунікацій Заблоцький С.О.

"ЗАТВЕРДЖУЮ"
Директор
ПП "Цифрові технології"
_____ Танчак З.В.
"11" 04 2018 р.

АКТ

про використання результатів дисертації Лавріва О.А. **"Методи та моделі надання послуг в гетерогенних розподілених інформаційно-телекомунікаційних системах"**

Даний акт складений про те, що у ПП "Цифрові технології" використані такі результати дисертації Лавріва О.А.:

- модель корпоративного клієнта та модель спільної cloud-інфраструктури системи надання інформаційно-комунікаційних послуг, які дали змогу оцінити вплив попиту корпоративних клієнтів на послуги до початку розгортання cloud-інфраструктури;
- метод адаптації каналу передавання в умовах надання мультисервісних послуг з гарантованою якістю обслуговування, який дав змогу оцінити досяжну якість надання послуг для кожного типу сервісу, який надається в інформаційно-телекомунікаційній системі, в умовах спільного використання системних ресурсів;
- методику розгортання системи захисту інформаційно-телекомунікаційної інфраструктури від зловмисних мережних атак у вигляді послуги IaaS, яка дала змогу уникнути проникнення зловмисних запитів до корпоративних серверів.

Внаслідок перевірки вказаних результатів на мережному обладнанні у ПП "Цифрові технології" встановлено, що результати знаходяться в межах п'ятивідсоткового середньоквадратичного відхилення від поданих у дисертації.

Провідний інженер Дрофяк А.М.