УДК 004.946

ВИКОРИСТАННЯ P2P ТА КЛІЄНТ-СЕРВЕРНИХ МОДЕЛЕЙ У БАГАТОКОРИСТУВАЦЬКИХ ІГРАХ

Кулик О.О.
Науковий керівник – асист. Солодов В.Д.
Харківський національний університет радіоелектроніки, каф. МІРЕС
м. Харків, Україна, email: d_res@nure.ua

This paper explores the evolving landscape of multiplayer game architecture, highlighting the transition from traditional client-server models to innovative peer-to-peer (P2P) frameworks and the emergence of hybrid systems. It delves into the technical challenges and solutions inherent in designing scalable, efficient, and secure multiplayer experiences.

Багатокористувацький режим у відеоіграх, також відомий як multiplayer, дозволяє гравцям взаємодіяти в віртуальному просторі через індивідуальні копії гри на персональних пристроях. Для встановлення з'єднання між гравцями використовуються централізовані сервери або прямі підключення, де пристрої обмінюються інформацією про дії гравців для забезпечення синхронізації ігрового процесу.

В архітектурі багатокористувацької гри виділяються дві основні моделі: Peer-to-Peer (P2P) і клієнт-сервер. P2P модель забезпечує децентралізацію та масштабованість, в той час як клієнт-серверна пропонує централізоване управління та стабільність. Далі в статті більш детально розглянуто переваги та виклики кожної з моделей.

Peer-to-peer (P2P) архітектура є ключовим елементом в проектуванні мережевих систем, включаючи багатокористувацькі ігри, де кожен учасник (пір) має можливість безпосередньо взаємодіяти з іншими учасниками без необхідності використання централізованого сервера. Ця модель сприяє створенню децентралізованих мереж, які можуть підвищити ефективність використання ресурсів та знизити залежність від одного контрольного вузла.

Архітектура P2P забезпечує масштабованість без централізованого сервера, знижуючи ризик вузьких місць і покращуючи продуктивність при додаванні нових учасників. Відмовостійкість підвищується за рахунок розподіленості, де відсутність окремих вузлів не впливає на роботу всієї системи. Зниження затримок досягається через прямий обмін даними між клієнтами, особливо ефективний для фізично близько розташованих учасників. Економія ресурсів відбувається завдяки використанню обчислювальних можливостей учасників мережі замість централізованих серверів. Однак, P2P стикається з викликами у забезпеченні безпеки даних через децентралізацію, вимагає складних алгоритмів синхронізації для управління станом гри між учасниками, та може призвести до нерівномірного розподілу навантаження.

P2P архітектура знаходить своє застосування в різноманітних областях, включаючи файлообмінні мережі, децентралізовані соціальні мережі, блокчейн-технології та, звісно, в багатокористувацьких іграх. У контексті ігор, P2P підходить для створення локальних мережевих ігор або онлайн ігор з невеликою кількістю учасників, де важливо знизити затримку та оптимізувати використання ресурсів.

Клієнт-серверна архітектура є основою багатьох сучасних мережевих систем, включаючи веб-сайти, корпоративні додатки, та інтернет речей (IoT). В цій моделі, клієнти (наприклад, веб-браузери або мобільні додатки) ініціюють запити до серверів, які обробляють ці запити та повертають відповідні результати. Централізація ресурсів та обробки даних на сервері дозволяє ефективно керувати великими обсягами даних та забезпечувати безпеку стану системи.

Клієнт-серверна архітектура вирізняється централізованим управлінням даними, де сервер служить основним джерелом інформації, що сприяє легкій синхронізації даних. Це також спрощує впровадження заходів безпеки, оскільки зусилля можуть бути зосереджені на захисті одного сервера, забезпечуючи контроль доступу та шифрування. Легкість розширення та оновлення є іншою перевагою, оскільки зміни в програмному забезпеченні вимагають втручання лише на сервері, не торкаючись клієнтських пристроїв. Серверна оптимізація дозволяє ефективно обробляти дані, мінімізуючи вимоги до обчислювальної потужності клієнтських пристроїв.

Однак, масштабування клієнт-серверних систем може зіткнутися з проблемами через потенційні вузькі місця на серверах, що вимагає значних інвестицій у апаратне забезпечення або хмарні ресурси для підтримки зростаючого обсягу користувачів та запитів. Відмовостійкість також стає викликом, оскільки проблеми з сервером можуть призвести до загальної недоступності послуг. Крім того, залежність від мережевого з'єднання обмежує доступність даних та функцій у випадках поганого інтернет-зв'язку, особливо у віддалених або погано підключених регіонах.

Клієнт-серверна архітектура широко застосовується у веб-розробці, корпоративних додатках, базах даних, електронній комерції, іграх з централізованим сервером для багатокористувацьких режимів та інших сценаріях, де потрібна надійна обробка та зберігання даних. Також вона є основою для більшості сервісів інтернету речей, де сервер обробляє дані з множини пристроїв.

Також існують гібридні мережеві архітектури, тобто інтегрують клієнт-серверні та однорангові (P2P) підходи, покращуючи масштабованість, гнучкість та відмовостійкість систем. Вони ефективні у сценаріях з високими вимогами до обробки даних, безпеки та надійності, знаходячи застосування в хмарних обчисленнях, IoT, обробці великих даних та медіа платформах.

Розробка та управління мережевими архітектурами, особливо гібридними, стикаються з технічними викликами, що вимагають глибокого розуміння технологій і застосування передових практик. Виклики безпеки в мережах, зокрема вразливість до атак типу Man-in-the-Middle (MitM) і Distributed Denial of Service (DDoS), потребують використання криптографічних протоколів як TLS для шифрування, а також аутентифікації та авторизації для контролю доступу. Для виявлення та реагування на підозрілу активність важливі системи IDS/IPS.

Масштабування вимагає технологій контейнеризації та оркестрації, як Docker і Kubernetes, для ефективного розгортання та управління мікросервісами, а алгоритми балансування навантаження забезпечують рівномірний розподіл запитів.

Управління даними в розподілених системах включає використання розподілених баз даних і систем управління версіями для забезпечення високої доступності та відмовостійкості. Протоколи консенсусу, такі як Raft або Paxos, забезпечують консистентність даних.

Зниження затримок у мережі досягається через використання edge computing для розміщення ресурсів ближче до кінцевих користувачів та оптимізації мережевих протоколів, наприклад QUIC, який зменшує кількість RTT.

Ефективна робота з великими обсягами даних вимагає впровадження технологій Big Data, як-от Hadoop для обробки датасетів та Elasticsearch для швидкого пошуку, а також використання спеціалізованих СУБД, таких як MongoDB або Apache HBase.

У підсумку, вибір між моделями залежить від специфіки гри, з важливістю інтеграції передових технічних рішень для створення захоплюючого ігрового досвіду. Клієнт-серверні системи пропонують стабільне управління ресурсами, але мають вразливості у масштабуванні та точки єдиної відмови. P2P моделі покращують масштабованість і зменшують залежність від центральних серверів, але вимагають додаткових зусиль для управління безпекою та консистентністю даних.

Список використаних джерел: 1. Intro to multiplayer network and server models. unity. URL: https://unity.com/how-to/intro-to-network-server-models (дата звернення: 20.02.2024). 2. Difference between Client-Server and Peer-to-Peer Network. GeeksforGeeks. URL: https://www.geeksforgeeks.org/difference-between-client-server-and-peer-to-peer-network (дата звернення: 20.02.2024). 3. Glazer J., Madhav S. Multiplayer game programming: Architecting networked games. – Addison-Wesley Professional, 2015. 4. Поповська К. О. Методи оптимізації процесу фрагментації контенту в пірингових файлообмінних мережах : автореф. дис. на здобуття наук. ступеня канд. техн. наук / Поповська Катерина Олегівна – Харків, 2017. – 28 с.