**УДК 004.528**

**О. П. Трусілов, М. В. Олексів**

Національний університет “Львівська політехніка”,

кафедра електронних обчислювальних машин

 **СИСТЕМА ВІДЕОЗВЯЗКУ НА ОСНОВІ КЛІЄНТ СЕРВЕРНОЇ АРХІТЕКТУРИ**

*© Трусілов О.П., Олексів М.В., 2019*

**Розглянуто проблему розробки веб додатку для відеозв’язку на основі клієнт серверної архітектури. В результаті аналізу існуючих аналогів розроблено додаток корпоративного використання для відеоконференцій.**

**Ключові слова: відеозв’язок , клієнт сервер, відеоконференції.**

**SYSTEM OF VIDEO STREAMING BASED ON CLIENT SERVER ARCHITECTURE**

*© Trusilov O.P. Oleksiv M.V.*

*Intron ltd; Lviv Polytechnic National University, Department of Computer Engineering*

**The problem of developing a web application for video communication based on server architecture clients is considered. As a result of the analysis of existing analogues, a corporate application for videoconferencing was developed.**

**Keywords: video, client server, video conferencing.**

**Вступ**

Відеозвязок в 21 столітті зайняв невід'ємну частину в спілкуванні між людьми. Через мережу Інтернет люди мають змогу спілкуватися друг з другом майже з будь якої точки планети, за допомогою спеціальних програм та додатків.

Метою статті є дослідження особливостей вибору архітектури та засобів розробки веб-додатку для відео та аудіо зв’язку.

 **Постановка задачі**

 Пропонується вирішити проблему кросплатформенності для додатків, які надають функціонал відео та аудіо зв'язку, за допомогою смотрення веб-додатку, який надає ці самі функції, але використовується за допомогою браузерів компьютера, що надає перевагу перед десктопними програмами в плані кросплатформенності програми, адже не завжди є змога встановити спеціальну програму на операційну систему, яку ви використовуєте.

**Вибір архітектури**

Для створення було обрано клієнт серверну архітектуру, що надає змогу використати головний плюс мого додатку, як кросплатформеність. Адже вся робота алгоритмів буде виконуватись на стороні сервера, а інтерфейсом може виступати будь який пристрій з будь якою операційною системою.



Рисунок 1. Модель клієнт-сервер

На рисунку 1 можемо бачити, що клієнтами виступають клієнтські пристрої, а мережею може виступати Інтернет.

 Модель клієнт-серверної взаємодії визначається розподілом обов’язків між клієнтом та сервером. Можна відокремити три рівні операцій:

* рівень представлення даних, який відповідає за представлення даних користувачу і введення від нього керуючих команд;
* прикладний рівень, який реалізує основну логіку застосунку і на якому здійснюється необхідна обробка інформації;
* рівень управління даними, який забезпечує зберігання даних та доступ до них.

Основна ідея цієї архітектури полягає в поділі мережевого додатку на кілька компонентів, кожен з яких реалізує специфічний набір сервісів. Компоненти такого додатку можуть виконуватися на різних комп’ютерах, виконуючи серверні і клієнтські функції. Це дозволяє підвищити надійність, безпеку і продуктивність мережевих додатків і мережі в цілому.

В клієнт серверній архітектурі велике значення відіграє функція, або як ще називають “роль” сервера.

В залежності від ролі, розрізняють такі сервери:

* Веб-сервер. Приймає HTTP-запити від клієнтів, зазвичай веб-браузерів, видає їм HTTP-відповіді. Є основою мережі Інтернет.
* Сервер застосунків або додатків. Виконує серверне програмне забезпечення, яке знаходиться на ньому.
* Сервер баз даних. Сервери баз даних використовуються для обробки запитів користувачів на мові SQL або NoSQL.
* Файловий сервер. Використовується для зберігання інформації у вигляді файлів.
* Поштовий сервер. Використовується для обслуговування пошти користувачів.

Для реалізації серверної частини проекту використовується такий набір технологій:

* мова програмування: Python;
* фреймворк для веб-розробки на Python: Django 1.11;
* СКБД: PostgreSQL.;

Для розробки проекту було обрано одну з найпопулярніших мов програмування - Python. Це інтерпретована мова програмування, високого рівня з строгою динамічною типізацією. Перевагою Python над іншими мовами програмування є швидка розробка, велика кількість бібліотек та фреймворків, що полегшує процес розробки, “чистий” синтаксис, що дозволяє швидко орієнтуватися в коді програми. Стандартний дистрибутив включає в себе велику кількість корисних модулів. Python широко використовується в різних сферах програмування, від веб-розробки до аналізу даних та розробки ігр. Може використовуватись як основна мова програмування, так і для створення розширень та додатків.

 Підтримує такі парадигми програмування, як:

* процедурне програмування;
* функціональне програмування;
* аспектно-орієнтоване програмування;
* обєктно-орієнтоване програмування.

Python запозичив багато рис з таких мов програмування, як C, C++, List, Fortran, Java, ACB тощо.

 Він портований майже під всі відомі платформи — від мобільних пристроїв до мейнфреймів. Також працює майже на всіх операційних системах, а в ОС Ubuntu входить в стартовий дистрибутив. Використовується в таких компаніях, як Google, YouTube, Intel, Cisco, HP, IBM, NASA тощо.

 На даний момент є дві повноцінні гілки Python — 2.х та 3.х. Останньою версією гілки 2.х є версія 2.7.15, а гілки 3.х є версія 3.6.5. Між ними немає повної сумусності. Зараз активно розвивається гілка 3.х, а 2.х просто підтримують. Не всі розробники поспішають перейти на новішу гілку, адже дуже багато проектів написано на на старішій версіі і іх треба підтримувати.

 Також слід описати фреймворк для Python — Dajngo.

 Django — високорівневий Python-фреймворк для розробки веб-систем. Однією з причин обрати Django було те, що проект будується за допомогою декількох частин, або модулів. Це дозволяє легко розширювати проект і надавати нового функціоналу для системи.

 Також до позитивних моментів в роботі з Django можна віднести його ORM. ORM — технологія програмування, яка зв`язує базу даних з концепціями ООП. Об`єкти БД в Django іменуються “моделями”. ORM надає змогу працювати з базою даних на більш високоріневому доступу до даних, що дає змогу не писати SQL-запити, але це не обов'язково.

**Висновки**

На основі досліджень було обрано архітектуру проекту, а саме клієнт-серверну, та обрано засоби для реалізації серверної частини додатку, поставлено задачу для розробки, та описано доцільність створення такого проекту.

*1.Мова програмування Python [Електронний ресурс]..Режим доступу:*[*https://uk.wikipedia.org/wiki/Python*](https://uk.wikipedia.org/wiki/Python)*. 2. Клієнт серверна архітектура [Електронний ресурс]. Режим доступу:*[*https://uk.wikipedia.org/wiki/Клієнт-серверна\_архітектура*](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0)*. 3. Іван Гудфелов, Йоша Бенгіо, Арон Коурвілле. «Машинне навчання» MIT Press, 2016. Режим доступу:* [*http://www.deeplearningbook.org*](http://www.deeplearningbook.org/)*. 4. Ліла В.Б. Алгоритм та программна реализація адаптивного метода навчання штучних нейронних мереж// Инженерний вестн. Дона, 2012. 5. Девід Е. Голберг «Дзен та мистецтво генетичного алгоритму». 3rd International Conference on Genetic Algorithms, pp. 80-85.*