Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

ЛАБОРАТОРНА РОБОТА №3

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Основи проектування розгортання»

Виконав:
студент групи ІА-34
Сухоручкін Гліб

Перевірив:
Асистент кафедри ІСТ
Мягкий М.Ю.

Київ – 2025

Зміст:

1. Вступ
2. Теоретичні відомості
3. Хід роботи
4. Діаграма розгортання FTP-серверу
5. Діаграма компонентів FTP-серверу
6. Діаграми послідовностей виористання FTP-серверу
7. Відповіді на питання до лабораторної роботи
8. Висновки

Вступ

Метою цієї роботи є навчитися проєктувати діаграми розгортання та компонентів для системи що проєктується, а також розробляти діаграми взаємодії, а саме діаграми послідовностей, на основі сценаріїв зроблених в попередній лабораторній роботі.

Проєкторваною системою цієї лабороторної роботи є FTP-server (використовуючи state, builder, memento, template method, visitor, client-server).

FTP-сервер повинен вміти коректно обробляти і відправляти відповіді по протоколу FTP, з можливістю створення користувачів (з паролями) і доступних їм папок, розподілу прав за стандартною схемою (rwe), ведення статистики з'єднань, обмеження максимальної кількості підключень і максимальної швидкості поширення глобально і окремо для кожного облікового запису.

Теоретичні відомості

Діаграми розгортання представляють фізичне розташування системи, показуючи, на якому фізичному обладнанні запускається та чи інша складова програмного забезпечення.

Головними елементами діаграми є вузли, пов'язані інформаційними шляхами. Вузол (node) – це те, що може містити програмне забезпечення. Вузли бувають двох типів. Пристрій (device) – це фізичне обладнання: комп'ютер або пристрій, пов'язаний із системою. Середовище виконання (execution environment) – це програмне забезпечення, яке саме може включати інше програмне забезпечення, наприклад операційну систему або процес-контейнер (наприклад, вебсервер).

Між вузлами можуть стояти зв'язки, які зазвичай зображують у вигляді прямої лінії. Як і на інших діаграмах, у зв'язків можуть бути атрибути множинності (для показання, наприклад, підключення 2х і більше клієнтів до одного сервера) і назва. У назві, як правило, міститься спосіб зв'язку між двома вузлами – це може бути назва протоколу (HTTP, IPC) або технологія, що використовується для забезпечення взаємодії вузлів (.NET Remoting, WCF).

Вузли можуть містити артефакти (artifacts), які є фізичним уособленням програмного забезпечення; зазвичай це файли. Такими файлами можуть бути виконувані файли (такі як файли .exe, двійкові файли, файли DLL, файли JAR, збірки або сценарії) або файли даних, конфігураційні файли, HTML-документи тощо. Перелік артефактів усередині вузла вказує на те, що на даному вузлі артефакт розгортається в систему, що запускається.

Артефакти можна зображати у вигляді прямокутників класів або перераховувати їхні імена всередині вузла. Якщо ви показуєте ці елементи у вигляді прямокутників класів, то можете додати значок документа або ключове слово «artifact». Можна супроводжувати вузли або артефакти значеннями у вигляді міток, щоб вказати різну цікаву інформацію про вузол, наприклад постачальника, операційну систему, місце розташування – загалом, усе, що спаде вам на думку.

Часто у вас буде безліч фізичних вузлів для розв'язання однієї й тієї самої логічної задачі. Можна відобразити цей факт, намалювавши безліч прямокутників вузлів або поставивши число у вигляді значення-мітки. Артефакти часто є реалізацією компонентів. Це можна показати, задавши значення-мітки всередині прямокутників артефактів. Основні види артефактів:

- вихідні файли;
- виконувані файли;
- сценарії;
- таблиці баз даних;
- документи;
- результати процесу розробки, UML-моделі.

Можна також деталізувати артефакти, що входять до вузла; наприклад, додатково всередині файлу що розгортується вказати, які туди входять компоненти або класи. Така деталізація, як правило, не має сенсу на діаграмах розгортання, оскільки може

зміщувати фокус уваги від моделі розгортання програмного забезпечення до його внутрішнього устрою, проте іноді може бути корисною. При цьому, можливо встановлювати зв'язки між компонентами/класами в межах різних вузлів.

Діаграми розгортань розрізняють двох видів: описові та екземплярні. На діаграмах описової форми вказуються вузли, артефакти і зв'язки між вузлами без вказівки конкретного обладнання або програмного забезпечення, необхідного для розгортання. Такий вид діаграм корисний на ранніх етапах розроблення для розуміння, які взагалі фізичні пристрої необхідні для функціонування системи або для опису процесу розгортання в загальному ключі.

Діаграми екземплярної форми несуть у собі екземпляри обладнання, артефактів і зв'язків між ними. Під екземплярами розуміють конкретні елементи – ПК із відповідним набором характеристик і встановленим ПЗ; цілком може бути, у межах однієї організації це може бути якийсь конкретний вузол (наприклад, ПК тестувальника Василя). Діаграми екземплярної форми розробляють на завершальних стадіях розроблення ПЗ – коли вже відомі та сформульовані вимоги до програмного комплексу, обладнання закуплено і все готово до розгортання. Діаграми такої форми являють собою скоріше план розгортання в графічному вигляді, ніж модель розгортання.

Хід роботи

Діаграма розгортання FTP-серверу

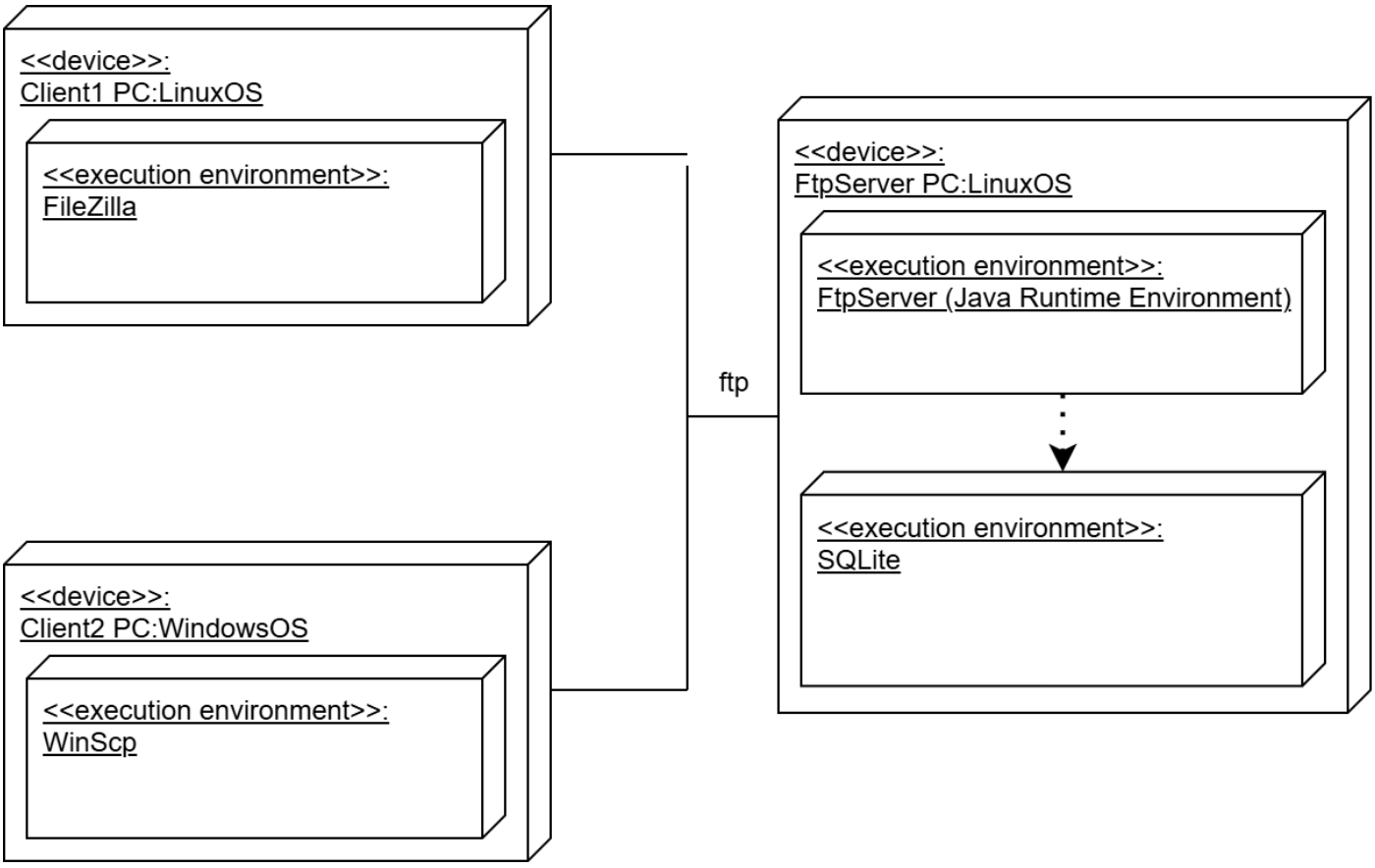

Рис. 1 – Діаграма розгортання FTP-серверу

Пояснення діаграми розгортання FTP-серверу

Клієнти:

Client1 PC: LinuxOS з середовищем виконання FileZilla.

Client2 PC: WindowsOS з середовищем виконання WinSCP.

Сервер: FtpServer PC: LinuxOS, усередині: середовище FtpServer (Java Runtime Environment) – Java-додаток FTP-сервера; окреме середовище SQLite – вбудована БД.

Зв'язки: клієнтські вузли з'єднуються з сервером по каналу ftp; усередині сервера застосунок FtpServer має залежність від SQLite (зберігання даних облікових записів/прав/статистики).

Діаграма компонентів FTP-серверу

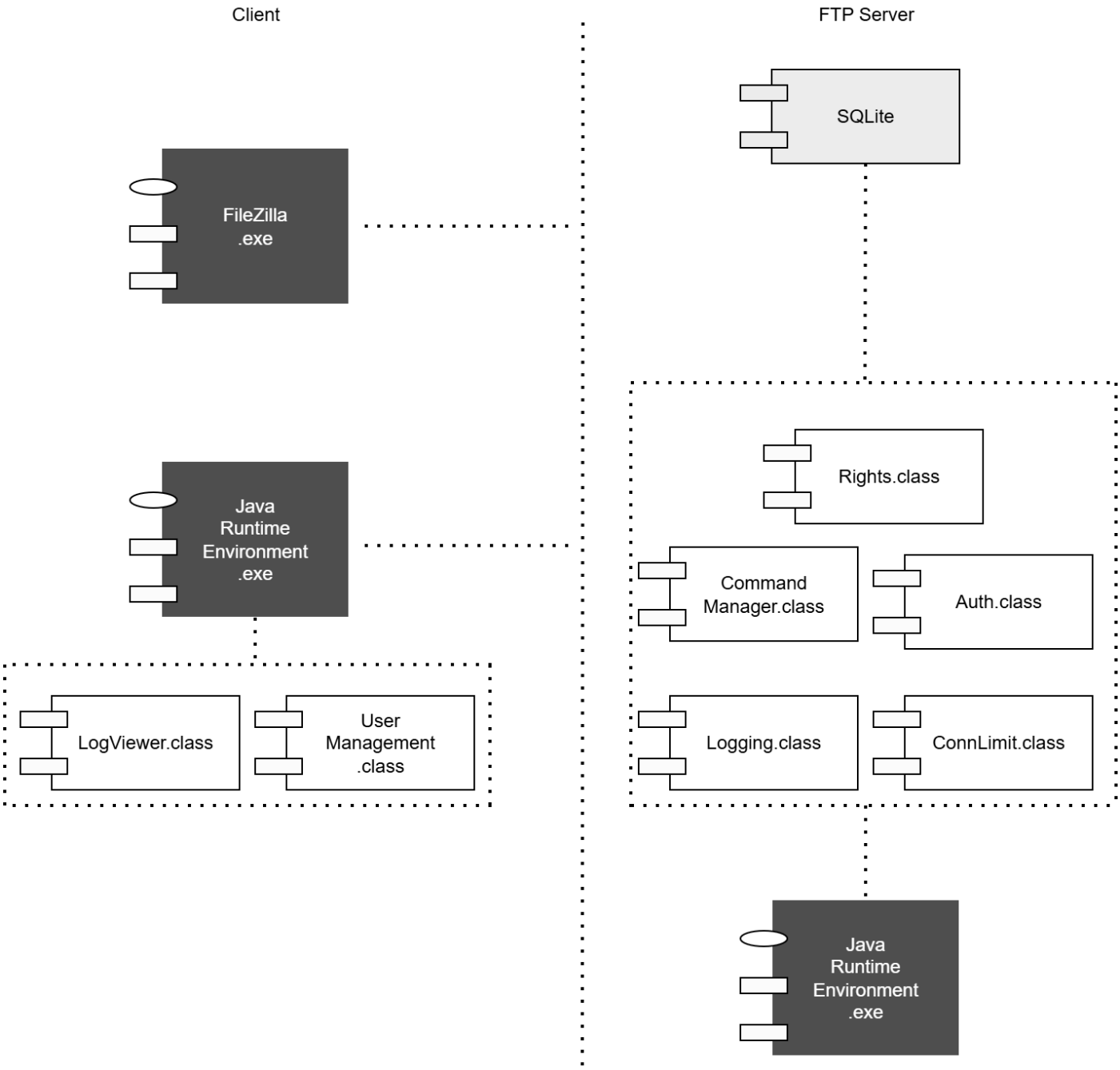

Рис. 2 - Діаграма компонентів FTP-серверу

Пояснення діаграми компонентів FTP-серверу

Клієнт

FileZilla.exe – FTP-клієнт, що під’єднується до сервера.

Java Runtime Environment.exe – середовище виконання для клієнтських утиліт.

LogViewer.class, UserManagement.class – клієнтські модулі для перегляду логів і керування користувачами; працюють у JRE та взаємодіють із сервером (пунктиром показані зв’язки).

Сервер

Java Runtime Environment.exe — запускає серверні класи.

CommandManager.class — диспетчер/маршрутизатор FTP-команд.

Auth.class — автентифікація користувачів.

Rights.class — перевірка прав доступу (r/w/e).

Logging.class — збір і збереження журналів.

ConnLimit.class — обмеження/контроль кількості з’єднань.

SQLite — вбудована БД для облікових записів, прав, статистики; використовується серверними компонентами.

Діаграми послідовностей використання FTP-серверу

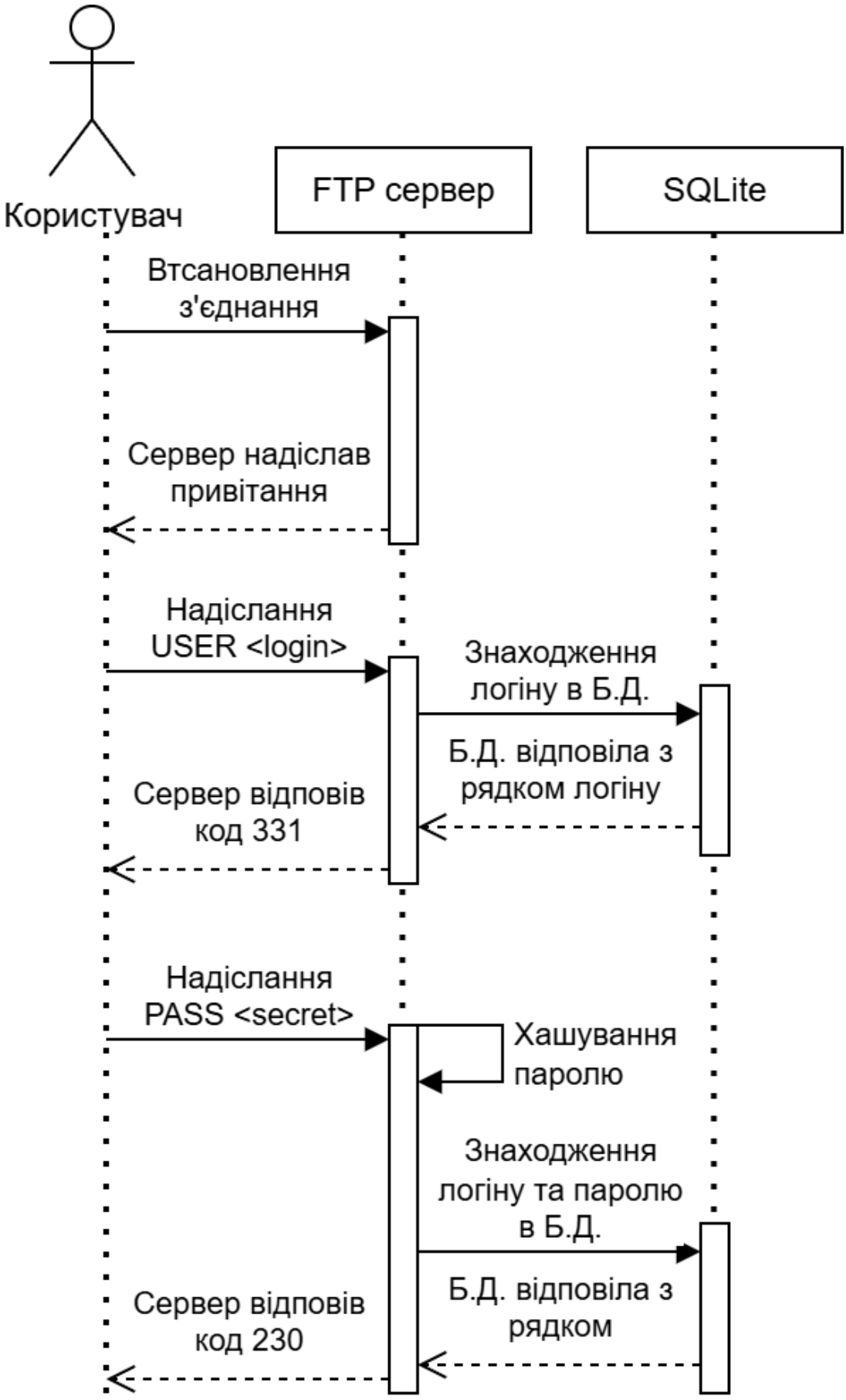

Рис 3. Діаграма послідовності використання FTP-серверу (Авторизація користувача)

Пояснення діаграми послідовності використання FTP-серверу

Встановлення з’єднання. Користувач під’єднується до FTP-сервера, сервер

активується й надсилає привітання.

USER <login>. Клієнт відправляє логін; сервер звертається до SQLite, знаходить запис із цим логіном і відповідає кодом 331 (логін прийнято, потрібен пароль).

PASS <secret>. Сервер хешує отриманий пароль, виконує пошук пари логін+хеш у БД; після успішного збігу БД повертає рядок, сервер відповідає кодом 230 (користувача авторизовано).

На діаграмі вертикальні «життєві лінії» та прямокутники активності показують, коли компонент обробляє запит; стрілки – обмін повідомленнями між Користувачем, FTP-сервером і SQLite.

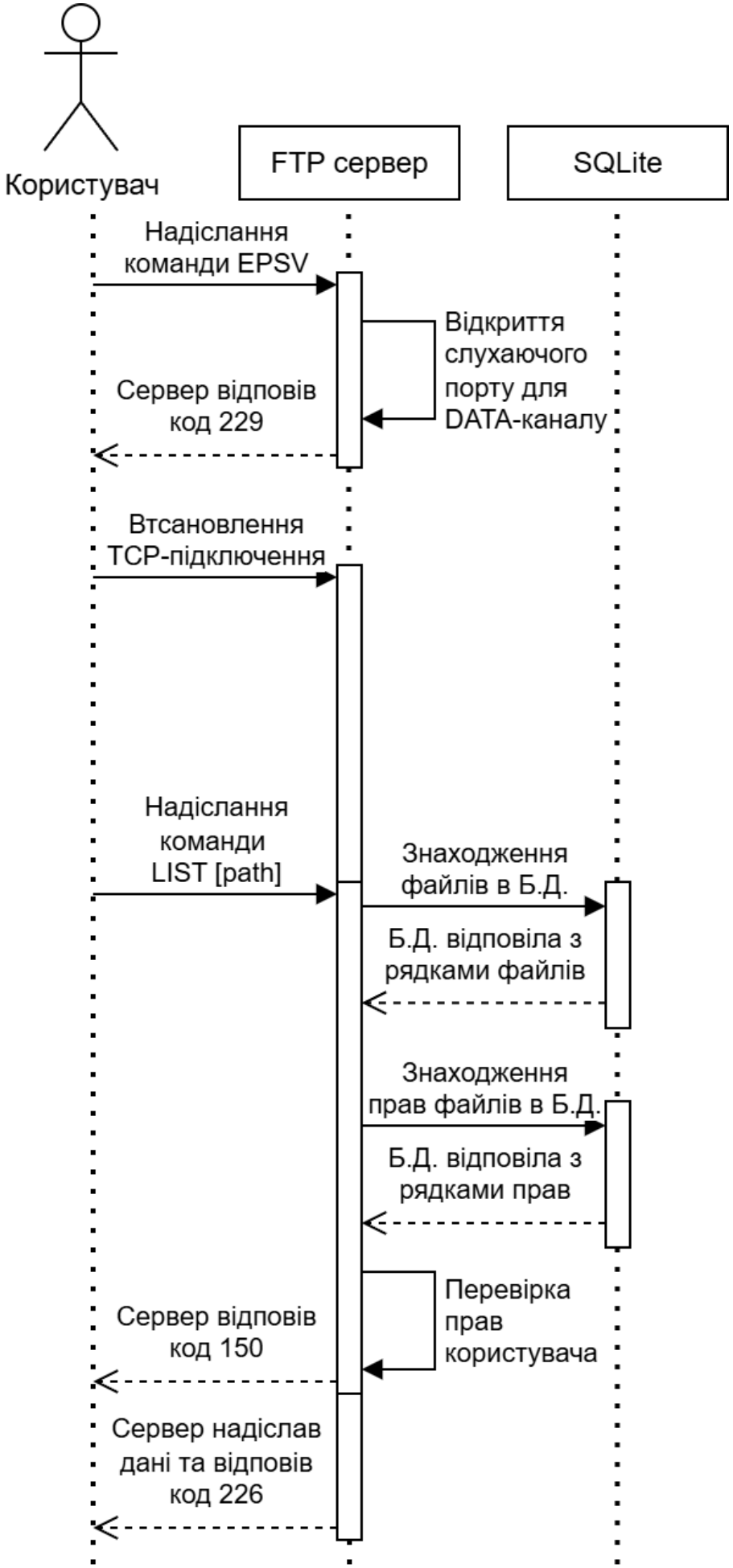

Рис 4. Діаграма послідовності використання FTP-серверу (Перегляд вмісту каталогу)

Пояснення діаграми послідовності використання FTP-серверу

EPSV: клієнт надсилає команду; сервер відкриває пасивний DATA-порт і відповідає кодом 229 (повідомляє порт).

Клієнт встановлює TCP-з’єднання з цим DATA-портом.

LIST [path]: клієнт просить список; сервер звертається до SQLite – спершу за файлами каталогу, потім за правами доступу для них.

Сервер перевіряє права користувача на кожен елемент і формує відповідь.

Сервер відповідає 150 (початок передачі), надсилає список по DATA-каналу і після завершення повертає 226 (передачу успішно завершено).

Відповіді на питання до лабораторної роботи

1. Що собою становить діаграма розгортання?

Діаграми розгортання показують фізичне розташування системи — на якому обладнанні запускаються складові ПЗ.

2. Які бувають види вузлів на діаграмі розгортання?

Два види: пристрій (device) – фізичне обладнання; середовище виконання (execution environment) – ПЗ, що може містити інше ПЗ (ОС, процес-контейнер тощо).

3. Які бувають зв’язки на діаграмі розгортання?

Зв’язки між вузлами зображують прямими лініями; вони можуть мати множинність і назву (наприклад, протокол HTTP, IPC або технології на кшталт .NET Remoting, WCF).

4. Які елементи присутні на діаграмі компонентів?

Показують компоненти (часто як фізичні файли: .exe, .dll, вихідні коди, HTML, БД і таблиці) та залежності між ними; компоненти можуть групуватися (серверні, клієнтські, middleware).

5. Що становлять собою зв’язки на діаграмі компонентів?

Залежності відображають, що класи одного компонента використовують класи іншого компонента.

6. Які бувають види діаграм взаємодії?

У документі виділено: діаграми кооперації (collaboration) та діаграми послідовностей (sequence).

7. Для чого призначена діаграма послідовностей?

Для моделювання взаємодії між об’єктами у часі, показуючи порядок і логіку виконання операцій.

8. Які ключові елементи можуть бути на діаграмі послідовностей?

Актори; об’єкти/класи з лініями життя; повідомлення (синхронні/асинхронні, повернення); активності; контрольні структури (alt, loop).

9. Як діаграми послідовностей пов’язані з діаграмами варіантів використання?

Діаграми послідовностей деталізують сценарії use-case: діаграма варіантів використання є вихідною концептуальною моделлю, а sequence-діаграми розкривають конкретний обмін повідомленнями для цих сценаріїв.

10. Як діаграми послідовностей пов’язані з діаграмами класів?

Sequence-діаграми показують динамічну взаємодію об’єктів, що є екземплярами класів зі статичної моделі (діаграми класів); тим самим вони використовують/уточнюють операції цих класів.

Висновки

У цій роботі було спроєктовано та описано ключові UML-артефакти для FTP-сервера: діаграму розгортання, діаграму компонентів і діаграми послідовностей використання для сценаріїв авторизації користувача та перегляду вмісту каталогу.