Міністерство освіти і науки України
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: **«Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.»**
Flex Automatical tool

Виконав:
Студент групи ІА-22
Сидорін Д.О.

Перевірив:
Мягкий М. Ю.

Київ-2024

Зміст

Тема:
Діаграма розгортання. Діаграма компонентів. Діаграма взаємодій та послідовностей.
Мета:
Засвоїти основні типи діаграм, які використовуються для моделювання програмних систем, зокрема діаграми розгортання, компонентів, взаємодій і послідовностей. Навчитися будувати й аналізувати такі діаграми, а також застосовувати їх для опису архітектури та процесів у програмних системах.

Завдання:

Візуальний додаток для складання "карт пам'яті" з можливістю роботи з декількома картами (у вкладках), автоматичного промальовування ліній, додавання вкладених файлів, картинок, відеофайлів (попередній перегляд); можливість додавання значків категорій / терміновості, обведення областей карти (поділ пунктирною лінією).

Хід роботи

1. Діаграма розгортання

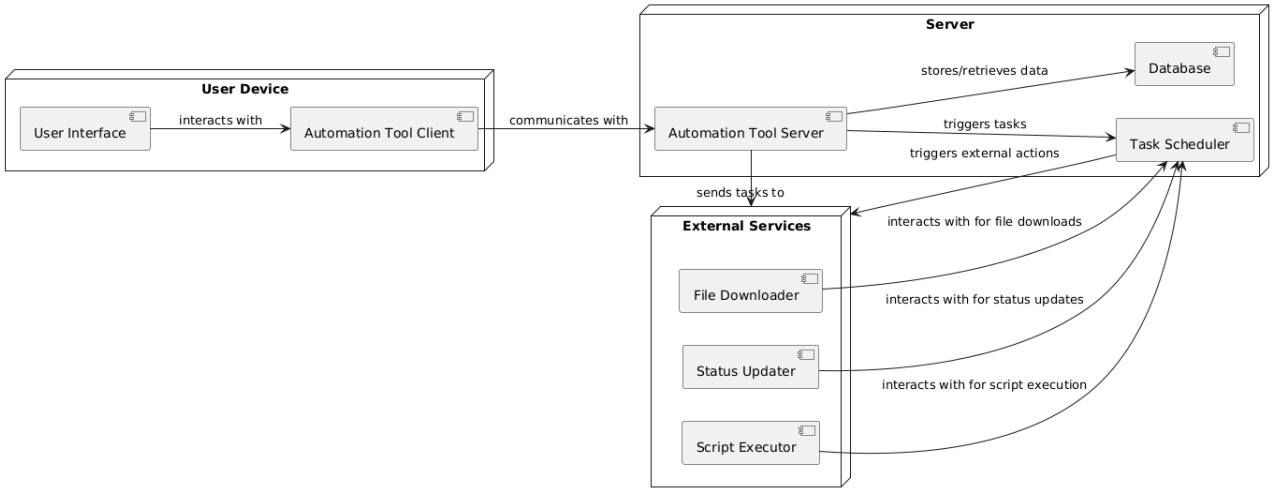

Рис. 1 — Діаграма розгортання

Ця діаграма розгортання відображає архітектуру системи автоматизації, яка складається з кількох компонентів, що взаємодіють між собою. Всі елементи системи (компоненти та вузли) мають чітко визначену роль і зв'язки, що дозволяє забезпечити коректне виконання автоматизованих завдань.

1. User Device (Користувацький пристрій):

- **User Interface (UI)**:

    - Це частина системи, з якою безпосередньо взаємодіє користувач. Користувач через цей інтерфейс може налаштовувати параметри автоматизації, запускати завдання та взаємодіяти з іншими функціональними частинами системи.
- **Automation Tool Client (ATC)**:
    - Клієнтська частина інструменту автоматизації, яка приймає введення користувача через інтерфейс і передає їх на сервер для обробки. ATC відповідає за передачу запитів до серверної частини.

2. Server (Сервер):

- **Automation Tool Server (ATS)**:
    - Основна серверна частина системи, що обробляє запити від клієнта та виконує основні функції автоматизації. ATS реалізує логіку створення правил автоматизації, запису макросів, планування завдань і обробки даних.
- **Task Scheduler (TS)**:
    - Компонент для планування завдань, який дозволяє виконувати автоматичні операції у певний час або за заданими умовами. Task Scheduler взаємодіє з іншими компонентами, такими як File Downloader, Status Updater і Script Executor, для виконання завдань.
- **Database (DB)**:
    - База даних для зберігання всієї інформації, необхідної для роботи системи: дані про інструменти автоматизації, правила, макроси, завдання та їх виконання. Всі дані зберігаються в базі для подальшого використання та маніпуляцій.

3. External Services (Зовнішні сервіси):

- **File Downloader (FD)**:
    - Зовнішній сервіс для автоматичного завантаження файлів. Цей сервіс активується через **Task Scheduler**, який передає йому завдання для завантаження файлів за певними правилами або умовами.
- **Status Updater (SU)**:
    - Зовнішній сервіс для оновлення статусів в комунікаторах або інших системах. Наприклад, може використовуватись для зміни статусу користувача в Skype або іншому чаті.
- **Script Executor (SE)**:
    - Зовнішній сервіс для виконання скриптів. Task Scheduler передає йому завдання для виконання скриптів, що можуть бути використані для різних автоматичних операцій.

Взаємозв'язки між компонентами:

1. **User Interface** взаємодіє з **Automation Tool Client**, передаючи користувацькі запити на виконання автоматизації.

2. **Automation Tool Client** передає запити на обробку до **Automation Tool Server**, де здійснюється основна логіка автоматизації.
3. **Automation Tool Server** в залежності від запиту звертається до **Task Scheduler** для створення і планування завдань.
4. **Task Scheduler** передає завдання до відповідних зовнішніх сервісів (File Downloader, Status Updater, Script Executor), щоб виконати необхідні операції.
5. **Automation Tool Server** також звертається до **Database** для збереження, оновлення або отримання даних про виконання завдань та налаштування автоматизації.

Загальний процес:

- Користувач через **User Interface** ініціює запит на автоматизацію.
- **Automation Tool Client** передає запит до **Automation Tool Server**, де здійснюється обробка.
- Якщо потрібно виконати завдання, **Automation Tool Server** звертається до **Task Scheduler**, який запускає зовнішні сервіси для завантаження файлів, оновлення статусів або виконання скриптів.
- Протягом цього процесу всі дані, які необхідно зберігати або отримувати, обробляються через **Database**.

2. Діаграма компонентів

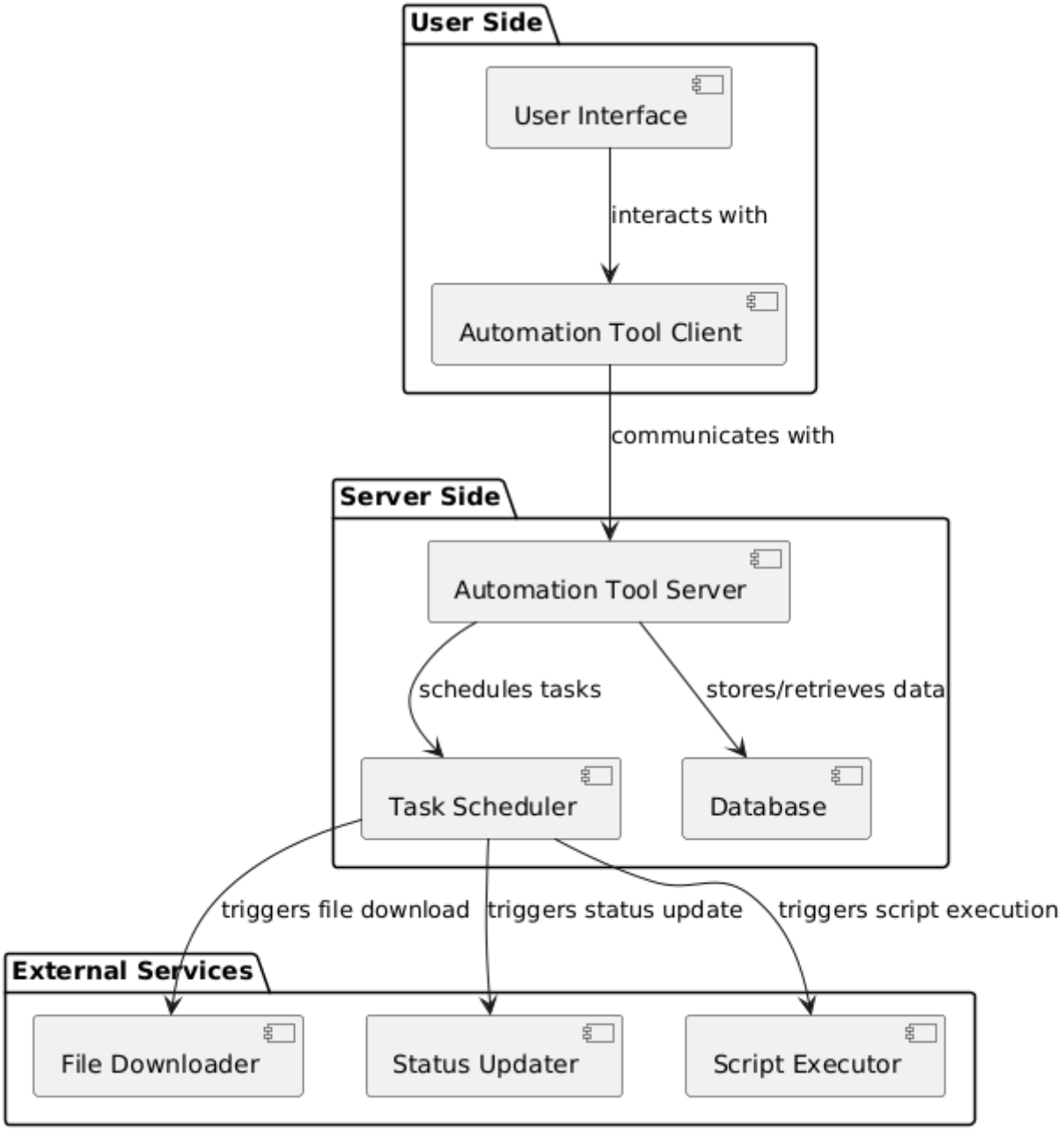

Рис. 2 — Діаграма компонентів

Ця діаграма компонентів відображає основні компоненти системи автоматизації та їх взаємозв'язки, а також визначає, як кожен компонент взаємодіє з іншими для досягнення цілей автоматизації. Система складається з трьох основних частин: користувацької частини, серверної частини та зовнішніх сервісів.

1. User Side (Користувацька частина):

- **User Interface**:

    - Це інтерфейс, з яким взаємодіє кінцевий користувач. Через нього користувач може ініціювати автоматизацію різних завдань (наприклад, завантаження нових фільмів, оновлення статусів у комунікаторах або виконання скриптів).
    - **User Interface** передає запити на виконання завдань до **Automation Tool Client**.
- **Automation Tool Client**:
    - Клієнтська частина системи, яка відповідає за прийом запитів від користувача через **User Interface** та їх передачу до **Automation Tool Server** для обробки.
    - Це компонент, який забезпечує зв'язок між користувачем і серверною частиною системи.

2. Server Side (Серверна частина):

- **Automation Tool Server**:
    - Основний компонент системи, який відповідає за обробку запитів та виконання основної логіки автоматизації.
    - Взаємодіє з **Task Scheduler** для планування завдань, а також з **Database** для збереження та отримання необхідних даних (правила, макроси, налаштування користувачів тощо).
- **Task Scheduler**:
    - Компонент для планування та виконання завдань. Цей компонент визначає, які завдання виконувати, коли і за яких умов.
    - Він отримує запити від **Automation Tool Server** для виконання завдань і передає їх до зовнішніх сервісів (наприклад, для завантаження файлів або оновлення статусів).
- **Database**:
    - Система для зберігання даних. Вся інформація про налаштування, правила автоматизації, макроси, користувачів та історію виконаних завдань зберігається в базі даних.
    - Взаємодіє з **Automation Tool Server** для збереження та отримання необхідних даних.

3. External Services (Зовнішні сервіси):

- **File Downloader**:
    - Зовнішній сервіс для завантаження файлів (наприклад, нових серій фільмів або книг), який активується через **Task Scheduler**.
    - **Task Scheduler** передає запит на завантаження файлів, і **File Downloader** виконує завдання за вказаним планом.
- **Status Updater**:
    - Зовнішній сервіс для оновлення статусів у комунікаційних платформах, таких як Skype або інші месенджери.

- Використовується для автоматичної зміни статусів користувачів (наприклад, "away" після тривалого періоду бездіяльності).
- **Task Scheduler** передає команду **Status Updater** для оновлення статусу.

- **Script Executor**:
  - Зовнішній сервіс для виконання скриптів або макросів.
  - **Task Scheduler** передає завдання **Script Executor**, щоб він виконав визначений скрипт для автоматизації певних операцій або команд.

Взаємозв'язки між компонентами:

1. **User Interface** та **Automation Tool Client**:
   - Користувач ініціює запити через **User Interface**, який передає їх **Automation Tool Client** для подальшої обробки.
2. **Automation Tool Client** та **Automation Tool Server**:
   - **Automation Tool Client** передає запити на сервер для обробки, де **Automation Tool Server** здійснює основну логіку автоматизації.
3. **Automation Tool Server** та **Task Scheduler**:
   - **Automation Tool Server** відправляє запити до **Task Scheduler**, щоб той ініціював виконання запланованих завдань.
4. **Automation Tool Server** та **Database**:
   - **Automation Tool Server** зберігає та отримує дані з **Database** (наприклад, для збереження правил, налаштувань користувачів або інформації про виконання завдань).
5. **Task Scheduler** та **External Services**:
   - **Task Scheduler** ініціює виконання завдань через зовнішні сервіси, такі як **File Downloader**, **Status Updater** та **Script Executor**, для виконання певних операцій.

3. Діаграма послідовностей

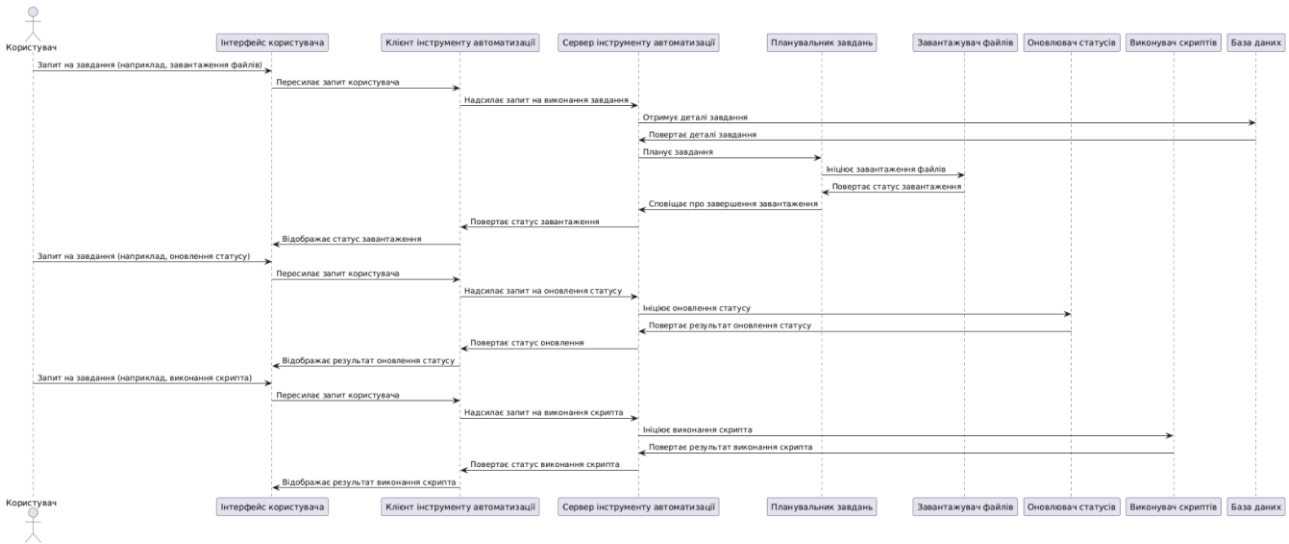

Рис. 3 — Діаграма послідовностей

Ця діаграма послідовностей описує процеси взаємодії користувача з системою автоматизації для різних завдань: завантаження файлів, оновлення статусу в месенджерах і виконання скриптів. Кожен процес проходить кілька етапів, де різні компоненти системи виконують свої функції.

1. Завантаження файлів:

1. **Користувач** ініціює запит на завантаження файлів через **Інтерфейс користувача**.
2. **Інтерфейс користувача** передає цей запит до **Клієнта інструменту автоматизації** (ATC).
3. **Клієнт інструменту автоматизації** надсилає запит на виконання завдання до **Сервера інструменту автоматизації** (ATS).
4. **Сервер інструменту автоматизації** звертається до **Бази даних** для отримання необхідної інформації про завдання (наприклад, налаштування для завантаження файлів).
5. **База даних** повертає необхідну інформацію до **Сервера інструменту автоматизації**.
6. **Сервер інструменту автоматизації** передає завдання на виконання до **Планувальника завдань** (TS).
7. **Планувальник завдань** звертається до **Завантажувача файлів** для завантаження необхідних файлів.
8. **Завантажувач файлів** виконує завантаження файлів і передає результат (наприклад, повідомлення про успішне завантаження або помилку) назад до **Планувальника завдань**.
9. **Планувальник завдань** інформує **Сервер інструменту автоматизації** про завершення завдання.
10. **Сервер інструменту автоматизації** передає статус виконання завдання до **Клієнта інструменту автоматизації**.
11. **Клієнт інструменту автоматизації** відображає результат виконання завдання через **Інтерфейс користувача**.

2. Оновлення статусу:

1. **Користувач** ініціює запит на оновлення свого статусу (наприклад, в Skype чи іншому месенджері) через **Інтерфейс користувача**.
2. **Інтерфейс користувача** передає запит до **Клієнта інструменту автоматизації**.
3. **Клієнт інструменту автоматизації** надсилає запит на виконання завдання до **Сервера інструменту автоматизації**.
4. **Сервер інструменту автоматизації** передає запит на оновлення статусу до **Оновлювача статусів** (SU).
5. **Оновлювач статусів** виконує оновлення статусу і повертає результат (наприклад, успішне оновлення) до **Сервера інструменту автоматизації**.
6. **Сервер інструменту автоматизації** передає результат виконання до **Клієнта інструменту автоматизації**.

7. **Клієнт інструменту автоматизації** відображає результат оновлення статусу на **Інтерфейсі користувача**.

3. Виконання скрипта:

1. **Користувач** ініціює запит на виконання скрипта через **Інтерфейс користувача**.
2. **Інтерфейс користувача** передає запит до **Клієнта інструменту автоматизації**.
3. **Клієнт інструменту автоматизації** надсилає запит на виконання завдання до **Сервера інструменту автоматизації**.
4. **Сервер інструменту автоматизації** передає запит на виконання скрипта до **Виконувача скриптів** (SE).
5. **Виконувач скриптів** виконує скрипт і повертає результат виконання (наприклад, успішне виконання або помилка) до **Сервера інструменту автоматизації**.
6. **Сервер інструменту автоматизації** передає результат виконання скрипта до **Клієнта інструменту автоматизації**.
7. **Клієнт інструменту автоматизації** відображає результат виконання скрипта на **Інтерфейсі користувача**.

Висновки:

У даній лабораторній роботі було розроблено систему автоматизації для виконання різноманітних завдань, таких як завантаження файлів, оновлення статусів у месенджерах та виконання скриптів, яка ілюструє основні процеси взаємодії користувача з додатком, діаграму розгортання, що описує фізичну архітектуру системи, та діаграму компонентів, яка демонструє модульну структуру додатку і взаємозв’язки між його частинами. Створення чіткої архітектури допомогло забезпечити ефективну взаємодію між компонентами та правильну реалізацію бізнес-логіки системи.