

COSC540 - Computer Networks and Network Security

Assessment – 6 Reflection Report



Bikash Neupane

220245756

bneupan2@myune.edu.au

Table of Contents

[Background 1](#_Toc165188838)

[Challenges faced across both assessments: 1](#_Toc165188839)

[Writing the Second Client - Easier or More Difficult? 1](#_Toc165188840)

[Compatibility Issues Using Different Languages: 2](#_Toc165188841)

**Reflection Report: Client Implementation Comparison**

# Background

In the context of Assessment 2 and Assessment 6, I implemented two client applications and one server application, each serving as a user interface to interact with a server through socket communication. In assessment 2, I implemented both server and client in Python whereas in assessment2, I implemented the client only in JAVA and the server remains the same. In this reflection, I'll compare the two implementations, discussing challenges faced, ease of development, and any compatibility issues encountered.

# Challenges faced across both assessments:

**Understanding Socket Communication:** Initially, understanding how sockets work and how to implement communication between a client and a server was a significant challenge for me in both assessments. Learning about socket programming and the client-server model was essential to catch the concepts required for implementation.

**Handling User Input and Commands:** Dealing with user input and processing commands posed a challenge in both assessments. Ensuring the client can interpret user commands correctly and send them to the server in the expected format required careful design and validation.

# Writing the Second Client - Easier or More Difficult?

Writing the client for Assessment 6 was both easier and more challenging as well in certain aspects.

**Easier: Familiarity with Socket Communication:** Since I had already implemented a client-server application in Assessment 2, I had a better understanding of how socket communication works. This familiarity made it easier to catch the concepts and implement the client in Java for Assessment 6.

**More Difficult: Managing Java's Strict Typing:** Java's strong typing system, while powerful, sometimes made tasks such as parsing and handling user input more complicated compared to Python. For example, in Python, I could easily split a string and process its parts without worrying about types, whereas in Java, I had to explicitly convert types and handle exceptions.

**Easier: Object-Oriented Approach:** Java's object-oriented nature allowed me to structure the code in a more organized manner. I could create methods for different functionalities, making the code more modular and easier to understand.

**More Difficult: Managing Resources:** Java requires explicit resource management, such as closing sockets and streams. While this ensures better resource handling, it also adds complexity to the code, especially when dealing with exceptions and try-catch blocks.

# Compatibility Issues Using Different Languages:

Overall, I didn't encounter significant compatibility issues when switching between Python and Java for client implementation.

**Socket Communication:** Both Python and Java provide robust support for socket programming, making it easy to establish communication between clients and servers.

**Language-Specific Features:** While Python and Java have differences in syntax and language features, these didn't pose compatibility issues as I was mainly dealing with basic socket operations and string processing, which are common to both languages.

**Development Environment:** Both Python and Java have highly enhanced development environments (IDEs) and libraries for network programming, ensuring a smooth development experience.

**Code Structure:** While the code structure differed between Python and Java due to their language-specific features, the overall logic and functionality remained consistent across both implementations.

In conclusion, while both client implementations had their challenges, writing the client for Assessment 6 in Java was easier due to my prior experience with socket communication. However, dealing with Java's strict typing and resource management added some complexity. Despite these differences, both clients effectively served their purpose of interacting with the server and demonstrated the versatility of socket programming across different languages.