Automatic Assessment of Tasks in the Course Programming Network Applications

Author: **Ing. Martin Krčma**, Supervisor: **Ing. Tomáš Dulík, Ph.D.**

Tomas Bata University in Zlin - Faculty of Applied Informatics

1. Motivation

The primary motivation for this thesis was to solve the problem associated with the **increasing number of students at the faculty** and to ease the workload of teachers in evaluating assignments. This work specifically focuses on the programming network applications course, where verifying the functionality of students' solutions often requires additional network resources (servers, clients, ...), making the assessment process time-consuming. Network applications **often have varied behaviors and communicate in different ways**, which poses a significant challenge in creating a universal solution that can accommodate all these testing needs.

Therefore, the goal was to develop a flexible and universal solution that would allow the testing of software assignments both **locally and within GitLab repositories**, without relying on external network resources. This will help students and, more importantly, teachers, as this solution will save them a significant amount of time, allowing them to focus on more important matters at the faculty. This solution is not limited to educational institutions, it can also be used in a wide range of software projects.

2. Conclusion

The result of this thesis is a **universal black-box testing tool** that effectively enables the testing of network applications. This tool, developed in Java and designed to be cross-platform, supports testing across a wide range of network applications using commonly employed communication protocols. It is easily **configurable via YAML**, and a straightforward configuration language has been created to define test scenarios using a basic set of keywords. Additionally, a **simple IDE** was developed to facilitate the creation and runtime testing of these test scenario configurations.

This thesis includes not only the the Network application testing tool **(NATT)** but also the IDE, tool documentation, example projects and configurations, a new set of assignments for the network programming course, GitLab repositories for each assignment and aditional tools for course.

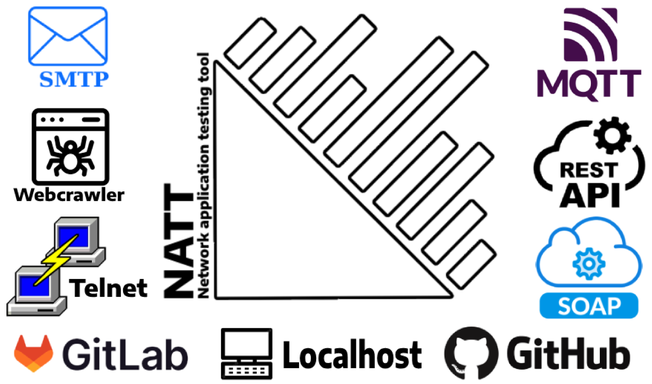

Figure 1. Network Application Testing Tool (NATT) - supported technologies

3. Main features of testing tool

This Black Box Testing Tool automates the testing and evaluation of software applications without needing internal implementation details. It supports a **wide range** of applications, operates **independently of external network resources** by creating virtual servers and clients, and offers flexible configuration for defining new test scenarios.

What does the tool allow you to test?

1. Testing simple **email** sending applications
2. Testing **clients** that use the telnet protocol
3. Testing **servers** that use the telnet protocol
4. Testing applications that use **REST API**
5. Testing **SOAP web services**
6. Testing **MQTT clients**
7. Testing **Web crawlers**
8. Testing the application through the **standard stream**

4. Principle of the testing tool

These two diagrams illustrate how application testing is conducted using the **NATT** black box testing tool. The diagrams show the process of **how the tool interacts** with different types of applications.

On the left side of each diagram, the testing tool communicates with the tested application through **virtually created modules (highlighted in blue)**. This communication is then evaluated to determine if the application meets the defined expectations.

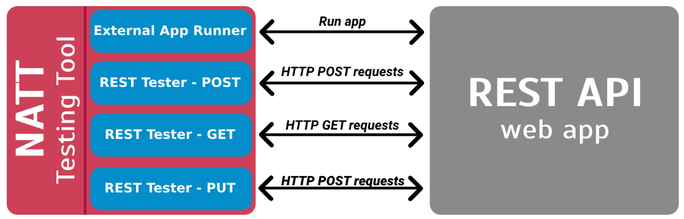

Figure 2. Testing of REST API application

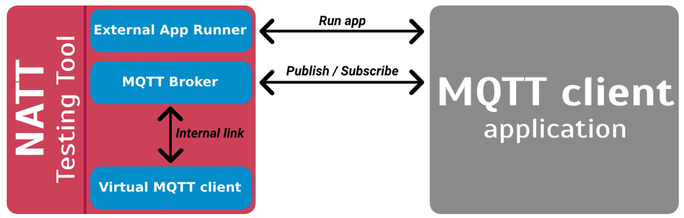

Figure 3. Testing of MQTT application

5. Test scenarios and configuration

The image below illustrates how test scenarios are defined for this tool. At the beginning is the Test Root node, which is the **entry point** for each test scenario configuration. This root node sets the overall context for the testing process and encompasses all test suites within it.

The diagram illustrates the **context of each test segment** and the duration for which variables or resources remain active. On the right side, a simplified test scenario configuration is shown, which NATT **uses to generate** an acyclic graph that is subsequently executed.

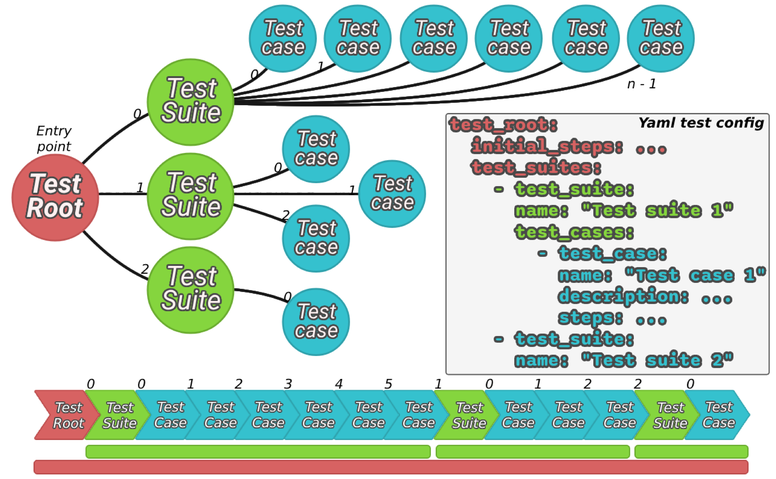

Figure 4. Testing structure and configuration

6. Usage of testing tool

1. **Educational Use:** Automates and standardizes student assignment evaluations in network programming courses.
2. **Professional Use:** Facilitates efficient testing and debugging of network applications in software development.
3. **Programming Competitions:** Provides automated testing and fair evaluation for multiple submissions in competitive programming events.

7. NATT IDE

The NATT IDE (Configuration Editor) offers a easy interface for **creating test configurations**. It includes an **auto-completion** feature for all keywords, automatically providing required parameters. Users can also **run tests** directly within the IDE to check their behavior.

Github: **https://github.com/0xMartin** NATT repository: **https://github.com/0xMartin/NetworkAppTestingTool** **martin.krcma1@gmail.com**