GALWAY-MAYO INSTITUTE OF TECHNOLOGY

Department of Computer Science & Applied Physics

B.Sc. Software Development – Distributed Systems (2017)
ASSIGNMENT DESCRIPTION & SCHEDULE

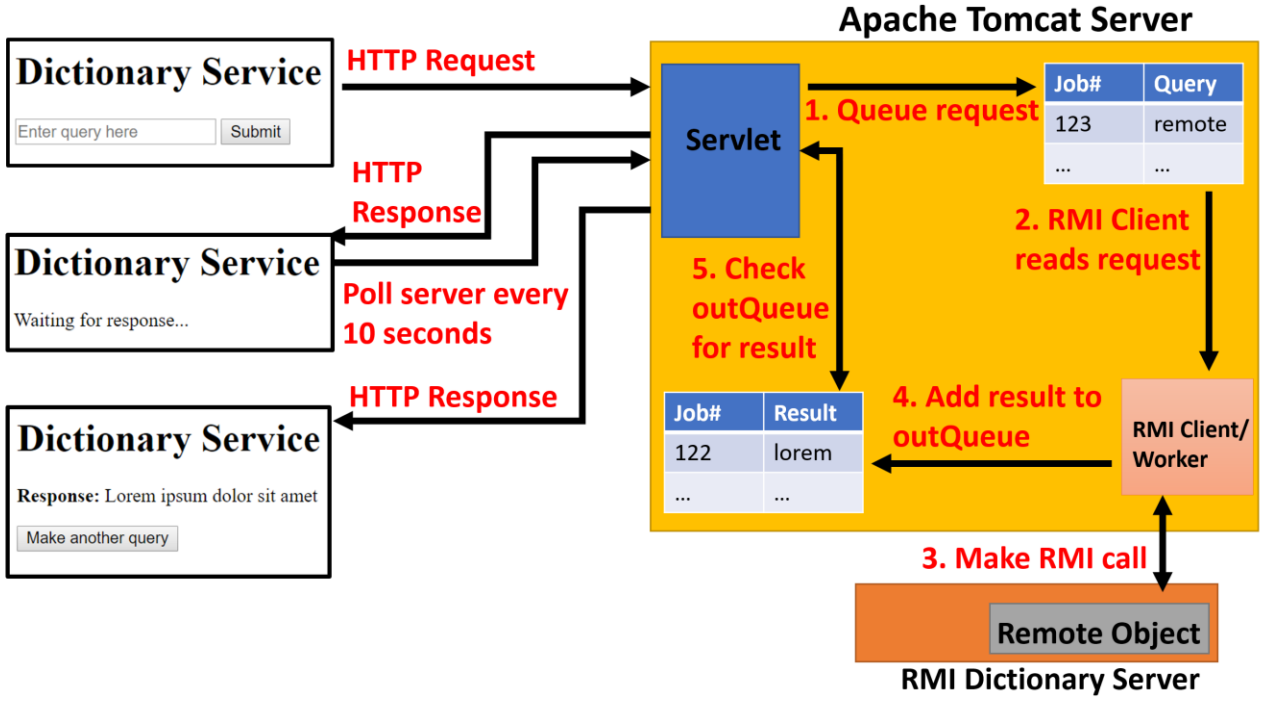
An Asynchronous RMI Dictionary Service

Note: This assignment will constitute 40% of the total marks for this module.

1. Overview

You are required to use the Servlet/JSP and Java RMI frameworks to develop a remote, *asynchronous* dictionary lookup service. A JSP page should provide users with the ability to specify a string which will be checked against the dictionary. The HTML form information should be dispatched to a servlet that adds the client request to an in-queue and then returns a job ID to the web client. The web client should poll the web server periodically (every 10 seconds) and query if the request has been processed. Client requests in the **inQueue** should be periodically removed and processed (every 10 seconds).

The processing of a client request will require a RMI method invocation to a remote object which implements an interface called **DictionaryService**. The remote object which implements **DictionaryService** should check if the string received exists in the dictionary, and return the dictionary definition of the string if it does exist in the dictionary, or "String not found" if it does not exist in the dictionary. Once the result of the dictionary lookup has been computed by the remote object, the returned response should be added to the **outQueue** on the Tomcat server and returned to the original web client when they next poll the server. The following diagram depicts the overall system architecture:



The point of this exercise is to give you some "hands-on" experience programming an asynchronous remote software service. Asynchronous communication is an important topic in distributed computing, as it provides a degree of scalability if the number of potential requests is unknown or may vary significantly.

2. Minimum Requirements

You are required to use the RMI framework to implement the dictionary lookup service. Your implementation should include the following features:

1. A web client request should be placed in a message queue to await processing. Each request should be allocated a job number. The job number should be added to an **inQueue** (a Map) along with the request string. The servlet handler should return the job number to the client which in turn should poll the server every 10 seconds for a response. When a response is received with a completed task, the result of the dictionary lookup should be displayed in the browser.

2. An interface called ***DictionaryService*** should expose a remote method with the following signature:

   public String lookup(String s) throws RemoteException;

   where *s* is the string to lookup in the dictionary, and the String returned is either the dictionary definition of s or the text "String not found". In the ***DictionaryServiceImpl***, before looking up the query string in the dictionary the thread should be put to sleep for a time, i.e. *Thread.sleep(1000)*, to slow the service down and simulate a real asynchronous service.

3. Deployment and Delivery

The project must be submitted by midnight on Friday 22th December 2017 using both Moodle and GitHub.

Use the package name ie.gmit.sw throughout your project

- **GitHub**: submit the HTTPS clone URL, e.g. https://github.com/my- account/my-repo.git, of the public repository for your project. All your source code should be available at the GitHub URL.

- **Moodle**: submit a Zip archive using the Moodle upload facility (under "Main Assignment (50%) Upload". Ensure that the name of the Zip archive is be *{id}*.zip where *{id}* is your GMIT student number. The Zip archive should contain the following files and structure:

File	Description
README.txt	Contains a description of the application, extra functionality added and the steps required to run the application. This file should BRIEFLY provide the instructions required to execute the project.
job-server.war	A Web Application Archive containing the resources shown under Tomcat Web Application. All environmental variables should be declared in the file WEB-INF/web.xml. You can create the WAR file with the following command from inside the "job-server" folder: **jar –cf job-server.war ***
dictionary-service.jar	A Java Archive containing the RMI Dictionary Service and a ServiceSetup class with a main() method. The application should be run as follows: **java –cp ./dictionary-service.jar ie.gmit.sw.ServiceSetup** You can create the JAR file with the following command from inside the "bin" folder of the Eclipse project: **jar –cf string-service.jar ***

4. Extra Features

25% of the project marks will be awarded for documented (relevant) extra features. Some example extra features:

- Add functionality which allows the client JSP to add/remove/modify entries in the dictionary
- Add multithreading functionality, so that multiple RMI clients can make queries to the RMI Dictionary Service concurrently. Could be implemented using e.g. Java Thread Pools.

5. Marking Scheme

Marks for the project will be applied using the following criteria:

Marks	Category
(30%)	Tomcat Web Application (including message queues)
(30%)	Asynchronous RMI Service (including RMI client in Tomcat app)
(20%)	Packaging & Distribution (Moodle and GitHub) Documentation (Readme.txt and code comments)
(20%)	Documented (and relevant) extras (e.g. functionality which allows new words to be added to the dictionary, multiple RMI client threads)

Each of the categories above will be scored using the following criteria:

- 0–30%: Not delivering on basic expectation
- 40-50%: Meeting basic expectation
- 60–70%: Tending to exceed expectation
- 80-90%: Exceeding expectations
- 90-100%: Exemplary

6. Additional Information

You may use any format which you wish to store the dictionary data on disk. The use of a full dictionary is not required, a sample dictionary containing ~50 words is sufficient to satisfy the minimum requirements for the project.

Extra features – the 1913 edition of Webster's Dictionary is available for free at the following link, if you wish to use a complete dictionary as an additional feature: http://www.gutenberg.org/ebooks/29765. Feel free to use an alternate dictionary definition list if you prefer (several are available online).

7. Copying & Plagiarism

Plagiarism is the passing off the work of another person as one's own and constitutes a serious infraction which may result in marks deducted and/or disciplinary proceedings. This assignment is an INDIVIDUAL assignment. While collaboration with other class members is acceptable for high-level design and general problem solving, you must individually code, implement and document your own submission.