LTAT.06.007 Distributed Systems
Practical Seminar 5

Farooq Ayoub
TEACHING ASSISTANT

Tartu, Estonia 10/03/2021
Recap

• Insights about different Performance Metrics
• Visualized Performance Metrics using JMeter Modular dashboard
Agenda

• **Goal:** To understand code migration through transparent communications in distributed systems.

• **Content:**
  - Java reflection utility
  - Implement code migration at method level of any sorting algorithm, e.g., bubble sort, quick sort, etc.
  - Sockets

• **Quiz**

After this lecture, you should be able to:

• Understand how code migration works
Session Content

Description

- Code migration from one process to another occurs using RPC (Remote Procedure Call) implementations.
- However, code also can be captured during runtime and migrated to external processes without modifying the binaries of the application being executed. This is called code offloading.
- In this practical seminar, we will explore how to migrate code during runtime between processes.
- We will combine RPC and socket primitives to achieve this.

Observation

Instructions to complete this practical session can be found in the course website: https://courses.cs.ut.ee/2021/ds/spring/Main/Instructions1
Java Reflection Utility

• Feature in java programming language that allows us to examine or modify behaviour of methods, classes, interfaces at runtime.
  • Provided by `java.lang.reflect` package
  • gives us information about the class to which an object belongs and also the methods of that class which can be executed by using the object.
  • Through reflection we can invoke methods at runtime irrespective of the access specifier used with them.
Remote Method Invocation (RMI)

- Client makes a call to the remote object, it is received by the stub which eventually passes this request to the RRL.

- When the client-side RRL receives the request, it invokes a method called `invoke()` of the object `remoteRef`. It passes the request to the RRL on the server side.

- The RRL on the server side passes the request to the Skeleton, which invokes the required object on server.

- The result is sent back all the way to client.
Marshalling and Unmarshalling in RMI

- **Marshalling:**
  - Whenever a client invokes a method that accepts parameters on a remote object, the parameters are bundled into a message before being sent over the network.
  - These parameters may be of primitive type or objects.
    - If primitive type, the parameters are put together and a header is attached to it.
    - If the parameters are objects, then they are serialized.

- **Unmarshalling:**
  - At the server side, the packed parameters are unbundled and then the required method is invoked.
Code Offloading

- Offloading is the opportunistic process that relies on remote servers to execute code delegated by a Client.
Session Instructions at Course Page
Quiz

Content

- Lecture 5 (Communications)
- Two attempts
  - One in Seminar Session
  - Next available until Monday 23:50 (Deadline)
- Open Quiz in Moodle
- Total Quiz Points = 100

Observation

Quiz review is available after the quiz is closed
Questions?

E-mail: farooq.ayoub.dar@ut.ee