Algoritmos e Sistemas Distribuídos- Project 1 (Default Variant)

João Carlos Antunes Leitão

NOVA Laboratory for Computer Science and Informatics (NOVA LINCS)

and

Departamento de Informática

Faculdade de Ciências e Tecnologia

Universidade NOVA de Lisboa

V 2.0 β

30^{th} September 2018

1 Overview

This document discusses the second ASD project for 2018/19. There are three options: A, B, C. These options all evaluate for a final project grade of 20. They differ both on the concrete objectives and on the number of students in the group. Regarding this last aspect, option A aims at groups composed of 3 students, option B targets groups of 2 students; and finally, Option C aims at allowing students to pursue an individual project that can has additional freedom.

For Option A and B students are free to use any programming language of technologies they feel more comfortable with. The concrete suggestion is for students to use the Scala/Akka ecosystem. Using Akka, you probably want to have each process in the system to be modeled as a set of Actors, where each actor represents one protocol used by that process. Notice that, you cannot make use of any mechanism provided by the runtime to get knowledge about the system membership. Java is also another option that students, although this might bring some additional complexity to your implementation. Students doing options A or B should discuss with the course professor before starting to use different programming languages.

Option C is the continuation of the individual project assigned to students in the first project.

Evidently, each student should only do one of the available options for the project, and they should strive to maintain the same groups as in the first project

Below each of the options is discussed.

Part I

System Description for Option A and Option B

The application layer (i.e, the service) is a simple key value store, where all replicas should have the same information. The service itself provides two client operations to manipulate the key value store: read(Key) and write(key, value). For simplicity both Keys and Values can be modeled as strings. As an implementation suggestion, students can, at the application level, rely on a HashMap. Below you can find the full description of the semantic of client operations.

read(K): Returns the current value associated with key K (or an indication that no value is associated). This operations has two possible ways to operate. The *strong consistent* version should return the value written by the most recent completed write operation over key K. This implies that the operation has to be ordered in the state machine (using the appropriate variant of Paxos). The *weak consistent* variant can return the most recent value known at the replica that processes the read. In this case the operation does not has to be ordered in the state machine.

write(K,V): This operation associates value V to key K. It should return to the user the previous value associated with key K. The operation must be ordered in the state machine using Paxos.

In addition to this the service should also support two additional management operations: AddReplica(Replica) and RemoveReplica(Replica). Replica can be any data format that assists students in their implementation, a suggestion is to provide the necessary information for contacting that replica (e.g, IP + PORT).
At bootstrap the replicas should be made aware (through a configuration file) of the initial set of replicas. This information should be used to initialize the Paxos/Multi-Paxos (see below) component. Replicas that fail should be automatically removed from the system.

Part II

Option A: Using Multi Paxos

In this option the students are asked to implement a replicated service using state machine replication. In this variant the state machine replication layer should rely on Multi Paxos. In this variant the state machine should be notified by the Multi Paxos layer about leadership changes, to enable the state machine layer to redirect client requests to the current leader for ordering.

Part III

Option B: Using Paxos

In this option the students are asked to implement a replicated service using state machine replication. In this variant the state machine replication layer should use Paxos.

Part IV

Option C: Individual Project

Continuation of the individual project from the first phase.

Part V

Other Project Considerations

2 Additional Implementation Considerations

At bootstrap, all replicas of the system should receive information (e.g., a configuration file) with the IP and port of all replicas at bootstrap time. This information should be passed both the Paxos and State machine layers to setup their initial parameters of execution (e.g., the majority size in Paxos). In the case of option A, this initial configuration should not define the initial leader of the system. That should be asserted using the regular mechanisms of the Multi Paxos solution.
Students should consider using batching to accelerate the execution of the state machine in their replicated system. Students might also consider to avoid batching for operations that manipulate the membership of the replicated systems (i.e., add replica and remove replica operations).

3 Experimental Evaluation

The students should build a client application that executes operations over the replicated system. In this context the client application might resort to multiple execution threads, where each thread is perceived as being an individual client. Each individual client executes only a single operation at a time, and might timeout in case of not receiving a reply to its operation. In this case the client should resubmit its operation maintaining the identifier of the operation (proposal is to use client:port:timestamp as identifier, assuming each client thread to use a different port for communicating).
Notice that clients might resubmit operations that were already executed by the state machine and for which the client did not receive the request). This should be handled to avoid repeating the execution of such operation. To do so, the application layer in the server side should store the replies generated for the last operation issued by each individual client, so that repeated operations can be identified and replied to, without resorting to the state machine layer.
Experiments should report the latency and throughput of the designed system (ideally through a latency throughout graph) for different number of concurrent clients. Additionally, students might also try to evaluate the effects on latency and throughput of replicas failing or joining the system).

4 Evaluation Rules & Criteria

The project includes both the delivery of the code and a written report that must contain clear and readable pseudo-code for each of the implemented layers alongside a description of the intuition of the protocol. A correctness argument for each layer will be positively considered during grading. The written report should provide information about any experimental work conducted by the students to evaluate their solution in practice (including results).

The project will be evaluated by the correctness of the implemented solutions, its efficiency, and the quality of the implementation. With a distribution of 50%, 25%, and 25% respectively for each of the layers.
The quality and clearness of the report of the project will impact the final grade in 10%, however the students should notice that a poor report might have a significant impact on the evaluation of the correctness of the used solutions (which weight 50% of the evaluation for each component of the solution).
Each component of the project will have a maximum grade of (out of 20):

Developed Distributed Protocols: 12/20

Test Application and Evaluation: 6/20.

Written Report: 3/20

(Yes, the sum of this is not 20)

5 Delivery Rules

Delivery of all components of the project is due on 8 December 2018 at 23:59:59.
Discussions of both projects will happen in the the 17 of December. I will only download mails with the project deliveries on December 14th.
The project should be delivered by e-mail with a subject stating "ASD Project Delivery 2 - <Student Numbers>"