Distributed Enabling Platforms

Final Exam Project
2014/2015

Project Goals

- Projects <u>require</u> students to <u>develop software</u> in <u>Java</u>.

- You must prove you master some concepts discussed during the course, <u>in theory</u> and <u>in practice</u>.

- Some project ideas are <u>proposed by the professor</u>.

- Students can propose their own project ideas, but they <u>must be accepted</u> by the professor.

- Projects are mandatory to successfully pass to the final <u>oral examination</u>.

Project Delivery and Discussion

- A project must be delivered at least two weeks before the oral examination.

- There are no pre-set dates: contact the professor when ready.

- Deliver (via email, code repo or similar ways) the source code, complete with any data and documents to allow anyone to compile, deploy and test the software.

- Deliver via email an electronically written project report describing the project and the design choices, implementation details, testing procedures and experimental evaluation.

- Every group (1 or 2 students max) must present 10 slides per student during the oral examination discussing the project and its outcomes.

- Be ready to answer queries on code organization, design choices, etc.

Common Requirements

- You must program in Java.

- You must collect and provide input data.

- You must implement testing procedures to debug your code.

- You can use common support libraries (e.g., junit, log4j, etc) that do not overlap with the project goals. Describe them and motivate their use in the project report.

- Do not hard-code configuration parameters, use configuration files/ services.

- You must test all the functionalities of your developed system/service and present and discuss the testing results in the project report.

Common Requirements

- Your software must run on a distributed platform (even if developed in pseudo-distributed modes).

- Your software must deal to some degree with autonomous entities contending shared resources.

- Your software must deal to some degree with some form of shared state distributed among nodes.

- Your software must deal to some degree with elasticity, scalability and/or fault tolerance.

Project 1: PAD-FS

- Implement a distributed file system supporting a consistency model you choose.

- Simple API: put, get, list.

- Flat or hierarchical.

- A fully-fledged solution like Dynamo is not requested.

Project 2: PadBook

- Implement an eventually consistent distributed publish-subscribe system with a social network interface.

- Design the API keeping in mind how it will be exploited.

- Choose the right consistency model!

Project 3: Pad Matrix

- Implement a software library to perform basic linear algebra operations on arbitrarily large matrices using Hadoop

- Must be a library, focus on simplicity of use for a programmer.

- Pick up some operations like matrix inversion, matrix transposition and matrix product.

- I will try it on an real Hadoop cluster to test its usability.

Project 3: Pad Matrix

- Implement a software library to perform basic linear algebra operations on arbitrarily large matrices using Hadoop

- Must be a library, focus on simplicity of use for a programmer.

- Pick up some operations like matrix inversion, matrix transposition and matrix product.

- I will try it on an real Hadoop cluster to test its usability.