Instituto Superior Técnico, Universidade de Lisboa
Large-Scale Systems Engineering

Project – Stage I

Goals

The ESLE project is an open-ended project intended to introduce students to hands-on experience with the deployment, management, and analysis of a distributed system in a realistic scenario close to what they will find in the industry or academia.

Assignment

Students need to choose an existing system which will be progressively studied throughout the project. By the end of the project, students should have a deep understanding of the system, be familiar with its scalability and performance characteristics and on which workloads and deployment scenarios it works best.

Ideally, students should select a system that caters to their interests. The selected system should obey to the following requirements:

- **Open-source** - so that the system is easier to study and, if needed, modify.
- **Client-server architecture** - the system should have a clear distinction between the 'server' and the 'client'. This is to simplify the analysis as the characteristics of peer-to-peer systems are harder to analyse.
- **Distributed** - which usually result in more interesting properties than scale up systems.
- **Real system** - published in one of the top-tier system's conferences (see the list below). Alternatively, students can select an open-source system/project they are (or would like to become) familiar with.

Examples of candidate systems include databases, graph processing systems, application servers, coordination services, network virtual functions, machine learning systems, among others.

The selected system should satisfy all the above requirements. Before settling in the target system, students are expected to discuss the requirements above with the professor during the laboratories.

For the first stage of the project students are expected to:

- Containerize the system using Docker and Docker Swarm. Students are free to use Kubernetes, Apache Mesos or other alternative orchestration platforms.
- Devise a simple benchmark that allows to assess the behaviour of the system.
- Analyse the scalability properties of the system.
- Analyse how the scalability properties can be explained considering the Universal Scalability Law.
- Decompose a request into a pipeline of stages.
- Study which portions of the pipeline are the bottleneck (or can become one) and reason about which of the techniques studied in the classes could be applied to remove or mitigate the bottleneck.

In this stage of the project, it is recommended that students develop the project in their own machines. We will have access to Google Cloud resources, but as credits are limited it is better to save those credits for later in the project. Further instructions to claim the credits will be made available in the course page. Cloud providers such as Amazon Web Service, Microsoft Azure or the Google Cloud platform itself usually have free tiers for students so feel free to try those as well.

Project files

The project should be hosted in a private git repository which should be shared with user *miguelammatos* on GitHub. Platforms such as GitHub offer free accounts for students. If the group cannot get access to a free private repository, contact the professor. All the artifacts, scripts, code, measurements, and the report should be present in the repository, together with a top level README.md file that explains: *i)* the structure of the repository and *ii)* how to run a small demo of the system with a few clients and servers.

Submission

The submission of the project will be done through Fénix.

The submission will consist of a report in the LLNCS format [1] with up to 5 pages, excluding references. All the other material relevant to the evaluation should be available in the repository at the time of the submission, as specified above. If deemed necessary, students can include annexes in the report but it should be possible to assess the work without considering the annexes.

The report should include the following:

- **Identification of the group members**: Name, Email and Student Number, and Group Number.
- **Introduction**: brief presentation of the selected system and the justification for its choice. **The report should also specify the git commit identifier that should be considered for evaluation purposes. Failure to do so will result in a grade penalty.**
- **System Description**: detailed description of the system, including an analysis of its main characteristics. Description and justification of the selected workload.
- **Results**: presentation and discussion of the scalability, performance and stage pipeline experiments conducted, together with any additional information the group deems relevant to understand the obtained results.
- **Conclusion**: discussion of the main insights obtained.

The deadline for the submission of the first stage is **October 14 at 19:00.**

System's conferences

The selected system can come from any of these conferences (restricted to the editions between 2019 – 2022):

- SOSP - ACM Symposium on Operating Systems Principles
- OSDI USENIX Symposium on Operating Systems Design and Implementation
- NSDI - USENIX Symposium on Networked Systems Design and Implementation
- Eurosys - European Systems Conference
- SIGCOMM - ACM Conference on Special Interest Group on Data Communications
- Middleware - ACM/IFIP/USENIX International Middleware Conference
- DSN - IEEE/IFIP International Conference on Dependable Systems and Networks
- ATC - USENIX Annual Technical Conference
- CoNEXT - International Conference on emerging Networking EXperiments and Technologies

As an example these are some systems that have been selected in previous (which **cannot** be selected this year): MaxiNet (IFIP'14), Yesquel (SOSP'15), Apache Flink,

Apache Kafka, ETCD, ElasticSearch, Tahoe-LAFS, FreeFlow (NSDI'19), Apache Cassandra.

Bibliography

[1] Information for Authors of Springer Computer Science Proceedings. https://www.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines