Institute of Technology
University of Washington – Tacoma
Instructor: Wes Lloyd

Term Project Proposal

Version 0.1

Due Date: Tuesday April 11th, 2017 @ 11:59 pm

Objective

The TCSS562 term project consists of building a prototype of ***two or more*** implementations of a back-end cloud service. The goal is not to produce a ***production*** quality service, but to replicate using ***at least two*** separate cloud technologies an equivalent prototype-quality implementation that enables the comparison of competing cloud technologies and/or service offerings. Once the prototypes are functional, benchmark tests will be performed to assess performance characteristics, cost, etc. The ultimate goal, is to produce a report where we learn about the internal workings of the competing cloud technologies so that we can quantify their performance and cost tradeoffs. Your project should help a software engineer or system analyst answer the question, "Why should I choose technology A over technology B for my cloud implementation?". In addition, this exercise will allow groups to explore cloud service offerings to learn about the respective features they offer and share these findings with the entire TCSS 562 class.

The following are some potential projects. By no means is this an exhaustive list of projects. Groups are encouraged to perform so initial exploratory research to determine the types of cloud services they would like to explore for the term project.

For self-hosted services, the intent is to compare a cloud-hosted variant with a self-hosted software application running on a virtual machine on the cloud to provide an "equivalent service". For example, redis and mongodb can provide key-value storage as an application installed onto a virtual machine. From a functional point-of-view this is equivalent to S3, but with a slightly different API for access.

Each term project proposal should propose 1 or 2 questions which they will intend to answer with the investigation.

1. Object/blob storage services comparison: Amazon S3, Google blobstore, Azure blobstore, self-hosted key-value storage:

   Which blob storage is most performant?
   Which blob storage is most cost effective?
   How does the type and size of data impact performance and cost for different blobstores?

2. Public cloud object/blob storage vs. self-hosted

   Why should I use public cloud blob storage vs. self-hosted? Obviously for availability, and fault tolerance, but what if we don't care about these criteria for a particular software application? What if we simply want cheap and fast temporary blob storage? Where is "the line drawn" for cost effectiveness? Are the publicly hosted cloud service alternatives actually less expensive? If so, which ones?

3. Cloud relational database services comparison: Amazon RDS, Azure SQL Database, Google Cloud SQL, Heroku

   Which relational database cloud service is best? For this project the team would locate a sample relational database online, or acquire a database from the instructor to host, and conduct a number of database performance experiments to evaluate the systems.

4. Cloud relational database service comparison vs. self-hosted

   What is the cost of relational database services compared to self-hosted databases in the cloud? For example, hosting a PostgreSQL DB on an EBS-backed Amazon EC2 VM already guarantees high availability and redundancy of the data (EBS volume). On the surface the only feature we are lacking is automatic database backups from accidental data deletion. What is the additional costs of Amazon RDS? And what are the full features gained? Are the costs justified? How does the performance compare?

5. Cloud application containers comparison for web application hosting: Amazon Elastic Beanstalk, Heroku, Google App Engine, others

   What is the advantage of PaaS Java web application hosting? Which public cloud container provides the best performance? How do the costs compare? How does the characteristics of the application impact performance and cost? Is one cloud provider a clear winner?

6. Cloud application container hosting vs. self-hosted

   Why should we use PaaS to host our web applications? What if we are not concerned about scalability? How does performance and costs compare for web application hosting using PaaS vs. self-hosting by using an Amazon EC2 VM with a web application container such as Tomcat or GlassFish?

7. Microservices hosting comparison: AWS Lambda, Azure Functions, Iron.io

   What are the key differences among public cloud microservices hosting platforms? How does their performance and scalability scale?

Other ideas from slide set 1:

1. Microservices hosting vs. self-hosted microservices
2. Amazon Container Service, Azure Container Service comparison
3. Amazon Container Service vs. self-hosted container orchestration on VMs (Docker SWARM or Kubernetes)
4. Amazon Elastic Load Balancer vs self-hosted load balancer (haproxy or nginx)
5. Cloud virtual machine performance comparions: AWS, Azure, Google
6. Storage system comparison: Amazon EBS, Amazon EFS, Amazon Ephemeral Storage
7. Queueing services comparison
8. Virtual machine imaging comparison

9. Private cloud projects

1 Requirements

The following are key requirements of the project proposal:

Each team will submit a 1 to 2 page project proposal that does not exceed 6000 characters including spaces.

The proposal must identify:

1. The member names of the project group.
2. The name of the group project coordinator. The project coordinator will be responsible for scheduling and arranging group meetings and work sessions, creating agendas for project check-in meetings for TCSS 562, and ensuring that tasks are assigned to group members.
3. At least 2 potential competing cloud system alternatives to compare. It is ok to make changes as needed later on throughout the project.
4. At least 2 questions which will be studied by the prototype systems evaluation. It is ok to make changes to the questions as needed later on throughout the project.
5. At least 2 metrics by which the systems can be compared based on a potential prototype implementation. For example, if you're comparing databases, then you might identify the number of SQL queries per second that can be performed as well as the average response time of individual queries in seconds. It is ok to make changes as needed later on throughout the project.
6. At least 2 references for each cloud system which discuss the capabilities (features) of the systems including potentially details about performance and cost.

If available, good proposals will:

1. Identify prior performance comparison studies which relate to the proposed project. Use scholar.google.com to search for research papers. Studies could include research papers, or IT industry articles which discuss the tradeoffs of technology alternatives. Research papers published at IEEE, ACM, or USENIX conferences or journals are rigorously peer-reviewed and are generally considered of higher quality.
2. Use the ACM conference paper template posted on the course website. (not required)

2 Future Deliverables

The final project will involve a group project presentation in the final class on Thursday June 1st, or during the final exam session on Tuesday June 6th. Requirements of the final project will be provided later on.

The final project will also involve a written report in ACM conference format. In the project report, groups should be prepared to describe their prototype system, identify the evaluation questions, describe benchmark tests, and create tables and graphs to describe results of benchmark tests. Project reports will also describe any related work or comparison studies, why the evaluation questions are relevant and interesting, and reflect on the results of the study. Additional details and requirements for the final project report will be provided later on.

3 Weekly Project Check-ins (10% of the TCSS 562 course grade)

Once groups are formed, each group will have a scheduled weekly 30-minute meeting with the instructor on Tuesday or Thursday between 2:30 - 5pm to discuss ongoing project work, to help resolve any potential roadblocks, gain suggestions on how to build the prototype systems, and make project refinements as needed. Each group will be assigned a regular meeting time each week. Each team will have 6 meetings starting the week of April 17th, with an optional final meeting in the last week of class.

4 Submission Deadline

Project proposals should be submitted in PDF format on Canvas no later than 11:59pm on Tuesday April 11th.

Change History

Version	Date	Change
0.1	03/29/2017	Original Version