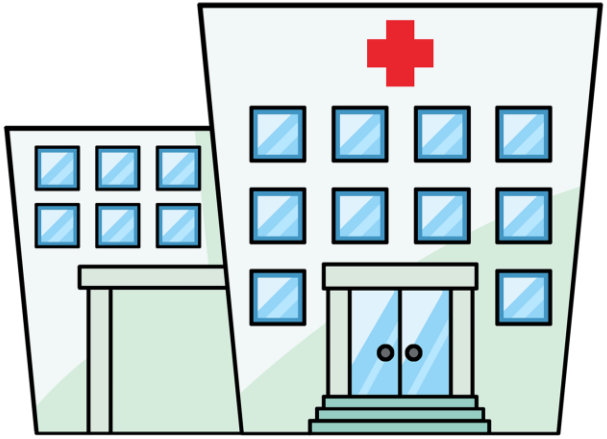
Happy Patients Hospital redesign

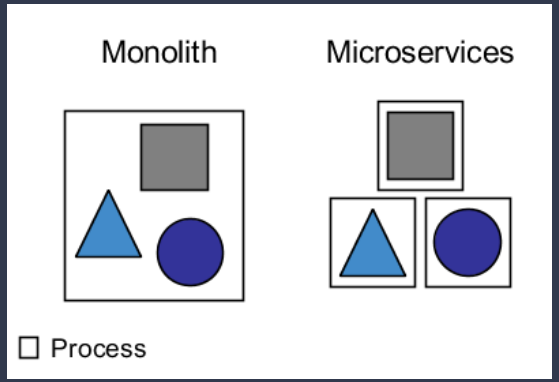
CS249 Fall 2017
Group 11

Introduction

- Happy Patients Hospital opened in 1880
- In 1990 started storing all their data in a **relational database**
- Making changes and deploying became difficult
- Happy Patient hospital wants to make a change from a **monolith** application to a scalable infrastructure



Goal of Project

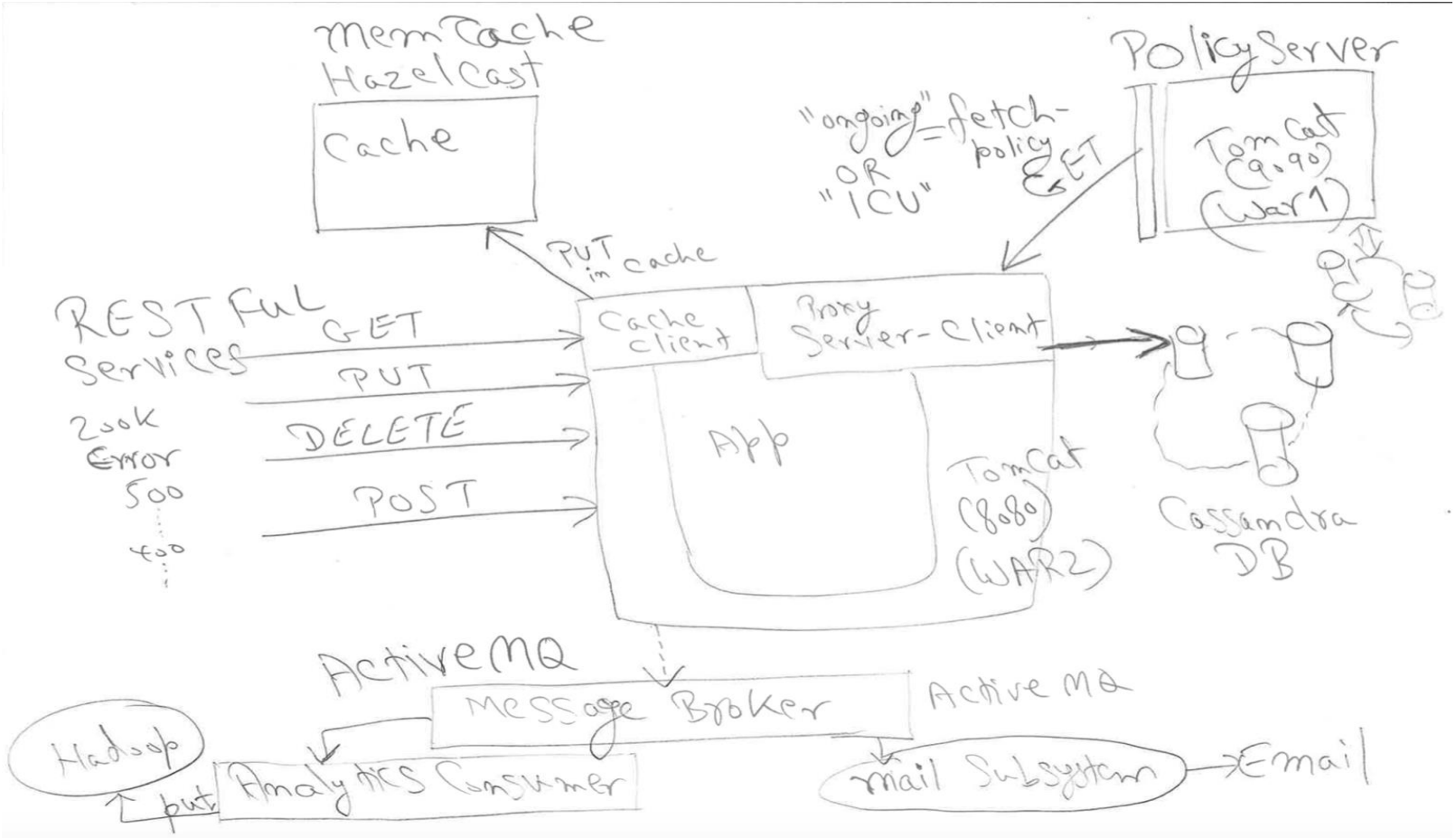


- Using the following technologies change the infrastructure of Happy Patient Hospital to a scalable and extendable
- Technologies:
  - Apache Tomcat
  - Apache Cassandra DB
  - Hazelcast IMDG
  - Apache ActiveMQ

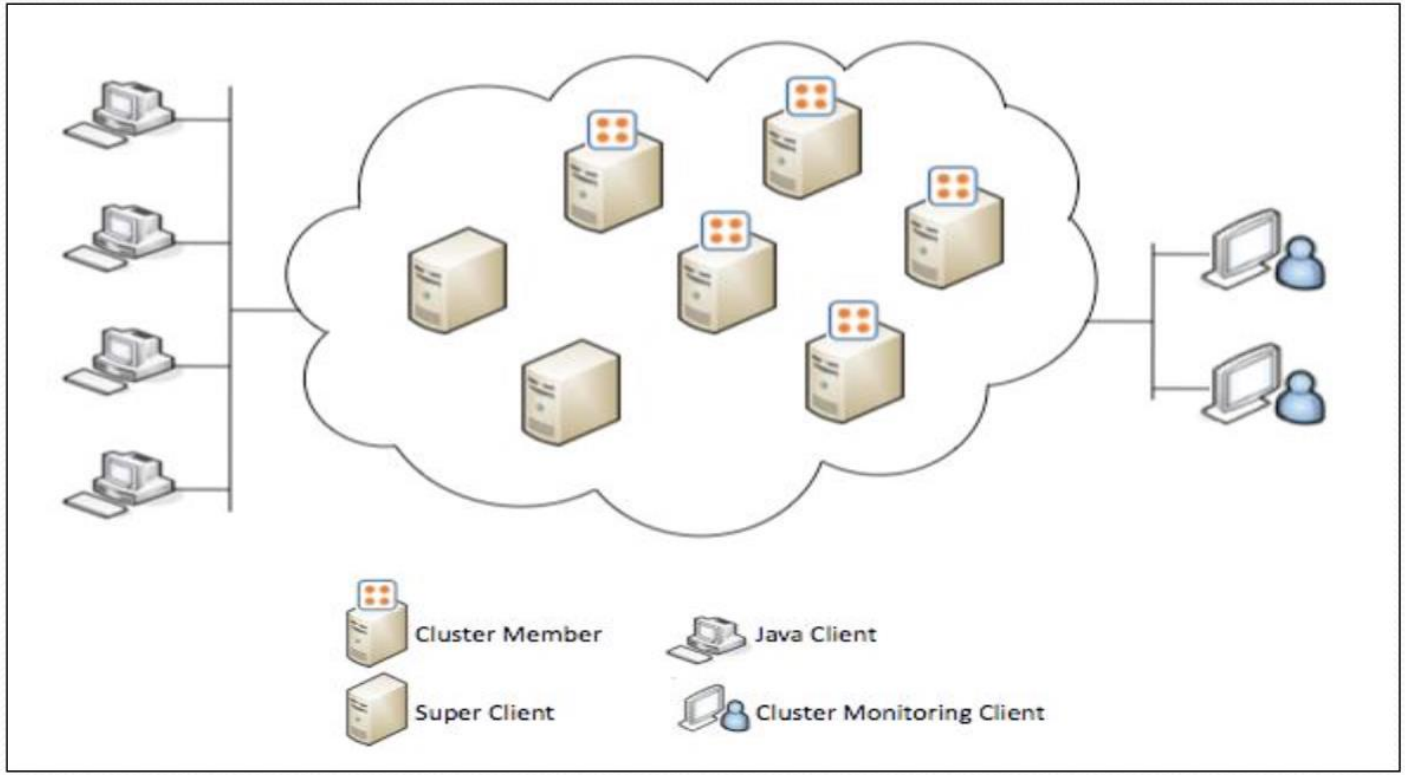
hazelcast

Apache ACTIVEMQ

Architecture diagram highlighting all major components.



Hazelcast architecture



[Source: http://www.hazelcast.com/documentation.jsp]

1- What you had to learn from scratch?

- New to distributed database,so it took time & effort to config, setup,learn distributed DB
- Learn Cassandra followed by its architecture, installation, and important classes and interfaces.
- Learnt from scratch, on how to perform operations on DB, such as create, alter, update, and delete on keyspaces, tables, and indexes using CQL as well as Java API.
- Learnt using Cassandra as a persistent store along with Hazelcast Mapstore Implementation
- Learnt hazelcast cache implementation in my application
- ServletContextListener and how to initialize when webapp starts
- Triggering context based events such as reading property from file with hot deploy

2- What you already knew?

- Knew concepts of RESTFUL
- Java API for RESTful Services is spec that provides support in creating web services according to the Representational State Transfer architectural pattern.
- Knew basic of WAR (Web ARchive) file
- Basic previous understanding of Tomcat implements web specifications including Java Servlet, JavaServer Pages , WebSocket, HTTP web server environment in which Java code can run.
- Maven and accessing central repo’s

3- Did you do any POC ?

We had written POC's for Cassandra integration, HazelCast Implementation, ActiveMQ which have all been integrated into our final webapp

4- Submit the HLL / architecture diagram highlighting all the different components.

Slide # 4

5- Submit any additional diagram that you think would be helpful for clarity

Slide #5

6- How did you set up the environment?

a) What problems did you face?

- Creating clusters on cassandra
- Updating the policy as a string vs having a property file

b) How did you solve the problems?

- Using KeyspaceRepository
- Policyserver used contextInitialized and event listener for updates

c) Did you use any other open sources/ tools etc? (apart from what was required by the project)

- Cqlsh to check Cassandra Entries
- Log4j for debugging purposes

7- What all you referred to learn concepts or see examples (Provide info on books, Links, papers etc..)

http://hc.apache.org/httpcomponents-client-ga/examples.html
http://tomcat.apache.org
http://www.vogella.com/tutorials/EclipseWTP/article.html
http://tomcat.apache.org/tomcat-6.0-doc/security-manager-howto.html
https://stackoverflow.com/questions/37678785/hazelcast-write-behind-using-cassandra
https://stackoverflow.com/questions/30080886/hazelcast-cache-implementation-in-my-application
http://docs.datastax.com/en/cassandra/3.0/
https://www.youtube.com/watch?v=fspXzjwfii0&t=46s
https://hazelcast.org/documentation/#imdg

8- Give a code walkthrough : Approach , Demo
Talking points