CS 683 Emerging Technologies
Fall Semester, 2008
Doc 14 SQS and Scaling
Oct 28 2008

References

Amazon Simple Queue Service Developer Guide API Version 2008-01-01, http://docs.amazonwebservices.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/

New Features for Amazon EC2 Coming Soon, http://aws.amazon.com/contact-us/new-features-for-amazon-ec2/

Microsoft Plans ‘Cloud’ Operating System, http://www.nytimes.com/2008/10/28/technology/28soft.html?_r=1&partner=rssnyt&emc=rss&oref=slogin

The CAP Theorem, http://camelcase.blogspot.com/2007/08/cap-theorem.html

Cloud computing news

EC2 Announcements

No longer in Beta

Windows Machine images in beta

Upcoming features

- Load Balancing
  - Automatically balance requests among EC2 instances
- Auto-scaling
  - Grow and shrink EC2 capacity to meet your demand
- Monitoring
  - Realtime monitoring of your EC2 usage
- Management Console
  - Web interface configure, manage & access AWS

Source http://aws.amazon.com/contact-us/new-features-for-amazon-ec2/

Microsoft Cloud Computing

Late next year Microsoft will make available Azure, a cloud OS

http://www.microsoft.com/azure/default.mspx

Source http://www.nytimes.com/2008/10/28/technology/28soft.html?_r=1&partner=rssnyt&emc=rss&oref=slogin

Scaling Web Applications

http://highscalability.com/

Planing for Scaling

ORM for Data Partitioning and Query Splitting
Monitoring process, resources, and uptime
Performance Testing and Capacity Planning
Static vs. Dynamic Content splitting
Bundling and Compressing JS and CSS
Logging
Pragmatic Caching
Functional Decomposition
Deployment
Asynchronous Practices
Lean Applications

From Joseph Kent Langley's blog at: http://www.productionscale.com/home/2008/10/24/things-to-consider-when-planning-your-application-system-and.html

ORM for Data Partitioning and Query Splitting

Split queries between updates and deletes from the start

ORM - Object Relational Mapping

Maps between objects and database tables

ORM Example - Tables

students

firstname	lastname	phone	code_id
John	Smith	555-9876	2000
Ben	Oker	555-1212	9500
Mary	Jones	555-3412	9900

codes

code	major
2000	Art
3000	History
9500	Electrical engineering
9900	Computer Science

ORM Example - SQL

```
INSERT
   INTO students (firstname, lastname, phone, code_id)
SELECT
   'Roger' AS firstname,
   'Whitney' AS lastname,
   '594-3535' AS phone,
   codes.code AS code_id
FROM
   codes
WHERE
   codes.major = ‘Art’
```

ORM Example - Using Glorp

```
person := Person first: 'Roger' last: ‘Whitney’.
person phone: '594-3535'.
person major: 'Art'.
session beginUnitOfWork.
session register: person.
session commitUnitOfWork.
```

Monitoring process, resources, and uptime

- Process Monitoring
  - This makes sure things are running
  - Example tools
    - God, Monit, SMF

- Resource Monitoring
  - Monitor CPU, Memory, Disk Space, Networking, etc.
  - Example tools
    - Nagios, Ganglia, Munin, ZenOSS

- UpTime Monitoring
  - Example Tools
    - webmetrics and pingdom

Performance Testing and Capacity Planning

How does your application perform under load

What sort of load to you expect

Static vs. Dynamic Content splitting

Web frameworks have overhead

Use web server for static pages

Web framework handles only dynamic pages

Bundling and Compressing JS and CSS

Bundle and compress javascript and CSS files in web application

Bundling reduces number of TCP connects needed

Compressing reduces total size sent

Logging

Log your application

- access
- request information
- errors
- problems

Monitor logs

Pragmatic Caching

Web frameworks support caching of pages/data

Understand and use caching available

Functional Decomposition

Decompose application into separate functions

app servers, monitoring, log aggregation,
databases, message queues

As required use separate machines for each function

Asynchronous Practices

When possible make functions asynchronous

Does logging have to be done in real time?

- Send logging info to logger
- Application continues
- Logger queues requests

Beware of the CAP theorem

CAP Theorem

You can only have two of the three CAP properties at the same time

Consistency

All clients see the same view even with updates

Availability

All clients can find some replica of the data

Partition-tolerance

The system properties hold even when the system is partitioned

From http://camelcase.blogspot.com/2007/08/cap-theorem.html. For a proof see http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.20.1495.

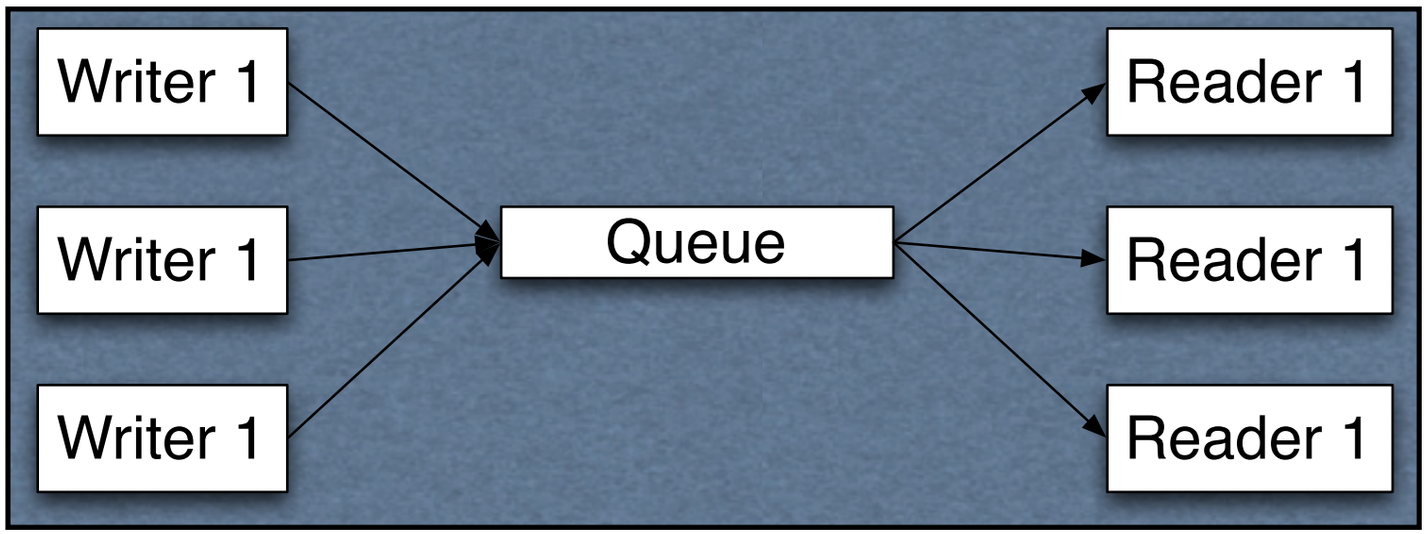
Deployment

Automate your deployment

Have roll back capability

Amazon Simple Queue Service

Message Queues



Unlimited queues and messages

- Each queue is named

Messages up to 8 KB in size

Multiple readers & writers allowed per queue

Redundant infrastructure

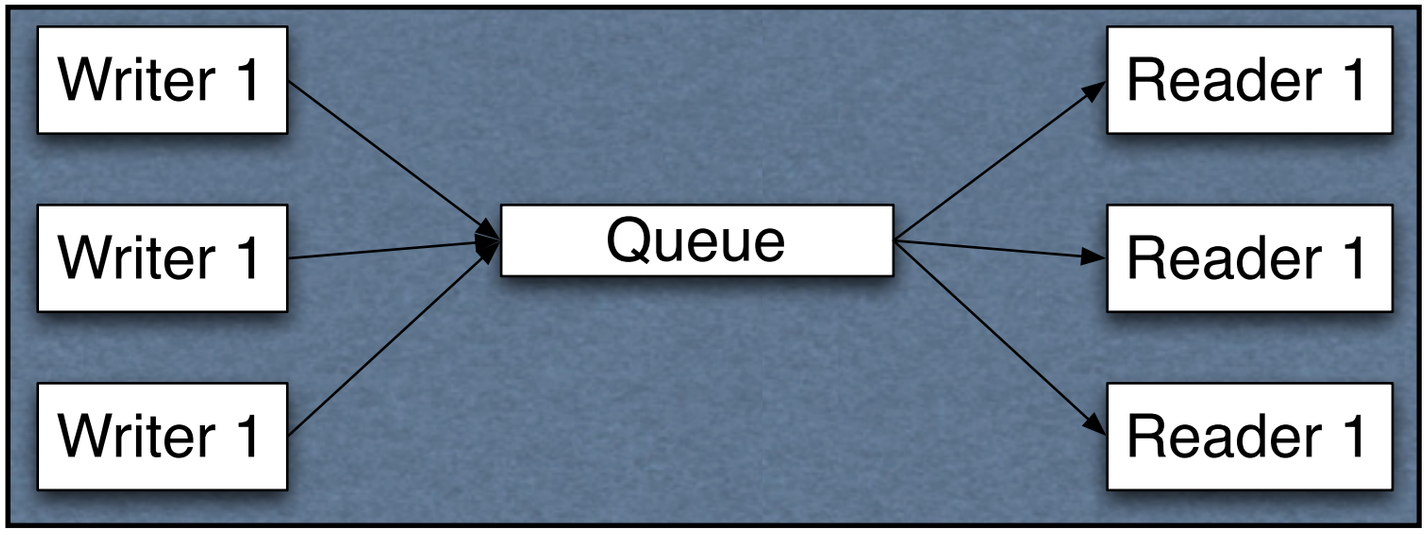
- Guarantees delivery of message at least once

Access to Queue

Via Soap or REST

From any location

Message Order



Best effort made to keep messages in order

If exact order is important add sequencing information to message

Not easy

- Multiple machine can write
- Timestamps between machine not synched

At-Least-Once Delivery

Remember the CAP theorem

Messages are stored on multiple machines

A delete may not delete all copies of a message

If a server is down during the delete

You may get the message a second time

Your application has to handle duplicate messages

Message Sample

CAP again

When you query for messages

A subset of the SQS servers are queried

May get no messages when messages exist

When have small number of messages (under 1000)

May not have access to all messages

Visibility Timeout

Reading a message does not delete the message

The reader could die before processing the message

Reader has to delete message when done processing

After being read a message is invisible in the queue

Message becomes available after the visibility timeout ends

Unless the reader deletes the message

Operations

CreateQueue
ListQueue
GetQueueAttributes
SetQueueAttributes
DeleteQueue

SendMessage
ReveiveMessage
DeleteMessage

Queue Attributes

Visibility Timeout