Measuring End-User Availability on the Web: Practical Experience

Matthew Merzbacher
UC Berkeley

Dan Patterson

Authors

- Matthew Merzbacher
  - Assistant Professor, Mills College (Oakland, CA)
  - Visiting professor @ Berkeley
  - Eclectic collection of papers from fault tolerance to animation to Programming languages to Databases.

- Dan Patterson
  - Undergraduate @ Berkeley

Introduction

- Availability, performance, QoS important in Web Svcs.
- End user experience -> meaningful benchmark
- Long term experiments attempted to duplicate end user experience
- Find out what the main causes are for downtime as seen by end user.

Driving forces

- Availability/uptime in “9”s not accurate
  - Optimal conditions, not real-world
- Actual uptime to end users include many factors
  - Network, multiple sw layers, client sw/hw
- Need meaningful measure of availability rather one number characterizing unrealistic operating environment
The Experiment

- Undergrads @ Mills College/UC Berkeley devised experiment over several months
- Made hourly contact on a list of several prominent/not-so-prominent sites
- Characterized availability using measures of success, speed, size
- Attempted to pinpoint area of failures

Experiment (cont’d)

- Coded in Java
- Tested local machines as well (to determine baseline and determine local problems)
- Random minutes each hour
- Results form 3 types of sites
  - Retailer
  - Search engine
  - Directory service

Results

- Availability broken up into sections
  - Raw, local, network, transient
- Kinds of errors broken up into
  - Local, Severe network, Corporate, Medium Network, Server
- Was response upon success partial? How long?

Different Tiers of Availability

<table>
<thead>
<tr>
<th>Ignoring problems</th>
<th>All</th>
<th>Retailer</th>
<th>Search</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw (Overall)</td>
<td>.9305</td>
<td>.9311</td>
<td>.9355</td>
<td>.9267</td>
</tr>
<tr>
<td>Ignoring local problems</td>
<td>.9888</td>
<td>.9887</td>
<td>.9935</td>
<td>.9857</td>
</tr>
<tr>
<td>Ignoring local and network problems</td>
<td>.9991</td>
<td>.9976</td>
<td>1.00</td>
<td>.9997</td>
</tr>
<tr>
<td>Ignoring local, network, and transient problems</td>
<td>.9994</td>
<td>.9984</td>
<td>1.00</td>
<td>.9999</td>
</tr>
</tbody>
</table>
Types of Errors

- Local (82%)
  - Network: Medium (11%)
  - Severe (4%)
  - Server (2%)
  - Corporate (1%)

Local Problems

- Most common problem
- Caused by
  - System crashes, sysadmin problems, config problems, attacks, power outages, etc...
- All had component of human error, but no clear way to solve via preventative measures
- “Local availability dominates the end-user experience”

Lost Data and Corporate Failure

- Just because response was received doesn’t mean service was available
- Experiment kept track of pages that appeared to be of a drastically different size (smaller) as unavailable (i.e. 404)
- If international versions failed -> corporate failure

Response time

- Wanted to define what “too slow” is
  - Chart availability vs. time
  - Asymptotic towards availability of 1
  - Choose threshold, all response times > considered unavailable
  - Client errors most frequent type of error, then transient network
How long should we wait?

![Graph showing availability vs. time (sec) for retailer, search, and directory.]

Retrying

- To users, unavailability leads to retry at least once
- How effective is a retry?
  - Need to test for persistence of failures
  - Consistent failures indicate fault @/near server
- Persistent, non-local failures
  - Domain dependent

Retrying (cont’d)

- Retry period of 1 hour unrealistic
- As in brick & mortar, clients have choice
- #retries, time btwn retries, etc based on domain/user dependent factors
  - Uniqueness, import, loyalty, transience

Effect of retry

<table>
<thead>
<tr>
<th>Error Type</th>
<th>All</th>
<th>Retailer</th>
<th>Search</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>0.267</td>
<td>0.271</td>
<td>0.265</td>
<td>0.265</td>
</tr>
<tr>
<td>Medium Network</td>
<td>0.862</td>
<td>0.870</td>
<td>0.929</td>
<td>0.838</td>
</tr>
<tr>
<td>Severe Network</td>
<td>0.789</td>
<td>0.923</td>
<td>1.00</td>
<td>0.689</td>
</tr>
<tr>
<td>Server</td>
<td>0.911</td>
<td>0.786</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>Corporates</td>
<td>0.421</td>
<td>0.312</td>
<td>1.00</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Green > 80% Red < 50%
Conclusion

- Successful in modeling user experience
- 93% Raw, 99.9% removing local/short-term errors
- Retry produced better availability, reduced error 27% in local, 83% non-local
- Factoring in retries produces 3 “9s” of availability.
- Retry doesn’t help for local errors
  - User may be aware of the problem and therefore less frustrated by it

Future Work

- Continue experiment, refine availability stats
- Distribute experiment across distant sites to analyze source of errors
- Better experiments to determine better the effects of retry
- With the above, we can pinpoint source of failures and make more reliable systems.