Internet Services

Introduction to Internet Services
Richard Martin

Rutgers University
Department of Computer Science

January 2003

Outline

- Definition
- History
- Overview
- Fox Paper
- Brewer paper
- Open Problems

Definition

- What is an Internet Service?
- Any client server interaction over the Internet?
- What is a service?
  - Client is a user of the server
  - Shared resources
  - Information or transactions are the key abstractions, not packets or data

Examples

- Auctions
  - Ebay, yahoo auctions
- Trading
  - E*Trade, schwab
- Shopping
  - Amazon model
- Entertainment and news
  - Cnn, AOL
- Computing Services (grid)
  - Seti @ home,
- Communication
  - Hotmail, AOL instant messenger
Internet Service History

- 1970s: email + FTP
- 1980s: email, ftp, news
- Early 1990s: + gopher, wais (search)
- Mid 1990s: + web, (hypertext)
- added search later
- Late 1990s: + DB access
- Today: + radio, movie/TV shorts
- Tomorrow?

Lesions from Giant-Scale Internet Services

- Author: Eric Brewer

Challenges

- High availability
- On-line evolution
- Growth

Examples

<table>
<thead>
<tr>
<th>Service</th>
<th>Nodes</th>
<th>Queries</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX Web cache</td>
<td>&gt;1000</td>
<td>100K/day</td>
<td>4-CPUs DEC 4100s</td>
</tr>
<tr>
<td>Intranet search engine</td>
<td>&gt;1000</td>
<td>&gt;500M/day</td>
<td>2-CPU Sun Workstations</td>
</tr>
<tr>
<td>Geniebox</td>
<td>&gt;100</td>
<td>&gt;500M/day</td>
<td>PC Based</td>
</tr>
<tr>
<td>Anonymous Web-based e-mail</td>
<td>&gt;5000</td>
<td>&gt;100/day</td>
<td>FreeBSD PCs</td>
</tr>
<tr>
<td>google</td>
<td>&gt;10000</td>
<td>80+M/day</td>
<td>1-2 processor PC’s</td>
</tr>
</tbody>
</table>
Tier Functions

- Tier 1: Request Distribution
  - load balancers, front ends, TP monitors
- Tier 2: Web servicing, data aggregation
  - Web servers, distillers
- Tier 3: Persistent state
  - Databases and storage modules

Availability Metrics

- Classic analysis:
  - Mean time to Failure (MTTF)
  - Mean time to Repair (MTTR)
  - Average Uptime

New Metrics

- New, 2 dimensional space:
  - Yield: % of queries completed per second
  - Harvest: % of data available
**DQ Principle**

- Total amount of data deliverable is constant
  - Bandwidth constraint
  - Data per query * Queries/sec = C

- Service quality defined by how much of total potential, C, is delivered
  - New metric needed to reason about both data and queries

**DQ In practice**

Which method is a better upgrade strategy?

**When is DQ useful**

- Can the data be partitioned?
  - Different users -> different datasets

- Can it be replicated?
  - Auctions
  - Trading
  - Shopping
  - Entertainment and news
  - Computing Services (grid)
  - Communication

**Big Ideas**

- Clustering and tiers architecture/clusters are the only way to go. (why?)

- Dual metrics needed to capture availability

- Constant = DQ is the best way to measure availability
Cluster based Scalable Network Services

- Authors:
  - Armando Fox: Prof. at Stanford
  - Steve Gribble: Prof. at Washington
  - Yatin Chawathe: ICIRI Research Lab
  - Eric Brewer
  - Paul Gauthier: CTO of Inktomi

Paper ideas

- Argues for cluster architecture
  - Seems natural today, what was the big deal?
- ACID vs. Base
- Presents scaling study

ACID vs. BASE

- ACID:
  - Atomic, consistent, isolated, durable
- BASE
  - Basically available, soft state, eventual consistency
- What do these really mean?

BASE Universe

- Stale data OK
- Use Soft state
  - Non-durable and eventually consistent
- Approximate answers are OK
- Compare to ACID:
  - How would a database designer react to the above statements?
  - Would you want your DBMS to behave in a BASE manner?
Layered Software Model

- Service: Service-specific code
  - Workers that present human interface to what TACC modules do, including device-specific presentation
  - User interface to control the service

- TACC: Transformation, Aggregation, Caching, Customization
  - API for composition of sandbox data transformation and content aggregation modules
  - Uniform caching of original, post-aggregation and post-transformation data
  - Transparent access to customization database

- SNS: Scalable Network Service support
  - Incremental and absolute scalability
  - Worker load balancing and overflow management
  - Front-end availability, fault-tolerance mechanisms
  - System monitoring and logging

Two Services

- Transend web proxy service
  - Front end for devices, modems
- HotBot search engine

Scalability

<table>
<thead>
<tr>
<th>Requests/Second</th>
<th># Front Ends</th>
<th># Distillers</th>
<th>Element that saturated</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-24</td>
<td>1</td>
<td>1</td>
<td>distillers</td>
</tr>
<tr>
<td>25-47</td>
<td>1</td>
<td>2</td>
<td>distillers</td>
</tr>
<tr>
<td>48-72</td>
<td>1</td>
<td>3</td>
<td>distillers</td>
</tr>
<tr>
<td>73-87</td>
<td>1</td>
<td>4</td>
<td>FE Ethernet</td>
</tr>
<tr>
<td>88-91</td>
<td>2</td>
<td>4</td>
<td>distillers</td>
</tr>
<tr>
<td>92-112</td>
<td>2</td>
<td>5</td>
<td>distillers</td>
</tr>
<tr>
<td>113-135</td>
<td>2</td>
<td>6</td>
<td>distillers &amp; FE Ethernet</td>
</tr>
<tr>
<td>136-159</td>
<td>3</td>
<td>7</td>
<td>distillers</td>
</tr>
</tbody>
</table>

Scalability Study

- Does a demonstration of 7 distillers and 3 front demonstrate scalability?
- Pilot plant: type of demonstration
  - Build working scale model of the actual system
  - Develop ideas so that most real problems are uncovered

See *mythical man-month* by Fred Brooks
Summary

- Architecture panned out
- Software model a little fuzzy
  - Is CGI scripts + a DB the same thing?
- Philosophy of of BASE used
  - But not well understood what that means compared to ACID