Workload Characterization of Selected JEE-based Web 2.0 Applications

Priya Nagpurkar, William Horn, U Gopalakrishnan, Niteesh Dubey, Joefon Jann, Pratap Pattnaik

IBM TJ Watson Research Center

II SWC 2008
Introduction

• **Motivation**: Web 2.0 likely to be an important emerging workload

• **Goals**
  - Determine if Web 2.0 workloads differ from legacy workloads in the way they impact the systems that host them
  - Evaluate the efficacy of current systems in hosting these workloads, and implications for future systems

• **This paper**
  - Setting up applications that exploit Web 2.0 features as benchmarks, generating Web 2.0 workloads
  - Characterization on IBM’s Power5 architecture
Outline

• Introduction

• Web 2.0 Overview

• Benchmarks and Workloads

• Results

• Related Work

• Conclusion
Web 2.0 Tag Cloud

- Collaboration and Social Networking, Mashups, Rich Internet Applications, Media Sharing, underlying technologies
**Social Networking**

**Social Bookmarking**

**Rich Internet Application**

**Mashup**
Web 2.0 Overview

Networking advances

Client (sophisticated browser)

Inter-server sharing:
• server-side mashups
• workflows via SOA

Emerging Applications:
• social networking
• server-side productivity applications
• software-as-a-service

Heterogeneous stacks:
• LAMP
• J2EE/JEE
• Project Zero
• Ruby on Rails

Web 2.0 is data-centric:
• user-contributed content
• user application data
Benchmarks

This paper focuses on Java benchmarks and the J2EE stack:

- **Lotus Connections (BLOGS and DOGEAR):** IBM’s Enterprise Social Networking Software
- **Java PetStore 2.0:** Reference, open-source application for building AJAX-based RIA’s using JEE 5, developed at SUN

**Lotus Connections BLOGS**
- Uses Apache Roller blog engine
  - Struts: editor, Velocity: rendering
  - Hibernate: provides persistence
  - Apache Lucene: powers search

**Java PetStore 2.0**
- Exploits JEE 5 features like Java Persistence API (JPA) and Java Server Faces (JSF)
- AJAX components implemented using DOJO
- Apache Lucene: powers search
Workloads

• Transactions
  ■ User-level actions for each application
    ▸ viewBlog (blogs: total 9),
    ▸ createBookmark (dogear: total 14),
    ▸ selectTag (PetStore: total 9)
  ■ Can involve multiple html pages

• Workload mix
  ■ Defines proportion of transactions
  ■ Based on commonly observed usage patterns (in internal deployment of Lotus Connections)
Experimental Setup and Test Infrastructure

- Extended the open source *Grinder* framework to drive benchmarks and generate reports using data collected from different layers

<table>
<thead>
<tr>
<th>time (mins)</th>
<th>threads</th>
<th>test rate</th>
<th>runs completed</th>
<th>runs with HTTP failures</th>
<th>runs with content failures</th>
<th>A-runnable threads</th>
<th>A-user time</th>
<th>A-system time</th>
<th>A-wait time</th>
<th>A-app errors</th>
<th>A-CPI</th>
<th>A-JVM heap (KB)</th>
<th>D-wait time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>12</td>
<td>325.09</td>
<td>1737</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>65</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1.23</td>
<td>205403</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>22</td>
<td>502.24</td>
<td>2744</td>
<td>0</td>
<td>1</td>
<td>7</td>
<td>81</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1.72</td>
<td>185378</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>32</td>
<td>523.95</td>
<td>2809</td>
<td>0</td>
<td>1</td>
<td>12</td>
<td>87</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1.91</td>
<td>167303</td>
<td>0</td>
</tr>
<tr>
<td>40</td>
<td>42</td>
<td>548.16</td>
<td>2936</td>
<td>0</td>
<td>2</td>
<td>19</td>
<td>92</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1.99</td>
<td>147045</td>
<td>0</td>
</tr>
<tr>
<td>50</td>
<td>52</td>
<td>519.46</td>
<td>2740</td>
<td>0</td>
<td>2</td>
<td>26</td>
<td>92</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2.08</td>
<td>147045</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>62</td>
<td>531.66</td>
<td>2760</td>
<td>0</td>
<td>4</td>
<td>23</td>
<td>94</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>2.08</td>
<td>147045</td>
<td>0</td>
</tr>
</tbody>
</table>
Experimental Methodology

• Gradual ramp-up: virtual users are added gradually until throughput stabilizes
  ■ 15 minute warmup time

• Virtual users generate requests with zero think time
  ■ Single virtual user emulates multiple users
    ▸ Randomly picks user from 100k profiles
  ■ Request type and input data for request chosen randomly
    ▸ In keeping with workload mix definition

• Measurements on the Application Server (SUT)
  ▸ 30 minute measurement interval following warmup
  ▸ HPMs: 30 second samples

• Repeatability
  ■ Restore databases to undo writes
  ■ Data represents average across three runs
Throughput and CPU Utilization

- **System under test (AppServer) > 95% busy**
- **< 5% time spent in system level code**
Increased Chattiness and Data-centric Behavior

<table>
<thead>
<tr>
<th></th>
<th>Blogs</th>
<th>Dogear</th>
<th>PetStore</th>
<th>Trade6</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPS</td>
<td>9.00</td>
<td>19.8</td>
<td>18.39</td>
<td>645</td>
</tr>
<tr>
<td>GET Requests/txn</td>
<td>22.38</td>
<td>30.27</td>
<td>31.15</td>
<td>1.14</td>
</tr>
<tr>
<td>POST Requests/txn</td>
<td>0.22</td>
<td>0.31</td>
<td>0.19</td>
<td>0.06</td>
</tr>
<tr>
<td>AJAX Requests/txn</td>
<td>0.70</td>
<td>0.13</td>
<td>0.89</td>
<td>0</td>
</tr>
<tr>
<td>Bytes Sent/txn</td>
<td>99.80</td>
<td>50.84</td>
<td>570.35</td>
<td>1.5</td>
</tr>
<tr>
<td>Bytes Received/txn</td>
<td>83K</td>
<td>191K</td>
<td>467K</td>
<td>10K</td>
</tr>
</tbody>
</table>

- **Chattiness**
  - Presence of AJAX requests
  - More GET and POST requests per transaction

- **Data-centric Behavior**
  - More data from client to server (user generated content)
  - Frequent database accesses
• CPI for Web 2.0 applications lower than Trade6
• Instruction cache misses not as significant
• Significant cycles spent in data cache miss stalls
• Relatively more memory accesses for PetStore 2.0 increase data cache miss penalty
Related Work

• Workload characterization studies
  ■ Web 2.0
    ‣ Empirical Evaluation of a Collaborative Web Application [Stewart et al IISWC08]
  ■ Enterprise Java applications
    ‣ Memory system characterization [Barroso et al ISCA98]
    ‣ Architectural evaluation of TPC-W [Marden et al HPCA01]
    ‣ Performance studies of commercial systems on multi-cores [Tseng et al IISWC08]
    ‣ Characterizing a complex J2EE workload [Shuf and Steiner ISPASS07]
  ■ Java applications
    ‣ Characterizing the memory behavior of Java workloads [Shuf et al]
    ‣ Study of allocation behavior of SPECjvm98 [Dieckmann and Hoelzle ECOOP99]
• New Benchmarks and characterization methodology
  ‣ General methodology for characterizing commercial workloads [Kunkel et al 00]
  ‣ The DaCapo benchmark suite [Blackburn et al OOPSLA06]
Conclusion

- Characterization of three JEE applications that incorporate Web 2.0 features
  - User-level characteristics capture increased chattiness and data-centric behavior
  - Stalls from instruction cache misses not significant
  - Server-side behavior does not otherwise vary significantly or consistently from Trade6

- Future work
  - Varying workload mixes
  - Further analysis of data centric behavior
    - Database usage patterns, impact on JPA layer
  - Additional workloads
    - Scripting languages
THANK YOU!

QUESTIONS?