Stochastic model for QoS assessment in multi-tier web services

Ricardo M. Czekster, Paulo Fernandes, Afonso Sales, Thais Webber, Avelino Zorzo

Practical Applications of Stochastic Modeling (PASM’2011)

Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Paleoprospec Project - PUCRS/Petrobras, CAPES and CNPq - Brazil

March 17th, 2011
Challenge

- IT infrastructures are multi-tier
  - Widespread use is the three-tier model
- Software Testing is crucial
- Performance Testing web services
- Testing environments

Three-tier model

**Presentation Layer**
- Web Servers

**Application Layer**
- Application Servers

**Data Layer**
- DBMS

Testing Environment
Software Performance Engineering

Architecture for the System Under Test

Queueing Networks and Stochastic Automata Networks

Stochastic model for QoS assessment under SLAs

Preliminary results

Discussion
Software Performance Engineering (SPE)

Overview

- test software performance
- qualitative analysis
- functional testing
- non-functional testing
  - low priority

Details

- methodology → Test Plan
- instrumentation and performance counters
- derive workload intensity to apply
- bottleneck analysis (processor, memory, network, database)
- performance test applications in isolation
Service Level Agreements (SLAs)

Overview
- contracts between service providers and customers
- objective: ensure high Quality of Service (QoS)
- SLA: agreed-upon deadline for services

Service Level Agreement (SLA) range

global incoming transactions → Preprocessing Delay → Main Application Server → processed transactions

Czekster et al. Stochastic model for QoS assessment in multi-tier web services March 17th, 2011
System Under Test (SUT)

- Command Center
- Users create demands
- Client classes: enterprise, normal
- Operators take ownership and SLA begins

Customer

- large company with global presence
- project start: ‘simple’ scalability analysis
  - growing user demand and different client priorities
Globally Distributed Teams
Architecture overview

Service Level Agreement (SLA) range

Preprocessing Delay

global incoming transactions

processed transactions

Main Application Server

Windows 2003

PRF01
Application Server #1

Load Balancer

PRF02
Application Server #2

PRF03
Application Server #3

Windows 2003

RedHat Enterprise

DBMS01
Database Server #1

DBMS02
Database Server #2

Czekster et al. Stochastic model for QoS assessment in multi-tier web services March 17th, 2011
Research Question

Conditions
- objective: performance testing the SUT for analysis
- other simultaneous workloads are present
- bottleneck is the main application server
  - it could overload and start to underperform
- mechanisms to detect performance degradation due to external interference
  - external factors are causing SLAs to fail...

“How to detect excessive load from external applications and ensure that SLAs will be met?”
Stochastic Automata Networks (SAN)

- modular approach based on automata
- states, transitions and events
- events are local or synchronizing with rates
- rates are constant or functional

SAN

---

**SAN**

- modular approach based on automata
- states, transitions and events
- events are local or synchronizing with rates
- rates are constant or functional

---

Czekster et al.  
Stochastic model for QoS assessment in multi-tier web services  
March 17th, 2011
- SLA is set to 10s, where 0.02s is spent on preprocessing.
Phase II

- Isolated environment
- Queueing Networks → Worst Case Analysis → $T_{phaseII} = 8.8s$
Phase I

- Model for the bottleneck server (PRF01)
- Shared environment
- Stochastic Automata Networks → stop-run behavior

<table>
<thead>
<tr>
<th>Type</th>
<th>Event</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>loc</td>
<td>arrival</td>
<td>$\lambda$</td>
</tr>
<tr>
<td>loc</td>
<td>service</td>
<td>$f_s$</td>
</tr>
<tr>
<td>loc</td>
<td>start$_1$</td>
<td>$\alpha_1$</td>
</tr>
<tr>
<td>loc</td>
<td>stop$_1$</td>
<td>$\beta_1$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>loc</td>
<td>start$_P$</td>
<td>$\alpha_P$</td>
</tr>
<tr>
<td>loc</td>
<td>stop$_P$</td>
<td>$\beta_P$</td>
</tr>
</tbody>
</table>

Function

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_s$</td>
<td>$\text{service time} \times \frac{1}{1 + f_1 + ... + f_P}$</td>
</tr>
<tr>
<td>$f_1$</td>
<td>$(\text{state APP}_1 == \text{Run}) \times \tau_1$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$f_P$</td>
<td>$(\text{state APP}_P == \text{Run}) \times \tau_P$</td>
</tr>
</tbody>
</table>
Methodology

- LoadRunner for load testing
- Total of runs: at least four each time
- Duration: four hours, incrementally
- Different times of day

Analysis

- Performance counters on Windows machines
- `vmstat`, `iostat`, `netstat` on Linux machines
- Scripts to combine results for each execution
- Automatic reporting tool
Modeling

Estimations for applications under execution

- Application profiles
- Huge variations
- We chose five types of applications
  - e.g. operating system is more stable

Scenarios

- Varying external workload intensity
- Scenarios from 45% to 90% of utilization
- Study which type of application is preferable

<table>
<thead>
<tr>
<th>Running time</th>
<th>APP₁</th>
<th>APP₂</th>
<th>APP₃</th>
<th>APP₄</th>
<th>APP₅</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>18h</td>
<td>8h</td>
<td>20h</td>
<td>5h</td>
<td>23:50</td>
</tr>
</tbody>
</table>
## Running time estimates for each application

<table>
<thead>
<tr>
<th>Scenario</th>
<th>$\tau_1$</th>
<th>$\tau_2$</th>
<th>$\tau_3$</th>
<th>$\tau_4$</th>
<th>$\tau_5$</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17%</td>
<td>13%</td>
<td>6%</td>
<td>4%</td>
<td>5%</td>
<td>45%</td>
</tr>
<tr>
<td>2</td>
<td>4%</td>
<td>2%</td>
<td>33%</td>
<td>1%</td>
<td>5%</td>
<td>45%</td>
</tr>
<tr>
<td>3</td>
<td>9%</td>
<td>16%</td>
<td>6%</td>
<td>14%</td>
<td>5%</td>
<td>50%</td>
</tr>
<tr>
<td>4</td>
<td>11%</td>
<td>3%</td>
<td>27%</td>
<td>4%</td>
<td>5%</td>
<td>50%</td>
</tr>
<tr>
<td>5</td>
<td>3%</td>
<td>19%</td>
<td>1%</td>
<td>32%</td>
<td>5%</td>
<td>60%</td>
</tr>
<tr>
<td>6</td>
<td>19%</td>
<td>4%</td>
<td>21%</td>
<td>11%</td>
<td>5%</td>
<td>60%</td>
</tr>
<tr>
<td>7</td>
<td>2%</td>
<td>11%</td>
<td>3%</td>
<td>49%</td>
<td>5%</td>
<td>70%</td>
</tr>
<tr>
<td>8</td>
<td>22%</td>
<td>17%</td>
<td>14%</td>
<td>12%</td>
<td>5%</td>
<td>70%</td>
</tr>
<tr>
<td>9</td>
<td>3%</td>
<td>12%</td>
<td>8%</td>
<td>52%</td>
<td>5%</td>
<td>80%</td>
</tr>
<tr>
<td>10</td>
<td>33%</td>
<td>2%</td>
<td>37%</td>
<td>3%</td>
<td>5%</td>
<td>80%</td>
</tr>
<tr>
<td>11</td>
<td>6%</td>
<td>8%</td>
<td>4%</td>
<td>67%</td>
<td>5%</td>
<td>90%</td>
</tr>
<tr>
<td>12</td>
<td>36%</td>
<td>28%</td>
<td>17%</td>
<td>4%</td>
<td>5%</td>
<td>90%</td>
</tr>
</tbody>
</table>
Results

- \[ T = T_{\text{phaseI}} + T_{\text{phaseII}} + T_{\text{proc}} < 10\text{s} \rightarrow T_{\text{phaseI}} < 1.18\text{s} \]

<table>
<thead>
<tr>
<th>Scenario</th>
<th>( U ) (%)</th>
<th>( N ) (tr.)</th>
<th>( X ) (TPS)</th>
<th>( R ) (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>95.55</td>
<td>21.48</td>
<td>50.00</td>
<td>0.43</td>
</tr>
<tr>
<td>2</td>
<td>99.27</td>
<td>135.50</td>
<td>49.32</td>
<td>2.75</td>
</tr>
<tr>
<td>3</td>
<td>93.44</td>
<td>14.25</td>
<td>50.00</td>
<td>0.28</td>
</tr>
<tr>
<td>4</td>
<td>99.37</td>
<td>157.63</td>
<td>48.83</td>
<td>3.23</td>
</tr>
<tr>
<td>5</td>
<td>90.78</td>
<td>9.85</td>
<td>50.00</td>
<td>0.20</td>
</tr>
<tr>
<td>6</td>
<td>99.43</td>
<td>175.67</td>
<td>47.91</td>
<td>3.67</td>
</tr>
<tr>
<td>7</td>
<td>91.20</td>
<td>10.36</td>
<td>50.00</td>
<td>0.21</td>
</tr>
<tr>
<td>8</td>
<td>99.44</td>
<td>177.44</td>
<td>47.70</td>
<td>3.72</td>
</tr>
<tr>
<td>9</td>
<td>94.96</td>
<td>18.85</td>
<td>50.00</td>
<td>0.38</td>
</tr>
<tr>
<td>10</td>
<td>99.49</td>
<td>194.25</td>
<td>41.94</td>
<td>4.63</td>
</tr>
<tr>
<td>11</td>
<td>94.73</td>
<td>17.99</td>
<td>50.00</td>
<td>0.36</td>
</tr>
<tr>
<td>12</td>
<td>99.48</td>
<td>192.32</td>
<td>43.54</td>
<td>4.42</td>
</tr>
</tbody>
</table>
Results

Analysis
- depending on application profile, SLA can be respected
- useful for testing some applications in a shared environment
- possibility to estimate maximum level to guarantee SLAs

Discussion
- this testing environment replicates production conditions!
- applications with priorities
- phase-type modeling to represent timeouts
Summary

Overview
- problem is present in many organizations throughout the world
- it is very interesting and hard to instrument
- other authors already studied SLAs in more stable environments
- ongoing work...

Next
- more measurements and workload intensities
- general model refinements
- on-the-fly parameterization and solution
- measure system close to milestone deliverables
Thanks

Acknowledgments:
- Alberto Avritzer (Siemens Corporate Research)
- Performance team at FACIN/PUCRS

Ricardo M. Czekster
ricardo.czekster@pucrs.br