Performance Modelling of Mobile and Middleware Systems

Dr. John Murphy / Andrew Lee
Performance Engineering Laboratory
Dublin City University
Overview

Introduction

1. Mobile GPRS Simulator
2. Enterprise Components
3. Container Model
4. Enterprise Tests
5. Performance of Enterprise Systems - Compass

Conclusion
What is the Performance Engineering Lab?

- Joint research group between Dublin City University (Electronic Engineering) & University College Dublin (Computer Science)
  
- 6 Academic staff
- 23 Postgraduate researchers
- Performance-related research
  - Mobile and multimedia
  - Distributed components

- [http://www.perfenglab.com](http://www.perfenglab.com)
1. Mobile GPRS Simulator

- Time for a new simulator!
  - Old simulators were inflexible
  - Results were inconsistent
  - No channel model
  - Showed what sort of performance to look for
  - Needed a more integrated approach
1. **Mobile Simulator**

**Inputs and Outputs**

- **Simulator inputs**
  1. the size of the cell,
  2. number of available GPRS channels
  3. GSM traffic intensity (average number of channels)
  4. density of users
  5. their traffic and mobility characteristics

- **Simulator will give an approximation of**
  1. delay
  2. throughput
  3. losses

- **This is for GPRS at the RLC/MAC layer to a particular user**
1. Mobile Simulator

Initial data

Inputs:

1. Cell size
2. User population density
3. Traffic & Mobility characteristics
1. Mobile Simulator

Nine categories of users

Traffic:  
- B - background
- I - interactive (WWW)
- S - streaming

Mobility:  
- F - fast
- M - medium
- S - slow

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Mobility</th>
</tr>
</thead>
<tbody>
<tr>
<td>B - 10%</td>
<td>F - 10%</td>
</tr>
<tr>
<td>I - 10%</td>
<td>M - 10%</td>
</tr>
<tr>
<td>S - 10%</td>
<td>S - 10%</td>
</tr>
<tr>
<td>B - 10%</td>
<td>F - 10%</td>
</tr>
<tr>
<td>I - 10%</td>
<td>M - 10%</td>
</tr>
<tr>
<td>S - 10%</td>
<td>S - 10%</td>
</tr>
<tr>
<td>B - 20%</td>
<td>F - 10%</td>
</tr>
<tr>
<td>I - 10%</td>
<td>M - 10%</td>
</tr>
<tr>
<td>S - 10%</td>
<td>S - 10%</td>
</tr>
</tbody>
</table>

HyPerformix 2003 Performance Engineering Conference
May 19-20, 2003
1. Mobile Simulator

Input load is aggregated from all users

F/B=10%  M/B=10%  S/S=10%
F/I=10%  M/I=10%  S/I=10%
F/S=10%  M/S=10%  S/B=20%

Load

GPRS Simulator

HyPerformix 2003 Performance Engineering Conference
May 19-20, 2003
1. Mobile Simulator

Results aggregated by user type

GPRS Simulator

- F/B = 10%
- F/I = 10%
- F/S = 10%
- M/B = 10%
- M/I = 10%
- M/S = 10%
- S/I = 10%
- S/S = 10%
- S/B = 20%
1. Mobile Simulator

System Architecture

1. User [i]
2. User [i+1]
3. User [i+2]
4. User [i+3]

RLC

No Error
No Link Adaptation

MAC

GSM RF

HyPerformix 2003 Performance Engineering Conference
May 19-20, 2003
1. **Mobile Simulator**

## Functional Blocks

1. Fully scalable mobility, traffic and population model, several data sources implemented,

2. Statistically based Link Adaptation
   - Channel conditions fed back from section 3
   - CS1 and CS2 available

3. Several scheduling algorithms available, WRR, LQ, OQ...
   - Voice traffic inserted here as Poisson distributed function
   - Outputs throughput and delay

4. PDF of channel conditions
   - Markov model of median C/I
   - Changes for each user depending on speed
1. Mobile Simulator

Example

- 25 users
- 6 streaming users
- 6 browsing Internet users
- 13 background traffic users

- Case 1 = No voice traffic
  Mean number of GPRS timeslots = 8
- Case 2 = Poisson distributed voice traffic
  Mean number of GPRS timeslots = 3.5
1. Mobile Simulator

Test Setup

Delay prediction

Case 1

Case 2

- Three sets of Weighted Round Robin (WRR) weights are considered:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ws</td>
<td>8</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>Wi</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Wb</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
1. Mobile Simulator

Some Results

Average user delays, normalised delay (ND) and average user-perceived bit-rate (UPBR) for “Case 1” with three different configurations of weighting factors (a, b and c)

<table>
<thead>
<tr>
<th></th>
<th>Streaming</th>
<th>Interactive</th>
<th>Background</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay (ms)</td>
<td>ND</td>
<td>UPBR (kbps)</td>
</tr>
<tr>
<td>a</td>
<td>63.6</td>
<td>1</td>
<td>15.7</td>
</tr>
<tr>
<td>b</td>
<td>44.2</td>
<td>1</td>
<td>22.6</td>
</tr>
<tr>
<td>c</td>
<td>40.2</td>
<td>1</td>
<td>24.9</td>
</tr>
</tbody>
</table>

Average user delays, normalised delay (ND) and average user-perceived bit-rate (UPBR) for “Case 2” with three different configurations of weighting factors (a, b and c)

<table>
<thead>
<tr>
<th></th>
<th>Streaming</th>
<th>Interactive</th>
<th>Background</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay (ms)</td>
<td>ND</td>
<td>UPBR (kbps)</td>
</tr>
<tr>
<td>a</td>
<td>98.7</td>
<td>1</td>
<td>10.1</td>
</tr>
<tr>
<td>b</td>
<td>70.3</td>
<td>1</td>
<td>14.2</td>
</tr>
<tr>
<td>c</td>
<td>62.8</td>
<td>1</td>
<td>15.9</td>
</tr>
</tbody>
</table>
1. Mobile Simulator

Conclusion for Mobile Simulator

• Simple models
• Very flexible and fast
• Test different resource allocation algorithms
• QoS differentiation demonstrated
• Allows clear indication of the origin of phenomena
• Dimension cells under different scenarios
  • Cell size, user mobility and user density
  • Traffic and precedence
• Data exported to Excel for analysis
2. Enterprise Components

- A component is a lego™ like piece of code
- Allow developers to concentrate on business logic
- Decouple the logic from architecture
- Distributed Components offer:
  - Security
  - Robustness
    - Persistence
    - Transactional capability
  - Scalability
2. Enterprise Components

Components

Reservation  Payment  Payment

ORB
2. Enterprise Components

Technologies

• **CORBA** (Common Object Request Broker Architecture)
  Open standard created by OMG

• **.NET**
  Developed by Microsoft

• **EJB** (Enterprise JavaBeans)
  Developed by SUN
  J2EE technology de facto standard
  65-75% enterprises (Gig 2002)
2. Enterprise Components

J2EE (Java 2 Enterprise Edition)
SUN Microsystems’ definition of Enterprise JavaBeans is:

The Enterprise JavaBeans architecture is a component architecture for development and deployment of component-based distributed business applications.

Applications written using the EJB architecture are scalable, transactional, and multi-user secure.

The application may be written once, and deployed on any server platform that supports the Enterprise JavaBeans specification.
2. Enterprise Components

Distributed Objects

1. Invoke Method

2. Communicate Method Invocation

3. Invoke Method

4. Communicate Result

5. Return Result

Stub

Skeleton

Server Object
3. Container Model

- **Container:**
  - Environment that surrounds the bean
  - Provides primary services
    - Transaction
    - Security
    - Naming
    - Concurrency

- **Java Virtual Machine (JVM)**

![Diagram showing Container Model with EJB Server, EJB Container, and Beans]
3. Container Model

Model of EJB Container/Server
3. Container Model

Workbench model overview
3. Container Model

Parameters of Model

- **Workload:**
  - client request load (use cases ↔ transactions)
  - client request arrival rate
  - transaction types

- **Configurable parameters:**
  - JVM heap size
  - thread pool size (ORB)
  - number of bean instances for an application component
  - data base connection pool size (entity beans) (= thread pool size)
  - CPU times + memory consumption induced by middleware components (session stateless, session statefull, entity)
  - For each developed component method: CPU time (CPU cycles) + memory consumption (estimates)

- **Results:**
  - throughput (transaction / s)
  - average response time
  - resource utilization
3. Container Model

System Model

- JVM_memory
- Set_Usage
- Allocate_memory
- CPU
- Release_memory
3. Container Model

Container Model

[Diagram showing the process of allocating and releasing thread instances]
3. Container Model

Test results for stateless session bean
4. Enterprise Tests
Trade 3 Overview

6 entity beans, 2 session beans, and 2 message driven beans
4. Enterprise Tests

Trade 3 Configuration options

- EJB or direct mode (JDBC)

- Order processing immediately or asynchronous MDB

- Access mode web application → server side services: RMI, SOAP-RPC, Web Services (soon)

- Workload mix: Standard, High-Volume
4. Enterprise Tests

Throughput

![Graphs showing HTTP Responses/s and Active Virtual Users over time.](image-url)
4. Enterprise Tests

Monitored values

- % Disk utilization
- % processor time
- available memory

120-140MB
5. Performance of Enterprise Systems

Traditional Approach

Traditional Monitoring/Testing Tools
- Need Production like environments
  Only available at end of project
  Costly to construct & maintain
- Find Problems one at a time
  Multiple Fix-Test-Fix cycles….
- Testing Takes Time and Money

Experience Shows
- Time is Critical
- Mistakes are costly
5. Performance of Enterprise Systems

Newer Method

• Tool developed at the PEL – Compass

• Looks beyond the first performance problem
  InSight Prediction

• Predicts performance for different hardware
  ForeSight Prediction

• Expands the performance test envelope
  ClearSight Prediction

• http://www.ejbperformance.org
InSight Prediction

Helps find all the Performance Problems at once

Synchronous Messaging
Too Many Method Calls
Single Threaded Code
In-efficient Code
DB Indexes
5. Performance of Enterprise Systems

Foresight Prediction

Find Performance Problems using less/different hardware

Content Providers

Fire Wall

Web Servers

Application Servers

Legacy System

DB Servers

DB Servers
Performance Testing Takes Time…

From this

To this

leaving performance problems undetected.
5. Performance of Enterprise Systems

Component Optimization

• Intelligent load balancing
  Switches load based on current and historic performance

• Self adaptive components
  Same functional components different performance curves
  Detecting and switching when needed

• Container Optimization
  Merging components for better performance
Conclusion

• **Mobile Multimedia**
  New Simulator developed around Workbench

• **Component Technology**
  Allows developers to concentrate on business logic
  Bring technology specific performance problems

• **PEL Research**
  COMPAS Performance Prediction
  [http://www.ejbperformance.org](http://www.ejbperformance.org)
  Component optimization