An Architectural Approach for Improving Availability in Web Services

Evangelos Parchas and Rogério de Lemos
Computing Laboratory - University of Kent at Canterbury, UK

- Motivation
- Architectural pattern
- Implementation and results
- Concluding and future work
Motivation

- Dependable systems that are built from existing components/systems/services;
  - existing systems cannot be trusted, they are not under the control of the architect;
  - the system has to be protected against faults;
- A good architecture improves the handling of faults:
  - error confinement, and reduction of system complexity;
- Application of traditional fault tolerant techniques:
  - Components – self-checking and comparison for error detection and confinement;
  - System – system dynamic reconfiguration for fault handling;
**Motivation**

- Not quite ‘Web services’:
  - HTML and text, but we are getting there;
Architectural Solution

For improving the *reliability* and *availability* of systems by using multiple sources of information:

- Components implement crash-failure semantics;
  - plus timing self-checks;
- System dynamic reconfiguration;
Architectural Solution

Failure assumptions:
- Bridge: arbitrary failures;
- Comparator: crash-failsures;
  - \( n \) failures for \( n+1 \) Bridges;
- Manager: does not fail;
Architectural Solution

Manager -> Comparator
Comparator -> Bridge_1
Comparator -> Bridge_2
Bridge_1 -> iOutput
Bridge_2 -> iOutput
Architectural Solution
Architectural Solution
NSCP is a design diverse technique that uses redundancies to check its own behaviour during execution.
Case Study: Stock Quotes

Implemented in Jini: set of APIs and network protocols that help in building and deploying distributed systems;
# Implementation Analysis

## System statistics:
- **total restoration time (ms)**

<table>
<thead>
<tr>
<th>Process</th>
<th>Average Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure detection</td>
<td>80</td>
</tr>
<tr>
<td>Reconfiguration</td>
<td>2</td>
</tr>
<tr>
<td>Stream restoration</td>
<td>123</td>
</tr>
<tr>
<td>Total restoration</td>
<td>205</td>
</tr>
</tbody>
</table>

## Reliability and availability of the services and final system

<table>
<thead>
<tr>
<th>Service</th>
<th>Reliability (failure rate)</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yahoo</td>
<td>4.2 f/p</td>
<td>0.999</td>
</tr>
<tr>
<td>Lycos</td>
<td>81.3 f/p</td>
<td>0.983</td>
</tr>
<tr>
<td>Quicken</td>
<td>8.4 f/p</td>
<td>0.998</td>
</tr>
<tr>
<td>MSN</td>
<td>6.9 f/p</td>
<td>0.998</td>
</tr>
<tr>
<td>System Service</td>
<td>0.6 f/p</td>
<td>0.999</td>
</tr>
</tbody>
</table>
Concluding Remarks

◆ Architectural solution for improving the reliability and availability of Web services:
  ◆ it might be naïve but it does the job;
    ◆ its feasibility was shown in the context of stock quote information;

◆ Current and future work:
  ◆ apply the architectural solution in the context of Web services;
  ◆ special attention will be given to the Bridge component for the purpose of fault tolerance:
    ◆ addition of the non-functional interface;
    ◆ automatic generation of an implementation;