Reflection, Self-Awareness and Self-Healing in OpenORB

Gordon Blair, Geoff Coulson, Lynne Blair, Hector Duran-Limon, Paul Grace, Rui Moreira and Nikos Parlavanetzas

Distributed Multimedia Research Group, Computing Department, Lancaster University, UK
Introduction

• What does self-healing mean to you?
  – Potential to mask failure, overcome environmental changes, manage changing user needs …
  – Specifically overcoming these in middleware

• Properties of a self-healing middleware?
  – *Openness* i.e. Access to underlying infrastructure
  – Ability to *reconfigure* structure at run-time
  – Maintain *integrity* of a running system

• Self-healing Middleware (OpenORB)
  – Reflective middleware that can support self-healing systems
Why Reflection?

- Support for introspection
  - The ability to inspect the structure and behaviour of the system
    - e.g. dynamic monitoring or accounting
- Support for adaptation
  - Short term dynamic re-configuration
    - e.g. changing protocol configuration
  - Longer term evolution
    - e.g. adding new multimedia service
The Open ORB Architecture: A Marriage of Three Technologies

- **Components**
  - Apply component-oriented programming at base and meta levels

- **Reflection**
  - Use reflection to access structure and behaviour of the underlying middleware platform
  - Four meta-models (Interface, Architecture, Interception and Resource)

- **Component Frameworks**
  - Domain-specific ‘life-support environments’ for plug-in components
Application Areas

- **Multimedia** - adaptive stream bindings
- **Mobile Computing** – dynamic protocol configuration to overcome heterogeneity and environment’s limited resources
- **The NETKIT Project** - application of Open ORB principles to programmable networks
- **Other areas**
  - Grid computing
  - Co-operative scientific visualisation
  - Distributed virtual environments
  - Digital libraries
Self-Adaptation in OpenORB

- Inject components for monitoring and adaptation

<table>
<thead>
<tr>
<th>Monitoring</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Event Collector</strong></td>
<td><strong>Strategy Selectors</strong></td>
</tr>
<tr>
<td>Observe behaviour of underlying functional components and generate relevant QoS events.</td>
<td>Select an appropriate adaptation strategy (i.e. strategy activator) based on feedback from monitors.</td>
</tr>
<tr>
<td><strong>Monitor</strong></td>
<td><strong>Strategy Activators</strong></td>
</tr>
<tr>
<td>Collect QoS events and report abnormal behaviour to interested parties.</td>
<td>Implement a particular strategy, e.g. by manipulating component graph.</td>
</tr>
</tbody>
</table>
Other Issues

• Self-Healing algorithms
  • Less reliance on static strategy selection
  • Larger scale self-healing systems
    – Ideas concentrate on middleware platforms