Focus Group: Resource Management and Optimizations

Michael Kircher, Prashant Jain

Michael.Kircher@siemens.com, pjain@gmx.net

Corporate Technology, Siemens AG
Munich, Germany

Proceedings of the Focus Group on Patterns for Resource Management and Optimization
EuroPLoP 2002, July 3-7, Kloster Irsee, Germany

Participants

Michael Kircher, Prashant Jain, Markus Völter, Peter Sommerlad, Dietmar Schütz, Ansgar Radermacher, Reinhard Maier, Titos Saridakis, Amir Raveh

Workshop goals

• Get Feedback on Resource Management
• Collect views about Resource Management
• Resource Management versus Performance, Reliability and Security
• Definition of a "Resource"
• Definition of "Resource Management"
• Categorization of Resources
• What are common resources?

Example Resources

• **Not reusable**—Sheet of Paper in the Printer, Piece of Cake, CPU cycles
• **Sharable by time-slice**—I/O Ports, Book in a Library, User Focus of a GUI, Printer, Chair, Keyboard during Pair Programming, Passage through a narrow corridor, TLS-key
• **Sharable by space-partitioning**—Network Switch, DSP, Bandwidth, Multi-user Computer System, Floor Space, Screen Space, Memory
• **Sharable, but no time-slice or space-partitioning**—Read-only Objects, TV-Set
• **Not covered**—Monitor, Process, Thread, Mutex, Transactions (because they are synchronization primitives themselves)
• **Unclear**—Queue, Socket (because it is unclear how they can be categorized)

• **Unknown** (to be categorized)—Objects, Service, Interrupt Channel, Remote Objects, Component Instances, I/O Channel

**Definition: What is a Resource?**

Entity that is available only in a limited supply

• There exists a requestor that needs the entity to perform a function

• There exists a mechanism that provides the entity on request

**Definition: What is Resource Management?**

• It is a strategy to supply resources to requestors.

• Requestor and Resource Manager conform to a contract that is compliant to the strategy.

• Differentiate between the illusion of unlimited resources and making the requestor take full responsibility of request, release, and synchronization of resources.

**Challenges in Resource Management**

• Acquisition

• LifeCycle

• Release

• Coordination

• Access

• Tracking/Monitor/Profile

• Conflict Resolution and Prevention

**Resource Properties & their implications**

• not reusable => not sharable (time & space)

• can be partitioned => sharable (space)

• can get synchronized => sharable (time)
• no exclusive access => sharable

**Derived Resource Management implications**

• reusability influences release strategy
• when you need to share, first partition, then time-slice (& synchronize)
• resource usage influences how it is to be managed

**Non-functional Requirements of Resource Management**

• Performance
• Security
• Reliability
• Simplicity
• Scalability
• Flexibility (Strategy, Configuration)
• Robustness
• Availability (time between request and actual acquisition)
• Usability (ease of use for the requestor)

**Conflicts**

• Performance vs. Security,
• Performance vs. Scalability,
• Security vs. Availability,
• Security vs. Usability,
• Flexibility vs. Robustness,
• Reliability vs. Availability
Resource Types

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3a</th>
<th>3b</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑</td>
<td></td>
<td>→</td>
<td>↓</td>
<td>↑ Performance</td>
</tr>
<tr>
<td>↑</td>
<td></td>
<td>↓</td>
<td></td>
<td>↑ Security</td>
</tr>
<tr>
<td>↓</td>
<td></td>
<td>↑</td>
<td></td>
<td>↑ Reliability</td>
</tr>
<tr>
<td>↑</td>
<td></td>
<td>→</td>
<td>↓</td>
<td>↑ Simplicity</td>
</tr>
<tr>
<td>→</td>
<td></td>
<td>↓</td>
<td></td>
<td>↑ Scalability</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>↑ Flexibility</td>
</tr>
<tr>
<td>↑</td>
<td></td>
<td>→</td>
<td>↓</td>
<td>↑ Robustness</td>
</tr>
<tr>
<td>↓</td>
<td></td>
<td>→</td>
<td>↓</td>
<td>↑ Availability</td>
</tr>
<tr>
<td>↑</td>
<td></td>
<td>→</td>
<td>↓</td>
<td>↑ Useability</td>
</tr>
</tbody>
</table>

Resource Categories

- Not reusable: acquire access, no synchronization on access, can be acquired only once, synchronization on acquisition (e.g. piece of cake)
- Partitioning: partition synchronously before allowing acquisition, no synchronization on access, support release (e.g. floor space)
- Time slicing:
  a) Acquire for exclusive access, release (e.g. shopping cart)
  b) Acquire as one of many users, synchronize access, release (e.g.)
- Non-exclusive access: nothing to do, but eventually acquire it and access it, no synchronization required, multiple users possible (e.g. cinema screen)