SOA Patterns: New Insights or Recycled Knowledge?

Gregor Hohpe
Software Engineer
www.eaipatterns.com

Could It Be So Easy?

- Buzzword compliant, but not a service-oriented architecture
- Synchronous call stack mentality
- No interface-implementation separation
SOA on Architect's Napkin

SOA on Developer's Napkin

“Architect’s Dream”

“Developer's Nightmare”
Service-oriented architectures involve a variety of new programming models.

We can expect to see a renaissance of design patterns that capture our experience in using them correctly.

Understanding Technology

• Syntax
  • Basic language mechanism
  • Artefact of crude input devices

• Constructs
  • "Vocabulary": Objects, Classes, Interfaces, Inheritance
  • Easily explained but no guidance on good design

• Principles
  • Separation of Concerns, Open-Closed, etc.
  • Help evaluate a solution

• Patterns
  • Guidance
BED ALCOVE

Design problem
Bedrooms make no sense.

Forces
First, the bed in a bedroom creates awkward spaces around it: dressing, working, watching television, sitting, are all rather foreign to the side spaces left over around a bed. Second, the bed itself seems more comfortable in a space that is adjusted to it.

Solution
Don’t put single beds in empty rooms called bedrooms, but instead put individual bed alcoves off rooms with other nonsleeping functions, so the bed itself becomes a tiny private haven.
Design Patterns in Software Development

- Substitute for industry standard vocabulary
- "Soft" around the edges
- Guide in the absence of strict rules
- Subject to hype cycle: "pattern" = "credible"

10 Years After GoF -- Why Revisit Patterns?

- New programming models bring new patterns.
- Patterns are expressed using the constructs of the underlying architectural style (e.g. OO, SOA).
- Good to document our knowledge when not all kinks are worked out yet.
- Patterns help discover higher levels of abstraction.
- Ultimately, some patterns can be implemented in the platform.
New Programming Models in SOA

- Event-based, Asynchronous Programming
  - Explicit state management
  - Sequencing, timing uncertainty
- Declarative Programming
  - Execution path chosen at run-time
  - XSLT, Rules engines
- Object-Document Mapping
  - Analogous to O-R mapping: subtle, but important
- Process Modeling
  - Many concurrent, long-running instances
  - No two-phase-commit style transactions
- Protocol Design

“Doodleware” Only Limited Help

- Example
  - Graphical process editors
  - Graphical transformation editors
- We love pictures
- Programming in pictures tedious
  - Scalability issues
  - Diff, Merge mostly unsupported
- Often a thin veneer over a complex (or unfamiliar) programming paradigm
New Patterns in SOA

- Asynchronous Messaging Patterns
- Conversation Patterns
- Orchestration Patterns
- Process / Workflow Patterns
- Endpoint Patterns
- Security Patterns
- Architectural Patterns
- ...

Async. Messaging Patterns

- Components communicate via message exchanges
- Simplified interaction model
- Compose solutions out of routing, transformation elements
- www.eaipatterns.com
  - 65 patterns
  - Java, C#, Axis, JMS, MSMQ, TIBCO, BizTalk
Conversations

- A process supports the desired exchange of messages through send and receive activities
- Each conversation corresponds to one process instance
- Each participant has a (potentially different) process definition
Conversation Patterns

- **Discovery**
  - Dynamic Discovery
  - Consult Directory
  - Referral

- **Establishing a Conversation**
  - Three-way Handshake
  - Acquire Token First
  - Rotate Tokens
  - Address Verification

- **Basic Conversations**
  - Fire-and-Forget
  - Request-Response
  - Request-Response with Retry
  - Polling
  - Subscribe-Notify
  - Quick Acknowledgment

- **Multi-Party Conversations**
  - Proxy
  - Contingent Requests
  - Reaching Agreement
  - Leader Election

- **Reaching Agreement**
  - Receiver Cancels
  - Sender Cancels
  - Binding Request
  - Binding Offer

- **Resource Management**
  - Lease
  - Renewal Reminder

- **Error Handling**
  - Do Nothing
  - Retry
  - Compensating Action

Process Patterns

- Long-running processes
- Multiple instances of a process template
- Orchestration ≈ process across multiple services exposed as composite service
- WS-BPEL (Business Process Execution Language)

**Orchestration** describes the automated arrangement, coordination, and management of complex computer systems, middleware, and services. --WikiPedia
Workflow Patterns (Subset)

- **Exclusive Choice (XOR)**
  - Choose one execution path from many alternatives

- **Multiple Choice (OR)**
  - Choose several execution paths from many alternatives

Workflow Patterns (Subset)

- **Synchronizing Merge**
  - Merge many execution paths. Synchronize if multiple paths are taken.

- **Multiple Merge**
  - Merge many execution paths without synchronizing.

- **Discriminator**
  - Merge many execution paths without synchronizing. Execute the subsequent activity only once.
Seems Trivial But…

Support by standards can be spotty.

<table>
<thead>
<tr>
<th></th>
<th>XPDL</th>
<th>UML</th>
<th>BPEL</th>
<th>XLANG</th>
<th>WSFL</th>
<th>BPML</th>
<th>WSCI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel Split</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Synchronization</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Exclusive Choice</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Simple Merge</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Multi Choice</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Synchronizing Merge</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Multi Merge</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+/-</td>
<td>+/-</td>
</tr>
<tr>
<td>Discriminator</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Deferred Choice</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Source: www.workflowpatterns.com

Take-Away

• New architectural styles bring new patterns.
• Use pattern languages as design check list.
• Use pattern languages as catalog of options.
• Patterns often highlight subtleties that vendors like to glance over.
• Keep us from reinventing the wheel.
• Patterns harvested from actual usage.
• We are likely to experience a pattern renaissance.
• Share your knowledge.
### SOA Pattern Resources: Books

<table>
<thead>
<tr>
<th>Title</th>
<th>Author(s)</th>
<th>Publisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise Integration Patterns</td>
<td>Gregor Hohpe, Bobby Woolf</td>
<td>Addison-Wesley, 2004</td>
</tr>
<tr>
<td>Integration Patterns</td>
<td>Microsoft Patterns &amp; Practices</td>
<td></td>
</tr>
<tr>
<td>Essential Business Process Modeling</td>
<td>Havey</td>
<td>O'Reilly, 2005</td>
</tr>
<tr>
<td>Pattern-Oriented Software Architecture Vol.2</td>
<td>Schmidt et al</td>
<td>Wiley, 2000</td>
</tr>
<tr>
<td>Pattern-Oriented Software Architecture Vol.4</td>
<td>Buschmann et al</td>
<td>Wiley, 2007</td>
</tr>
<tr>
<td>SOA Patterns</td>
<td>Arnon Rotem-gal-oz</td>
<td>Manning, 2008 (?)</td>
</tr>
</tbody>
</table>

### SOA Pattern Resources: On-line

<table>
<thead>
<tr>
<th>Pattern Type</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration Patterns</td>
<td>msdn.microsoft.com/practices/Topics/patterns</td>
</tr>
<tr>
<td>Messaging Patterns</td>
<td>eaipatterns.com</td>
</tr>
<tr>
<td>Remoting Patterns</td>
<td><a href="http://www.voelter.de/publications/books-rem.html">www.voelter.de/publications/books-rem.html</a></td>
</tr>
<tr>
<td>Orchestration Patterns</td>
<td>orchestrationpatterns.com</td>
</tr>
<tr>
<td>SOA Patterns</td>
<td><a href="http://www.rgoarchitects.com/SOAPatterns/">www.rgoarchitects.com/SOAPatterns/</a></td>
</tr>
<tr>
<td>Workflow Patterns</td>
<td>workflowpatterns.com</td>
</tr>
<tr>
<td>Conversation Patterns</td>
<td>conversationpatterns.com (WIP)</td>
</tr>
<tr>
<td>Service Interaction Patterns</td>
<td>serviceinteraction.com</td>
</tr>
<tr>
<td>Data Integration Patterns</td>
<td>msdn.microsoft.com/practices/Topics/patterns</td>
</tr>
<tr>
<td>Web Service Security Patterns</td>
<td>msdn.microsoft.com/practices/Topics/patterns</td>
</tr>
</tbody>
</table>