

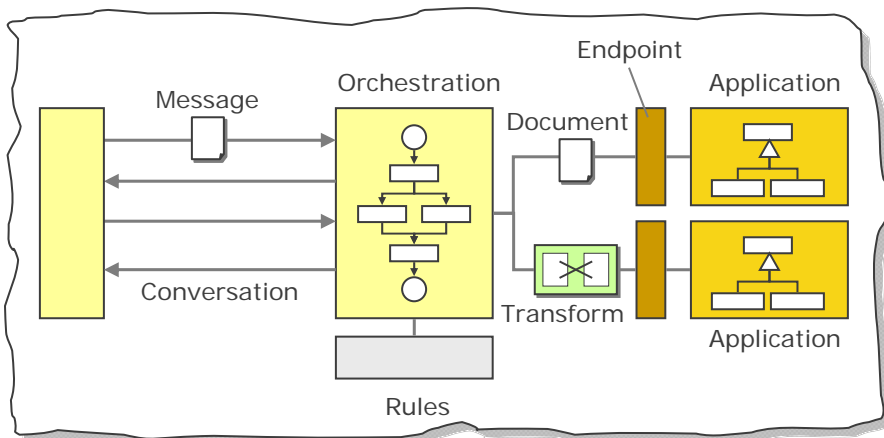
# Software Service Engineering: Architect's Dream or Developer's Nightmare?

Gregor Hohpe  
Software Engineer  
www.eaipatterns.com



© 2008 Google, Inc. All rights reserved.

## SOA on Architect's Napkin

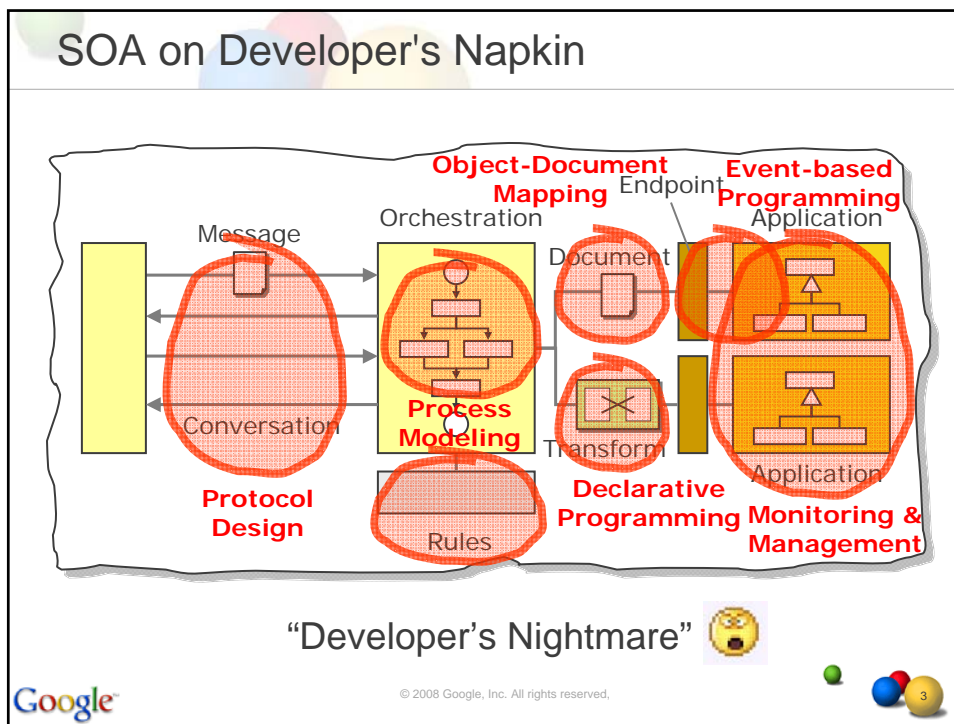


“Architect’s Dream” 😊



© 2008 Google, Inc. All rights reserved.





Service-oriented architectures involve a variety of new programming models.

We need to capture design experience in using them correctly.

Patterns are the best suited for this purpose.

© 2008 Google, Inc. All rights reserved.

## 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



© 2008 Google, Inc. All rights reserved.



Looser Coupling → Fewer Constraints

Fewer Constraints → Fewer Assumptions

Fewer Assumptions → More Uncertainty

More Uncertainty → More Failure Scenarios

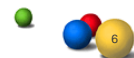
→ More Mistakes

→ More Testing

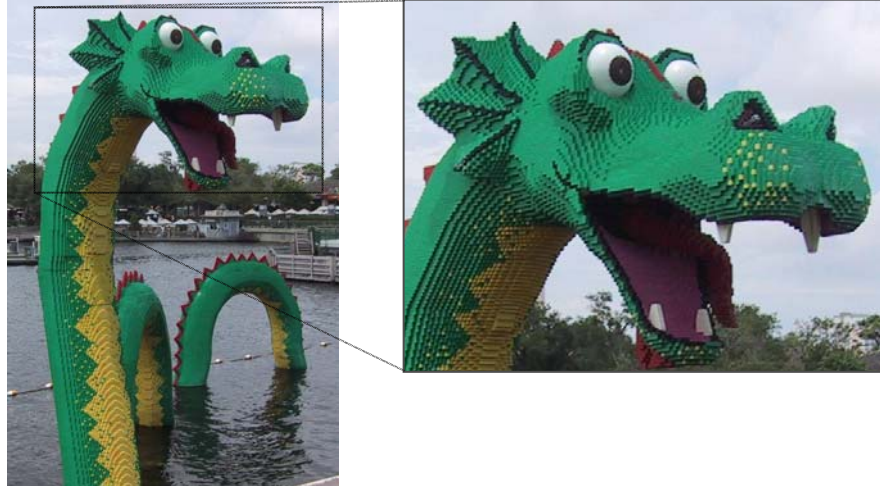
→ More ...



© 2008 Google, Inc. All rights reserved.



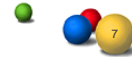
## Composability



"The ability to build new things from existing pieces."



© 2008 Google, Inc. All rights reserved.



## Composition Considerations

- Introduces a new layer into the system: the composition layer
- This layer needs to be well defined and tested

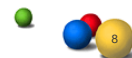
*"Great composers are few and far in between."*

-- Gregor's Ramblings

- Independent variability can lead to surprises
- Need to extract accurate state of the system
  - Design-time analysis
  - Run-time observation



© 2008 Google, Inc. All rights reserved.

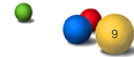


## Deferring Decisions until Run-time

- Loosely coupled systems enable independent variability
- Not all parts are under central control
- Build a system that can evolve locally without global impact
- Critical to extract accurate state of the system:
  - Design-time
  - Run-time
- “Reverse” MDA



© 2008 Google, Inc. All rights reserved.

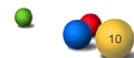


## Tooling Support Still Weak

- Copy-Paste
- Diff
- Refactoring
- Debugging
- Version Control Integration
- Rapid test – fix – refactor cycles
- Test-driven development



© 2008 Google, Inc. All rights reserved.



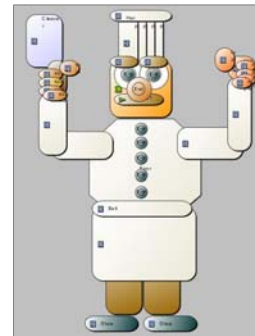
## Extra Slides



© 2008 Google, Inc. All rights reserved.

### “Doodleware” Only Limited Help

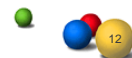
- Example
  - Graphical process editors
  - Graphical transformation editors
- We love pictures
- Programming in pictures tedious
  - Scalability issues
  - Diff, Merge mostly unsupported
- Often a thin veneer over a complex (or unfamiliar) programming paradigm



“EAI Art”



© 2008 Google, Inc. All rights reserved.

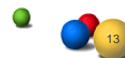


## Design Patterns

- “Mind sized” chunk of information (Ward Cunningham)
- Human-to-human communication
- Expresses intent (the “why” vs. the “how”)
- Observed from actual experience
- Not a firm rule
- Can be soft around the edges -- not copy-paste code



© 2008 Google, Inc. All rights reserved.



## Patterns Origin – C. Alexander

### BED ALCOVE

#### Design problem

Bedrooms make no sense.

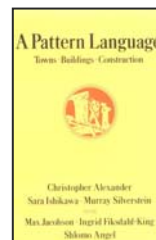
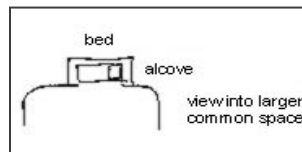
#### Forces

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.

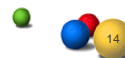
Second, the bed itself seems more comfortable in a space that is adjusted to it.

#### 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.



© 2008 Google, Inc. All rights reserved.

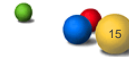


## "New" Patterns in SOA

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



© 2008 Google, Inc. All rights reserved.

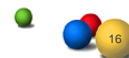


## Design Patterns in Service Engineering

- New programming models bring new patterns.
- Substitute for industry standard vocabulary
- Guide in the absence of strict rules
- Subject to hype cycle: "pattern" = "credible"
- Patterns help discover higher levels of abstraction.



© 2008 Google, Inc. All rights reserved.



## New Patterns in SOA

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



© 2008 Google, Inc. All rights reserved.

