Applying Pattern Concepts to Systems (Enterprise) Architecture

Steve Mazza

Naval Postgraduate School
Monterey, CA

SE4920, Summer/2014

Introduction

Intent

"...to describe the research being performed to investigate the application of patterns to the practice of systems architecting."

Goal of the enterprise architect: develop and implement a complex system using a methodical and repeatable approach.

Definition

"A pattern is a model or facsimile of an actual thing or action, which provides some degree of representation (an abstraction) to enable the recreation of that entity over and over again."

Frameworks

Considerations:

- strategic business goals
- business rules
- existing and legacy systems
- applicable technologies

Frameworks (contain many similarities)

- Zachman
- EA^{3}Cube
- DoDAF

A Short History Of Patterns

Christopher Alexander (Architect) pioneered the use and documentation of patterns. He was the first to recognize their value.
Documentation of patterns consists of:

- Name: should be descriptive and represent the proposed solution
- Context: addresses the setting for the problem
- Problem: describes what the pattern addresses (the challenge or solution)
- Solution: describes the application of the pattern

His establishment of patterns tried to lower the cognitive burden of design "by exploring large design spaces on behalf of the architect." Making useful generalizations about reusable elements helps reduce the difficulty in thinking about architecting solutions.

History

Patterns In Information Technology

IBM established the following steps in using patterns to address eBusiness:

1. Develop a high-level business description
2. Develop a solution overview diagram
3. Identify business patterns
4. Identify integration patterns
5. Identify composite patterns
6. Identify application patterns
7. Integrate a package into a solution
8. Identify run-time patterns
9. Identify run-time and product mappings

History

Patterns In Software Development

- Introduced by the Gang Of Four (Gamma, Helm, Johnson, and Vlissides
- Considerably furthered by Martin Fowler
- Establish common vocabulary
- Reduce costly late cycle changes
- Weave parts of the overall software system architecture into a whole

Documenting Software Patterns

Fowler (2003)

"When people write patterns, they typically write them in some standardized format – as befits a reference. However, there's no agreement as to what sections are needed because every author has his or her own ideas..."

Discovering Patterns

Patterns are usually discovered through our brain's natural tendency to create abstractions on complexity.
Abstractions tend to suggest more general solutions which, if sufficiently developed and documented, can lead to more broadly applicable solutions.

Capturing Implicit Knowledge With Patterns

- Implicit knowledge is only useful if it is shared usefully (in a manner that allows its application).
- Patterns offer a formal method to capture key aspects of that knowledge and facilitate its transfer and reuse.
- Patterns are only useful if they can be used by others.
- Formal means of documentation ensure accurate transfer of implicit knowledge.

Potential Benefits Of Architectural Patterns

Benefits include:

- Improved communications with stakeholders and design teams
- Application of sound architectural concepts and implementations.
- Reuse
- Reduced effort with regard to system testing, integration, and maintenance
- Improved development efficiency and productivity
- Reduced cost of documentation (example)
- Control of complexity through use of well known patterns

Architectural Pattern Research

Two issues applying patterns to systems architecture

- Architecture of a system requires a higher level of abstraction than that found in the software that may be part of the system.
- Patterns need to address interfaces to non-software parts in the pattern description.

Documenting Architecture Patterns

Minimal sections (just like Alexander):

- Name
- Context
- Problem
- Solution

Other useful sections:

Aliases	Resulting context	Known issues
Email	Keywords	Forces
Related patterns	Sketch	Rationale
Example	Interfaces	Date documented
Authors	References	

A Proposed Pattern Hierarchy

System architecture patterns are broken into:

1. Structural patterns: physical part of the architecture
2. System requirements patterns: format of a properly formed requirement
3. Systems engineering activities patterns: indicate how the process of engineering activity is performed
4. Systems engineering roles patterns: describe how the engineering role is performed
5. Systems process patterns

When Not To Use Architecture Patterns

Downfalls:

- Structural constraints inherent in patterns limits creativity
- Experts within their domain see patterns as having little use

When to avoid using patterns:

- New or unique requirements preclude the existence of a pattern
- Looking for a unique solution
- Technological innovation out-paces developed patterns

"For designs that are proven and effective, and addressing problems common across multiple systems an domains, there should be a strong motivation to leverage the benefits that can accrue from the application of patterns."

Summary

- Patterns are models or abstractions of reality
- Complexity exceeds mental capacity
- Patterns can exist at multiple levels
- Patterns are a powerful part of the architect's toolbox
- Patterns help solve difficult problems by leveraging existing knowledge
- Patterns help minimize the possibility that details will fall through the cracks
- Patterns expedite knowledge transfer