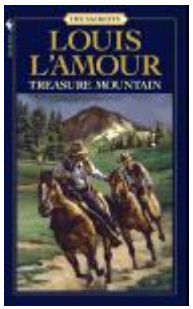
Patterns and Systems Architecture

Feb 18, 2010

Dr. Robert Cloutier
Associate Professor

© Robert Cloutier, 2009

“The human eye has a readiness for patterns. Much is not seen simply because the mind is blind, not eyes. The eyes see in lines, curves, and patterns. Man himself works in patterns simple or complex, and such things are often evidence of man’s previous presence.”

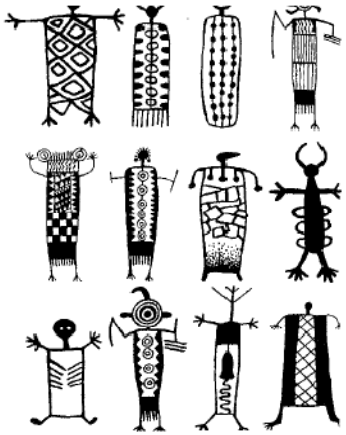


William Tell Sackett, Treasure Mountain
By Louis L’Amour, 1972

© Robert Cloutier, 2009

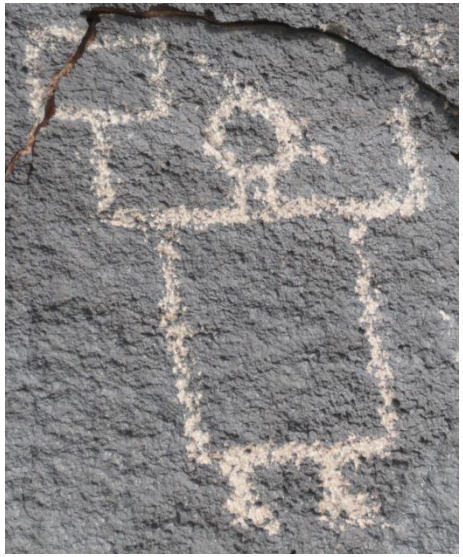
Patterns in Prehistoric Art, Society, and Communications

Patterned Body Anthropomorphs



Petroglyphs, Coso Range, California

Source: A Field Guide to Rock Art Symbols Of the Greater Southwest, Alex Patterson, Johnson Books, Boulder, 1992



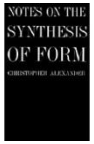
Rock Art at Petroglyph National Monument, Albuquerque, NM

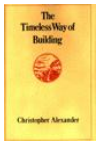
Petroglyphs are powerful cultural symbols that reflect the complex societies and religions of the surrounding tribes.

© Robert Cloutier, 2009

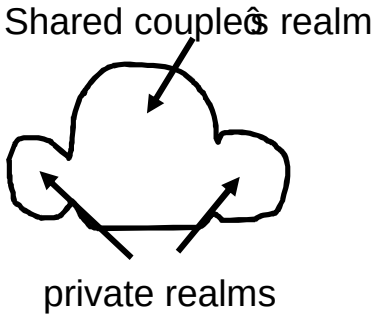
Brief Pattern History

- 1964 ï Christopher Alexander
  - ï Books on Architecture, building and urban planning
  - ï Notes on the Synthesis of Form
  - ï A Pattern Language
  - ï A Timeless Way of Building

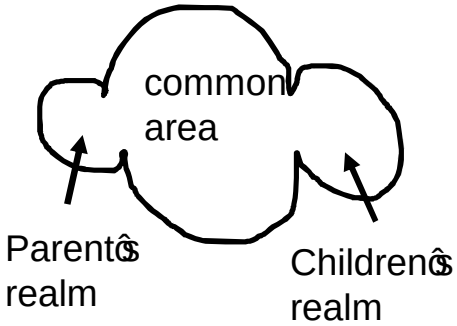




- 1987 - Ward Cunningham & Kent Beck
  - ï Decided to use some of Alexander's ideas
  - ï Developed five patterns for guiding novice Smalltalk programmers
  - ï Presented paper at OOPSLA'87 in Orlando
    - *"Using Pattern Languages for Object-Oriented Programs"*.
- 1995 - Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
  - ï Software Design
  - ï Design Patterns: Elements of Reusable Object-Oriented Software



House for Couple



House for Couple
w/ Small Children

© Robert Cloutier, 2009

Abstraction – Why Patterns Work

Photograph of a White Rose

Abstraction White Rose
Georgia O'Keefe, 1927

- Abstraction hides detail
- Does not obfuscate the item of interest

I know I can not paint a flower on the desert on a bright summer morning but maybe in terms of paint color I can convey to you my experience of the flower or the experience that makes the flower of significance to me at that particular time.

Georgia O'Keefe, 1931

© Robert Cloutier, 2009

Implicit & Explicit Knowledge

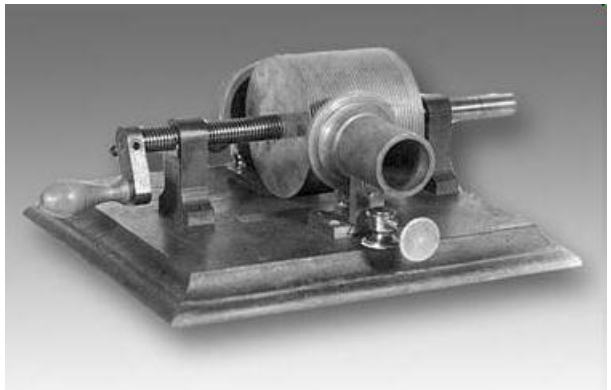
- Implicit Knowledge
  - Internal, unshared knowledge acquired via formal or informal mentoring, and work experience in the work environment
  - The holder of that implicit knowledge may become a bottleneck in applying systems experience on current or future projects
- Explicit Knowledge
  - Implicit Knowledge made known through documentation and sharing

Describing architectures in the context of known and understood patterns should foster better and more consistent understanding across the many stakeholder communities

© Robert Cloutier, 2009

Patterns – From Telegraph Repeater to Phonograph

- Many of Edisonôs world-changing inventions were created through the recognition of patterns of connection between seemingly unrelated technologiesé
- Edisonôs awareness of patterns also allowed him to see how the component parts of solutions he had generated in one area of experimentation could apply to application in another completely different area.
- Years of observing how incoming Morse code messages were tapped out by a chiseled stylus on most telegraph equipment led Edison, in another moment of inspiration, to see how this same chiseled stylus might be used to transfer sound waves onto cylindrical records.
- é As he collected data, Edison would regularly ask ñHave I seen this anywhere else before? Is there a pattern here?ò



Source: Innovate Like Edison ï The Success System of Americaôs Greatest Inventor, Michael J. Gelb and Sarah Miller Caldicott, Dutton, New York, New York, 2007.

Pictures: http://www.videointerchange.com/audio_history.htm

© Robert Cloutier, 2009

Architecture Patterns

System architecture patterns constitute **high-level structures** appropriate to the design of the major elements and subsystems of the **system or enterprise** of interest. They express the relation between the context, a problem, and a solution, documenting attributes and usage guidance. Architecture patterns are **time-proven** in **solving problems** which are **similar in nature** to the problem under consideration.

Value of Architecture Patterns

- Knowledge Management
  - Enables reuse of good concepts and implementations, and to preserve them for future projects
- Control Complexity
  - Architectural patterns may help control the complexity of an architecture by standardizing it on a well known and practiced pattern
- Common Understanding
  - Describing parts of the architecture in the context of known and understood architecture patterns results in a common understanding of the architecture
- Mitigate Risks
  - Using and applying known architecture patterns in architecture design introduce less risk than a new architecture design

© Robert Cloutier, 2009

Patterns Are Not Created - They are Mined

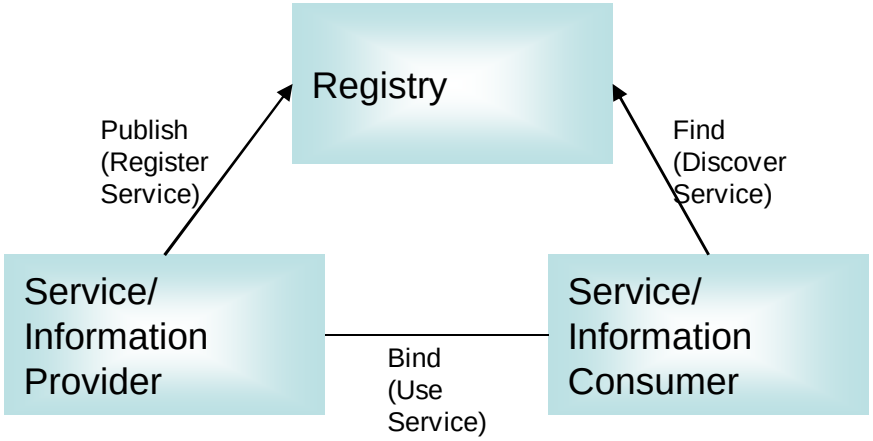
- Over time, similar designs are arrived at independently by different designers on various projects

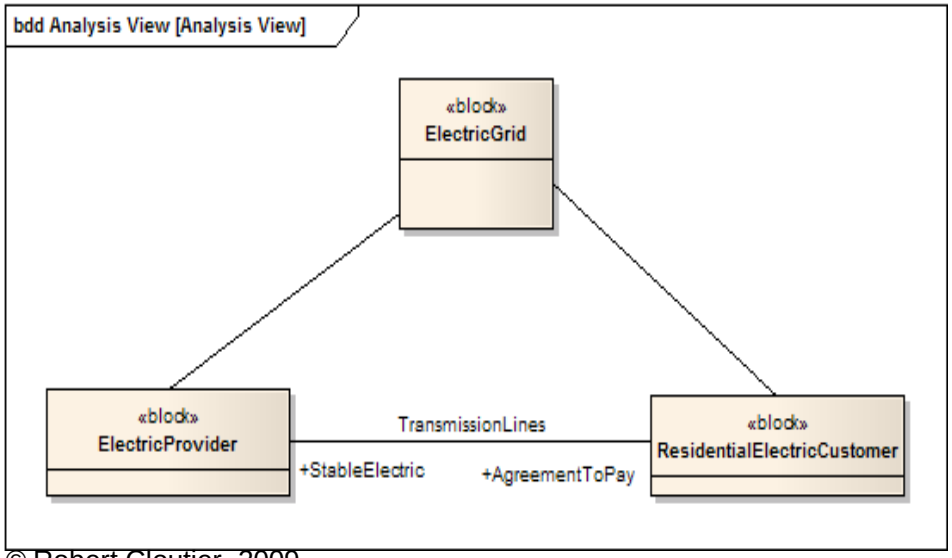
- Where the same elements exist in multiple designs, the design can be studied and documented to encourage reuse
  - This reuse prevents engineers from having to reinvent, or rediscover the same concept

- Rule of 3
  - A concept is only a pattern when one can identify at least 3 independent, observable applications utilizing the proposed pattern as part of the solution

© Robert Cloutier, 2009

Understanding Patterns - The Simple SOA Pattern & Electricity Production and Deliver Example





© Robert Cloutier, 2009

Some Interesting Observations

- Service Level Agreements include:
  - Consistent electricity at constant voltage & wattage, under varying loads
  - Standard plugs
- Consumer can add new AC, Freezer, etc. without notifying electric provider
- Without the electric meter, the electric company has no idea what each consumer consumes
- There are multiple providers on the Grid
  - Electricity is not routed to you - if you are using Green Power Company, those electrons may never arrive at your home
- What happens when demand outstrips supply?
  - Borrowed from another service provider
- Implementation of service is transparent to consumer
  - Consumer does not normally care how it is produced

System Architecture Pattern Drivers

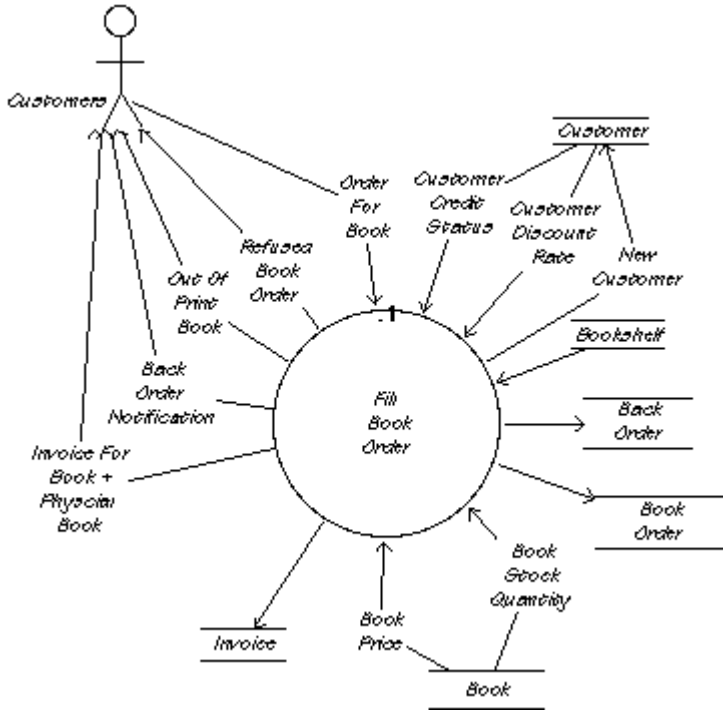
- Corporate systems engineering departments attempt to capture knowledge explicitly through artifacts
- A significant component of this corporate knowledge, however, is implicit
- This implicit knowledge is useful to others only if it is shared
- If a pattern exists only in the form of implicit knowledge, it is not accessible by others
- Developing increasingly complex systems drives a need for a common lexicon between systems architects
- In communities that use patterns, the patterns often become standardized through multiple implementations, presentations at research and professional conferences, and publication in research journals
- This standardization fosters reuse of designs and even code that might be generated from the architecture patterns.
  - Such reuse can in turn improve development efficiency and productivity

© Robert Cloutier, 2009

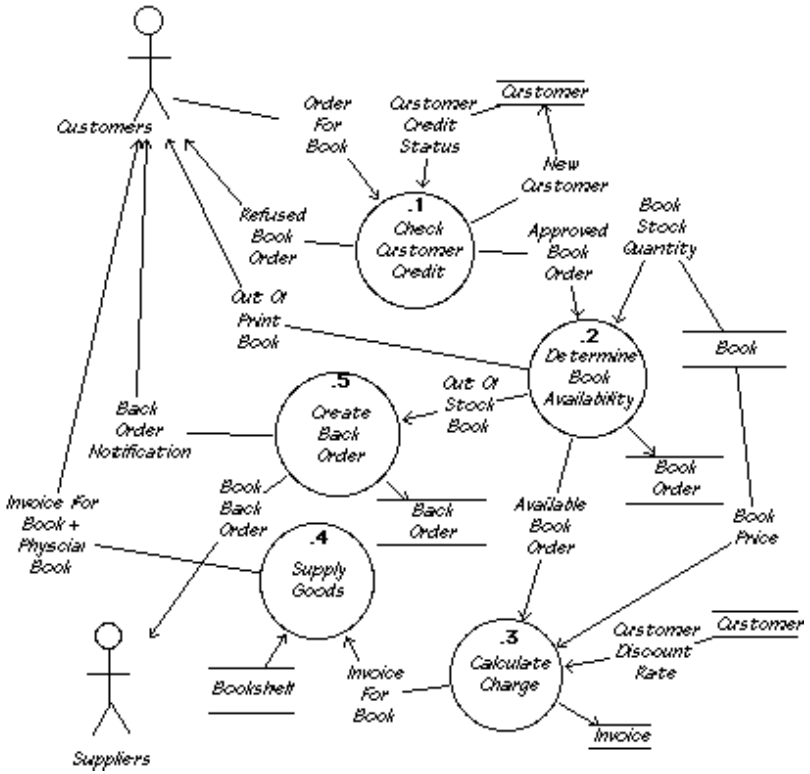
BREAKOUT

© Robert Cloutier, 2009

Book Purchase Example



High Level Use Case for
Customer wants to buy a book

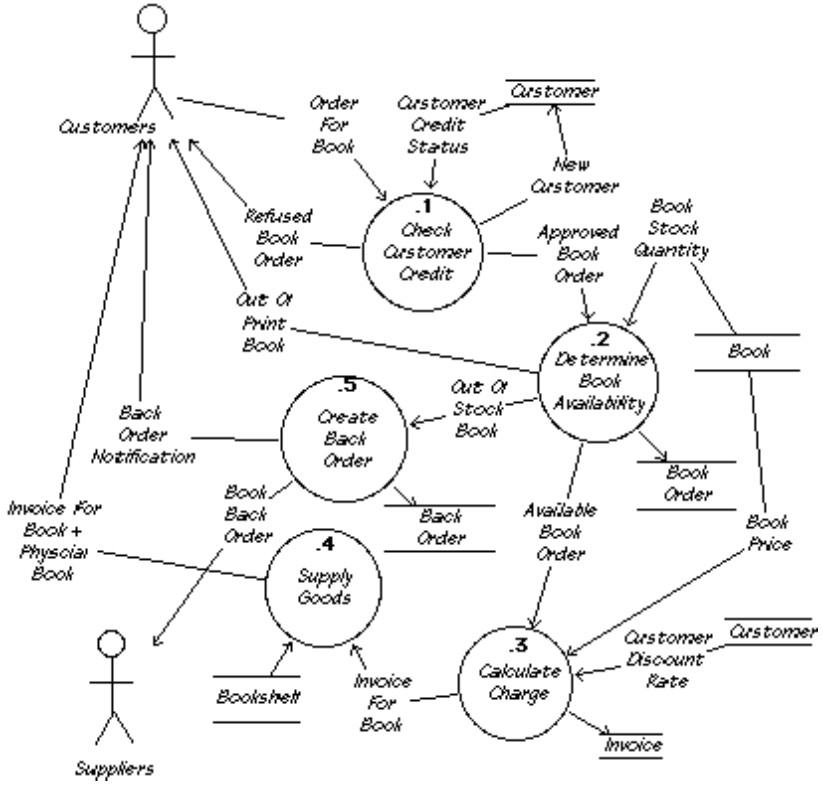


Detailed Use Case for
Customer wants to buy a book

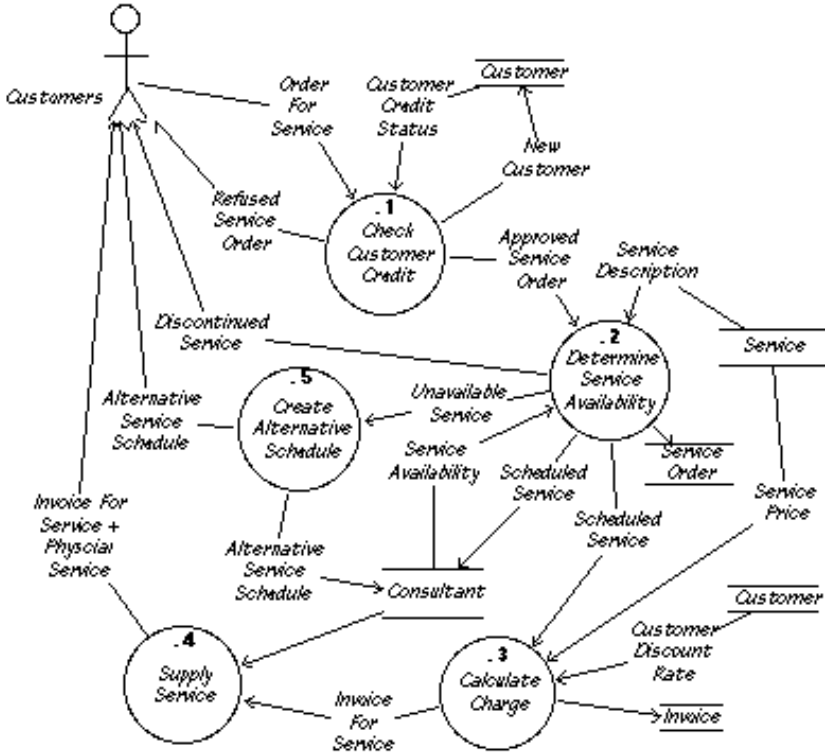
Diagram Source: *Requirements Patterns via Events / Use Cases,*
http://www.systemsguild.com/GuildSite/SQR/Requirements_Patterns.html

© Robert Cloutier, 2009

Generalize to: Service Purchase



Detailed Use Case for
Customer wants to buy a book

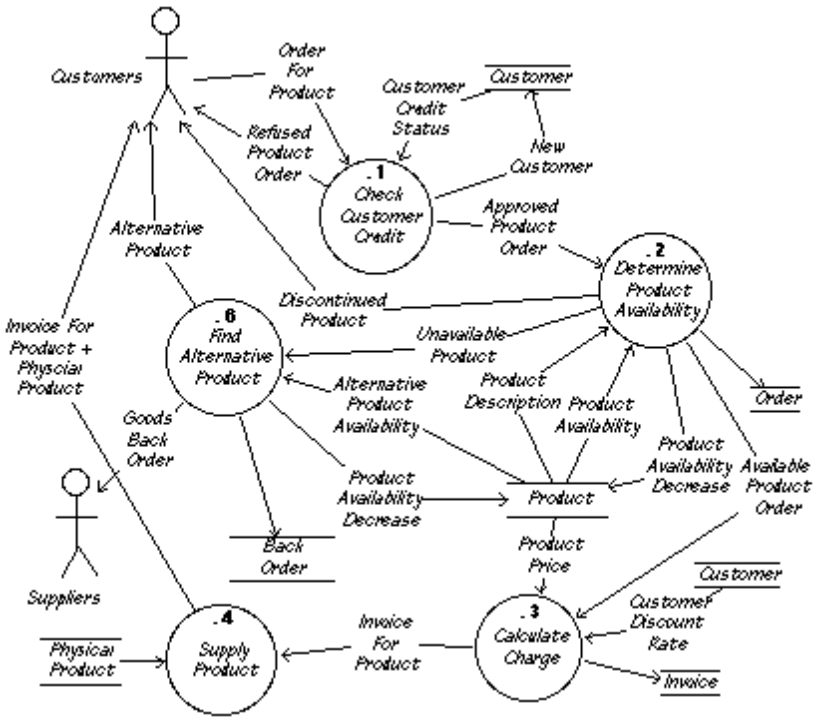


Detailed Use Case for
Customer wants to buy a service

Diagram Source: *Requirements Patterns via Events / Use Cases,*
http://www.systemsguild.com/GuildSite/SQR/Requirements_Patterns.html

© Robert Cloutier, 2009

Generic Customer Purchase Pattern



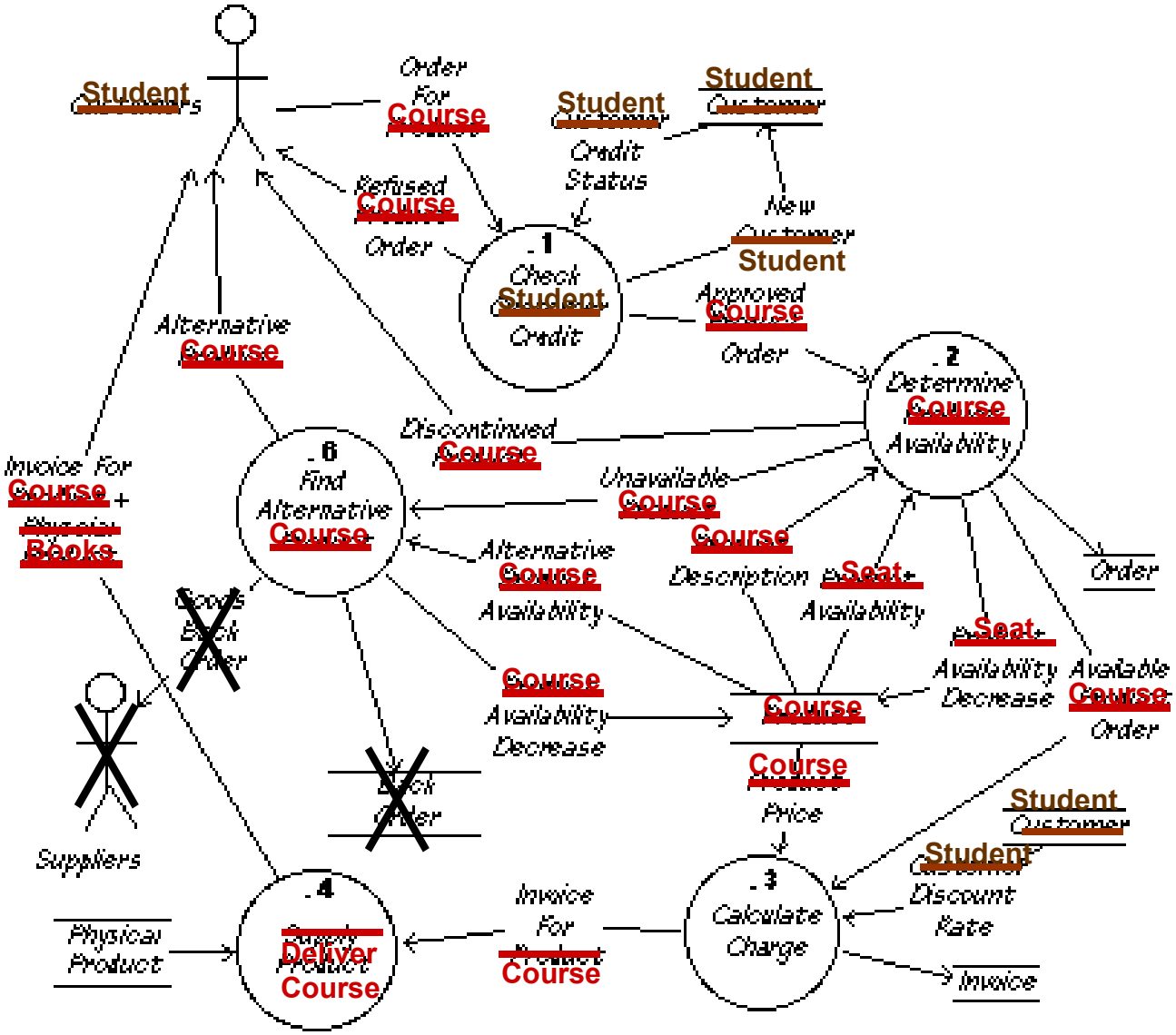
A requirements process pattern for the event: Customer wants to buy a product or service

- Further generalization allows me to use it for purchasing almost anythingé
- Can I use this as a starting point to design an Automobile Purchase system?
- What about to order High Speed Internet?
- What about a course registration system?

Diagram Source: *Requirements Patterns via Events / Use Cases,* http://www.systemsguild.com/GuildSite/SQR/Requirements_Patterns.html

© Robert Cloutier, 2009

Let's Architect a Student Registration System



© Robert Cloutier, 2009

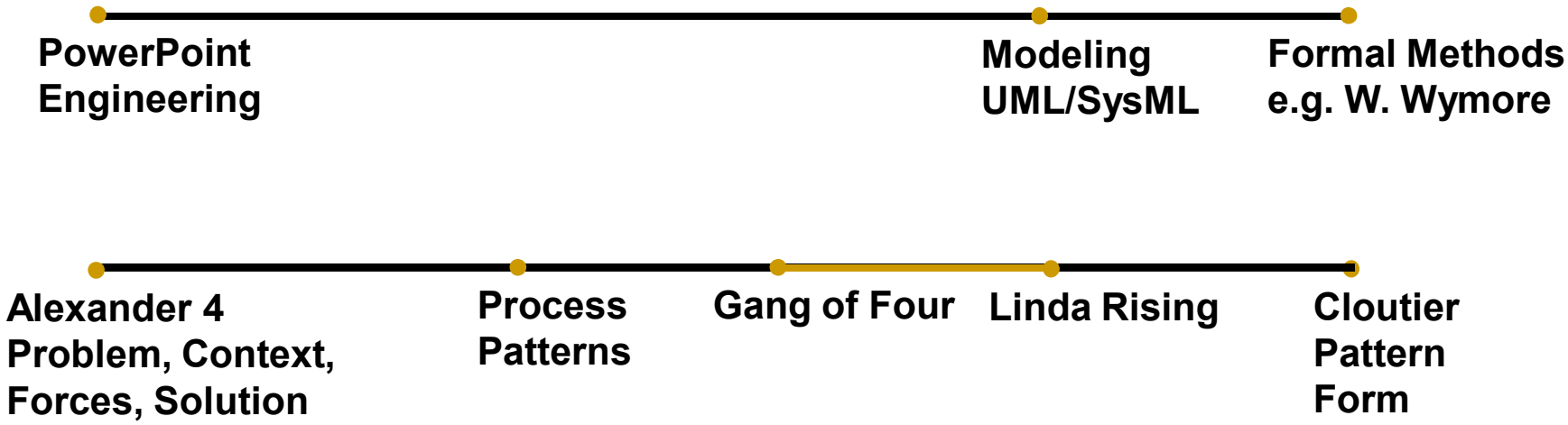
Looking at SW Pattern Forms:

- Was no set manner to describe patterns – even where used extensively
- Appears to be some common attributes used to describe software patterns
- Example to the right are some software pattern description template examples

Pattern Template	Patterns for Effective Use Cases	Architecture Patterns	U.S. Treasury Architecture Guidance	A Pattern Language for Pattern Writing
[Ris03]	[Ado03]	[TOG02]	[TOG02]	[Mes]
Pattern Name	Pattern Name	Name	Name	Pattern Name*
Aliases				*Aliases*
Problem	Problem Statement	Problem	Problem	Problem*
	Metaphoric Story		Implementation	
Context	Context	Context	Structure	Context*
Forces	Forces affecting the Problem	Forces	Interactions	Forces*
			Consequences	*Indications (symptoms)*
Solution	Solution	Solution		Solution*
Resulting Context		Resulting Context	Assumptions	*Resulting Context*
Rationale		Rationale	Rationale	Rationale*
Known Uses				
Related Patterns		Related Patterns		*Related Patterns*
Sketch	Picture			
Author(s)				*Acknowledgements*
Date				
Email				
References				
Keywords				
Example	Examples	Examples		Examples*
		Known Uses		*Code Samples*
				*** - required, *italics* - optional**

© Robert Cloutier, 2009

Thoughts on Pattern Documentation Completeness



© Robert Cloutier, 2009

A Pattern Form for System Architects

HØGSKOLEN i Buskerud

Pattern Name:	The name of the pattern should be descriptive to enable the pattern user to understand the usage
Aliases:	Other names by which the pattern may be known
Keywords:	Keywords which assist in locating appropriate patterns when needed
Problem Context:	Brief discussion of the constraints the pattern may impose
Problem Description:	What is the problem this pattern can be used to solve
Forces:	Challenges that exist in the problem being addressed
Pattern Solution:	Discussion on how the pattern solves the problem being addressed
Sketch:	This can be one or more diagrams necessary to represent the pattern
Interfaces:	Describe critical interfaces/information flows for the pattern
Resulting Context:	What are the unaddressed issues remaining when the pattern is applied/used
Example:	An example of how the pattern may be applied
Known Uses:	Where is the pattern being used
Related patterns:	Patterns that work in conjunction or in association with this pattern
References:	Additional info that may be useful
Pattern Rationale:	Why the pattern works
Author(s):	Who documented the pattern

© Robert Cloutier, 2009

Documenting the Perform C2 Pattern Using the SE Pattern Form

HØGSKOLEN i Buskerud

Pattern Name:	Perform C2, Perform Command and Control
Aliases:	None known
Keywords:	Command and Control, Plan, Detect, Control, Act, C2
Problem Context:	Many systems are designed to perform detailed planning, management and execution of a situation. Command and control is the process of planning, managing and performing a complex set of tasks. This pattern does not address "Prepare" precondition (though one might argue that prepare and plan go together) nor "Assess" post condition.
Problem Description:	In command and control (C2), the situation is typically managed in identifiable phases. The situation may move back and forth between the stages. Those stages are Plan/Detect/Control/Act.
Forces:	Terminology may vary from one domain to the next, and should be adapted in the application of the pattern.
Pattern Solution:	This pattern provides the basis for developing the command and control (C2) interfaces and information that moves through the stages of C2. It provides the A0 context and the first level of decomposition using IDEF0.
Model:	See next page.
Interfaces:	Information flows between the stages of this pattern, as well as feedback loops. Some information is generated only in a particular stage and then output in the form of reports. Names of information can be modified as required by specific domain application. Adapting this pattern may mean that some flows are deleted because they are not required in a particular domain. However, some thought should be given before deletion – for instance, a sensor in the defense business may be radar, while in police work, it may be a person, or even a global positioning capability in a cell phone.
Resulting Context:	Further work will be required to define the tasks to be performed within each stage, and the allocation of tasks to systems, hardware, software or people.
Example:	This pattern may be used in the modeling of a C2 system for military or paramilitary operations system (such as police or homeland defense) where there would be a planning phase, a detection of a situation or "bad guy", an identification and controlling or managing of the information, and a required action to perform the mission. The pattern may even be extended to motor vehicle fleet operations.
Known Uses:	Command and Control applications in various domains
Related patterns:	OODA (Observe, Orient, Decide, Act), MAPE (Monitor, Assess, Plan, Execute), PF2T2EA (Plan, Fix, Find, Track, Target, Engage, Asses)
References:	1. Command, Control, and Communications Systems Engineering, Walter R. Beam 2. Science of Command and Control: Part II, Coping with Complexity 3. MCDP 6 Marine Corps Command and Control Handbook
Pattern Rationale:	This is a time-tested doctrine used by the military that may be applicable to other domains.
Author(s):	Harry Johnson Ph.D., Ken Hartnett, Satya Moorthy, Robert Cloutier. 2006.

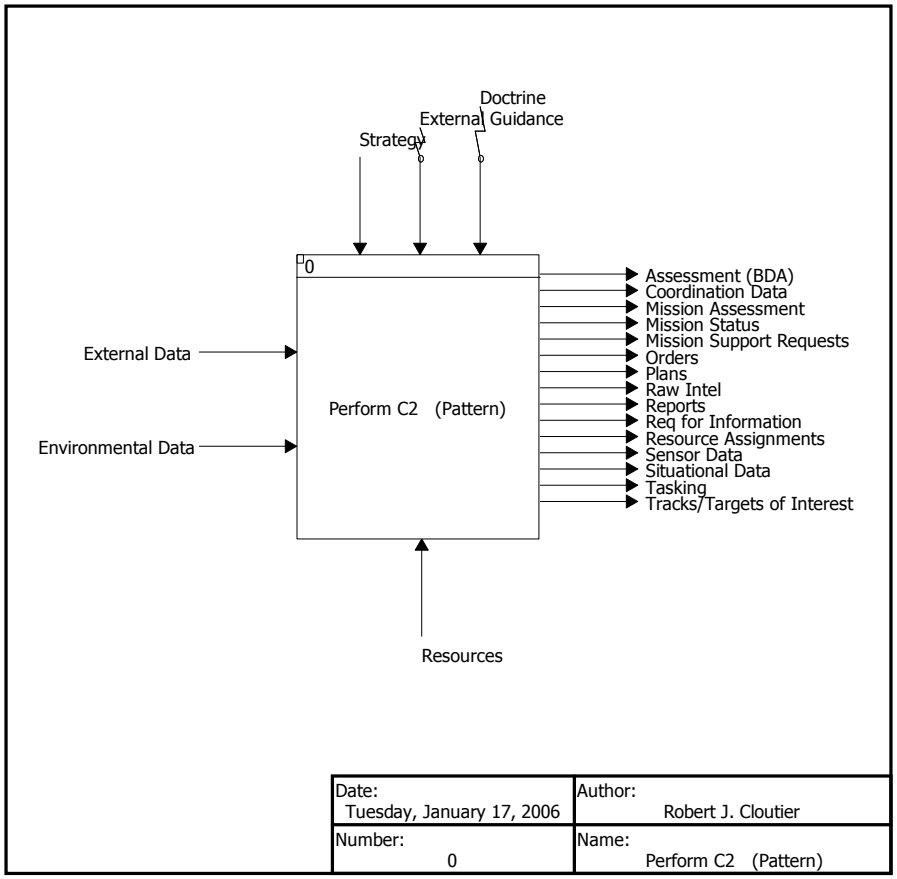
© Robert Cloutier, 2009

System Architecture Pattern Example - Perform C2 A-0 Diagram

- Pattern began as a successfully implemented logical architecture, on the first system.
- Next project for that same architect had similar requirements - able to reuse most of the logical architecture.
- A year later, a mid-grade architect assigned to a similar project - his mentor told him about previous work.
- The two architects went back to the earlier project models, and used them to guide the future logical architecture.

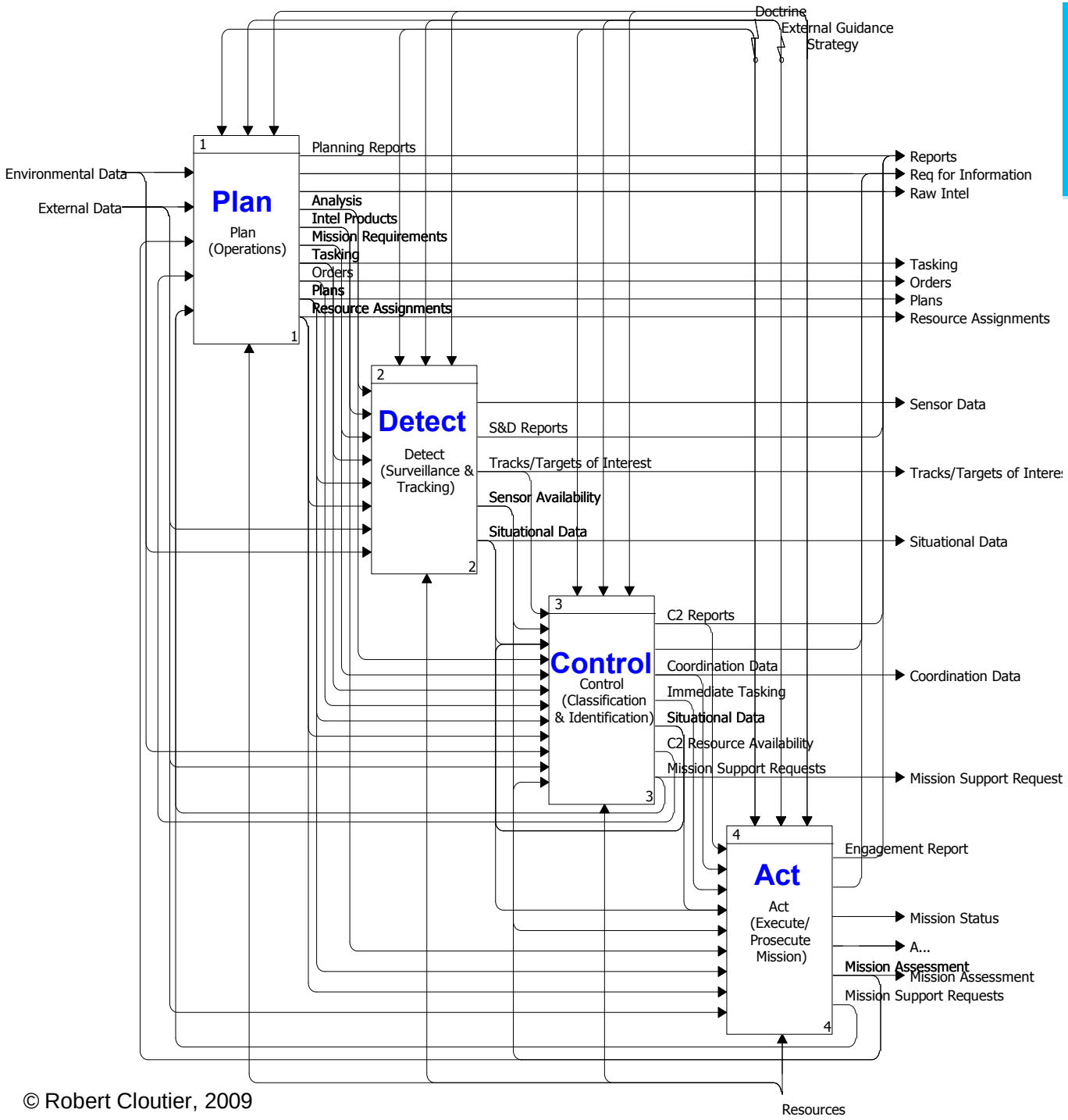
- Later, they found out about my research.

- The Perform C2 pattern was documented using those guidelines.



While the pattern is documented in IDEF0 (instead of UML/SysML), it does demonstrate the magnitude of a system architecture pattern

© Robert Cloutier, 2009



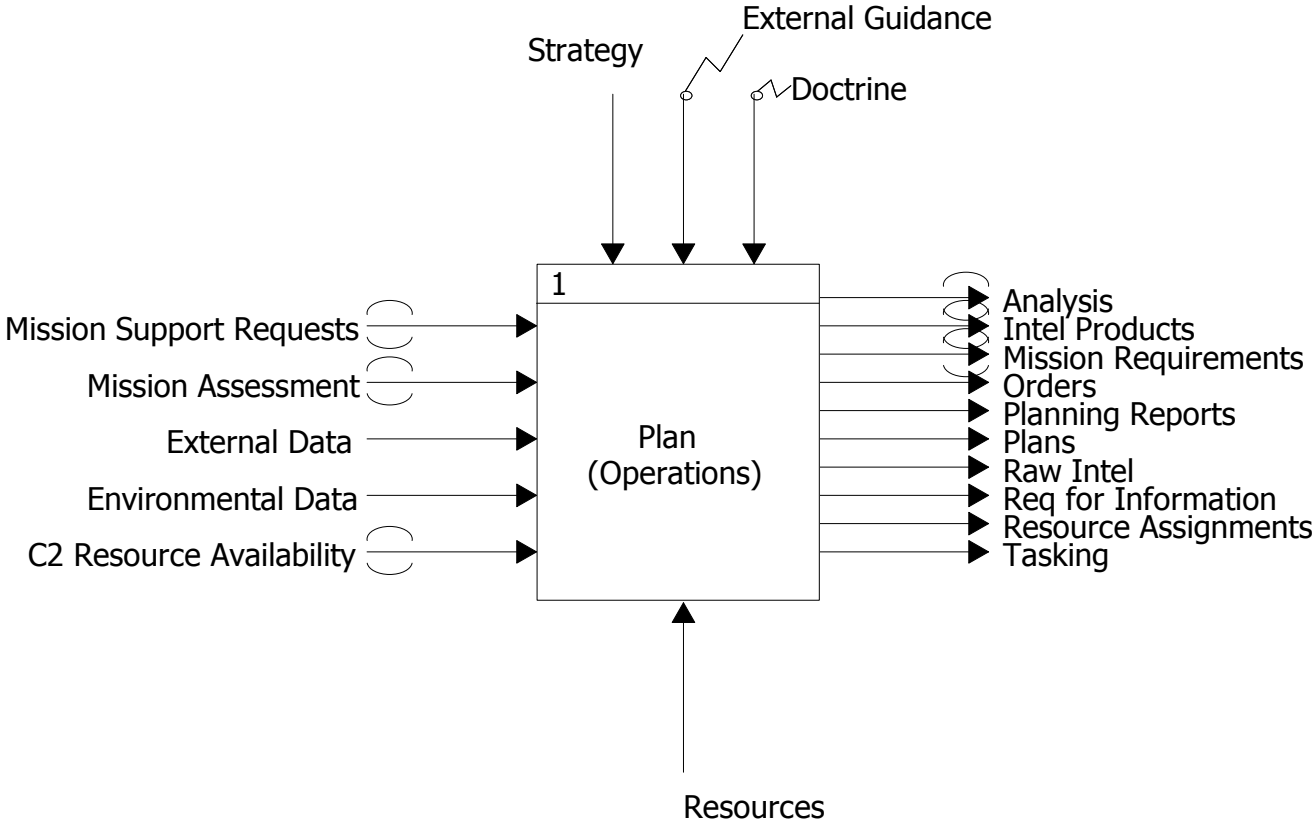
© Robert Cloutier, 2009

HØGSKOLEN i Buskerud

- This pattern was mined from 4 different systems
  - Including a Coast Guard system and a DHS proposal)
- Inputs and Outputs are generalized for reuse
- When using a pattern like this,
  - Unnecessary IERs are deleted
  - New and unique IERs are added

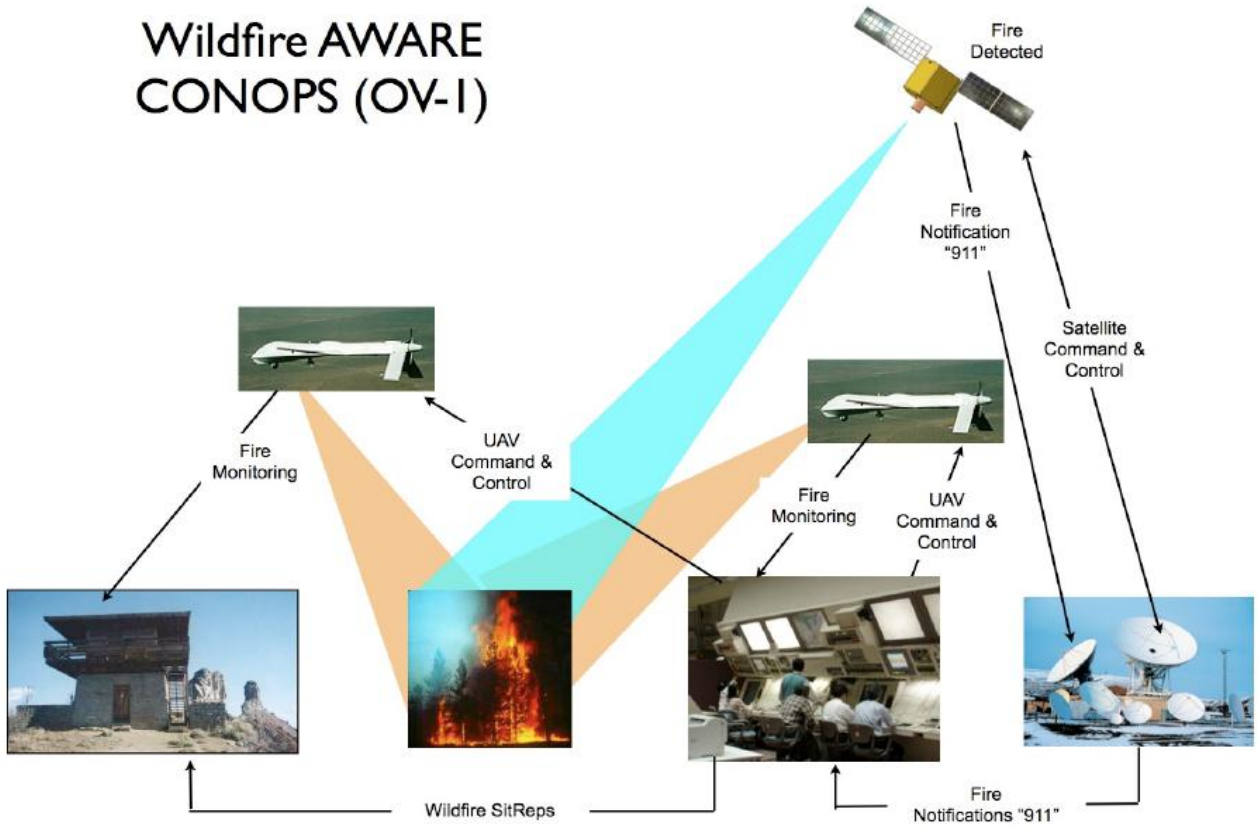
- **Has captured implicit and explicit knowledge from C2 experts and can be used for:**
  - A Stakeout
  - Planning a seizure
  - Tracking a potential threat
  - Prosecuting a threat
  - Controlling assets in a coordinated operation (the extended enterprise)

The Smaller "Plan Operation" Pattern



© Robert Cloutier, 2009

The FireSAT Mission Example

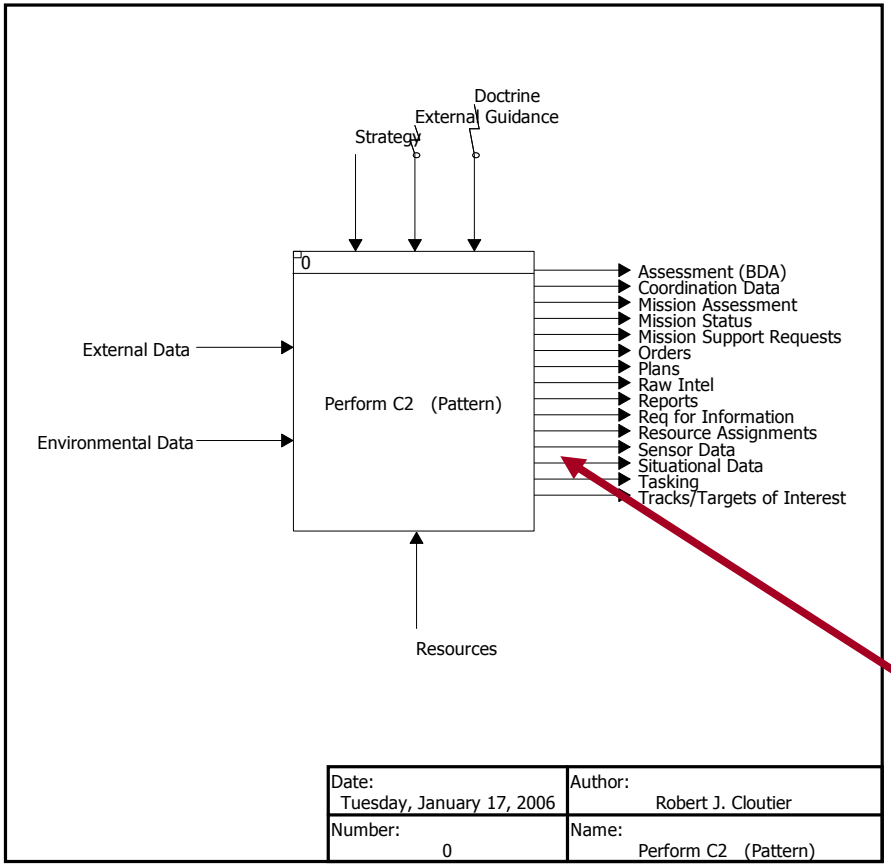


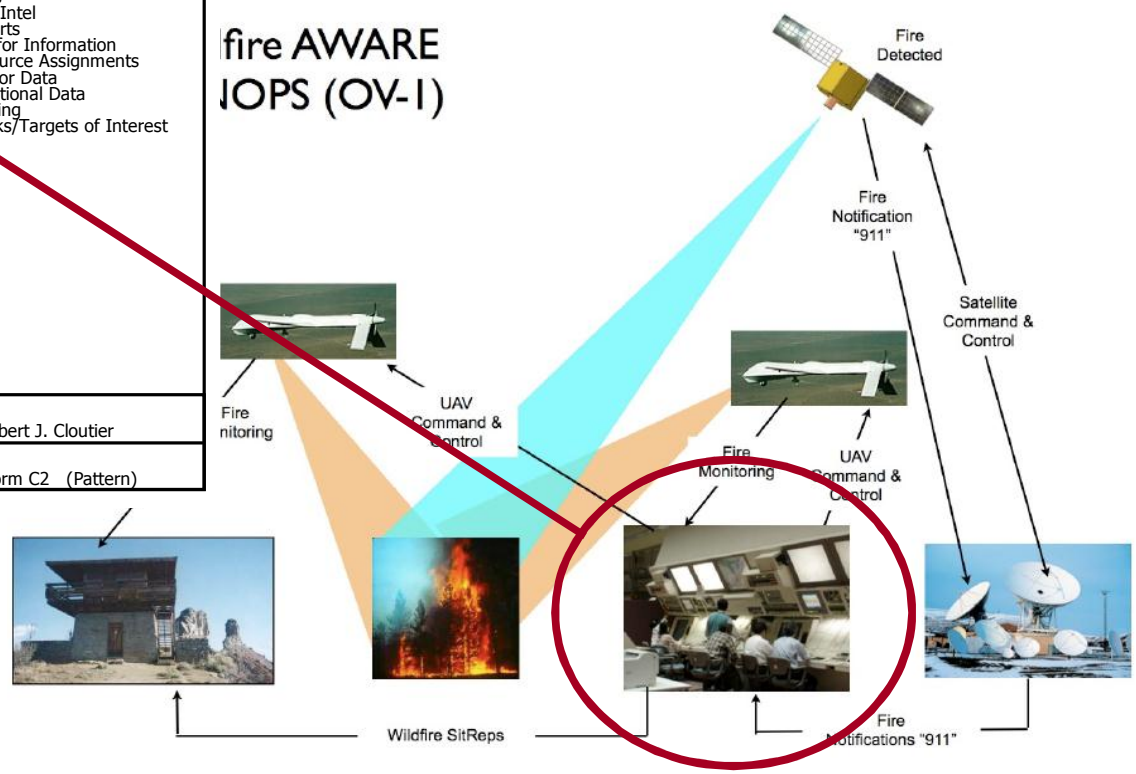
- Introduced in 1998 by Larson & Wertz
- Intended to be an exercise for space engineering students
- Based on the fictitious NASA and the US Forest Service requesting a study
- Investigate feasibility of developing the mission concept for an on orbit fire detection system
- The stated system need:

  The United States needs a more effective means to detect and monitor potentially dangerous wildfires

© Robert Cloutier, 2009

Applying the Perform C2 to FireSAT A-0 Diagram





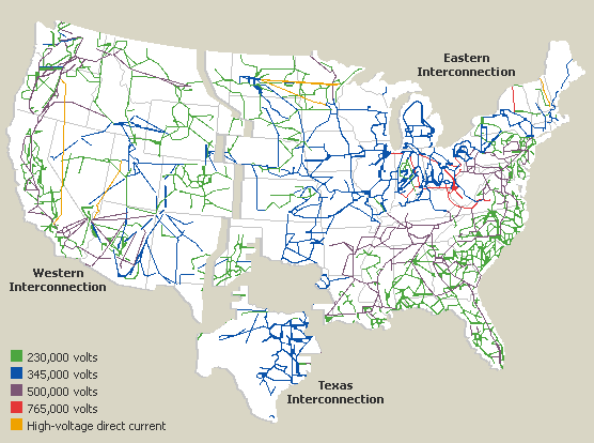
© Robert Cloutier, 2009

DRAFT - Hubs and Nodes Pattern

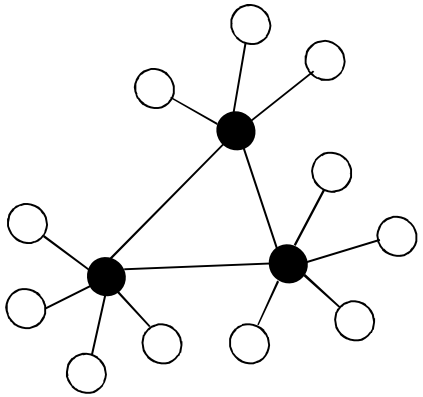
Pattern Name:	Hubs and Nodes
Aliases:	Small world network, Hubs and Spokes
Keywords:	Robustness, Resilience, small world network, error tolerant, attack tolerant
Problem Context:	More and more it is important to build distributed systems that are robust, able to withstand attacks from threats, and to be error tolerant
Problem Description:	When there is a need to distributing system capability,
Forces:	Cost of implementation and duplication of capability/information must be considered.
Pattern Solution:	By distributing the capability/knowledge across hubs instead of each node, cost and complexity is more balanced while providing a resilient system
Sketch:	See next page
Interfaces:	It is desirable for the interfaces between the hubs to be identical to those between hubs and nodes. The only difference may be the bandwidth – more information will most likely travel between the hubs
Resulting Context:	By distributing the capability/knowledge across hubs instead of each node, cost and complexity is more balanced while providing a resilient system
Example:	In an SOA, services might be deployed at the hubs, and connected nodes access closest available hub
Known Uses:	Power grids, Airline routing, Physical package routing, the Internet
Related patterns:	The Perform C2 may work well since current thought is to distribute that capability
References:	A. Barabasi, M. Vidal, J. Park
Pattern Rationale:	Considerable research has shown that this is an both a designed architecture and an emergent architecture which provides for robustness and resilience
Author(s):	Cloutier, 2009

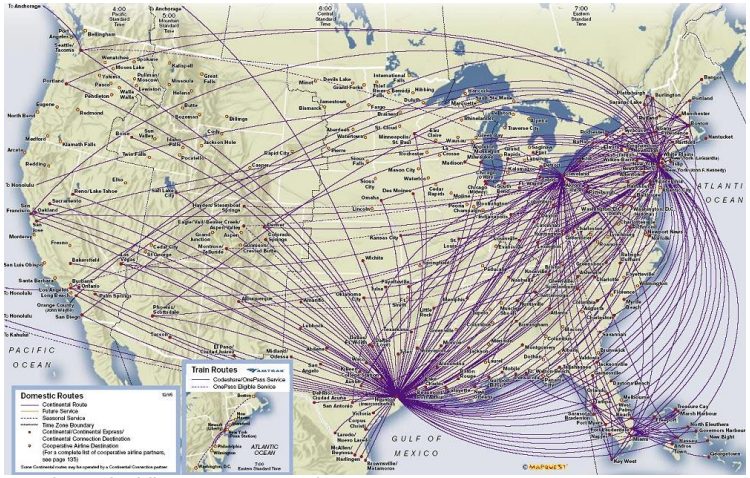
© Robert Cloutier, 2009

DRAFT - Hubs and Nodes Architecture

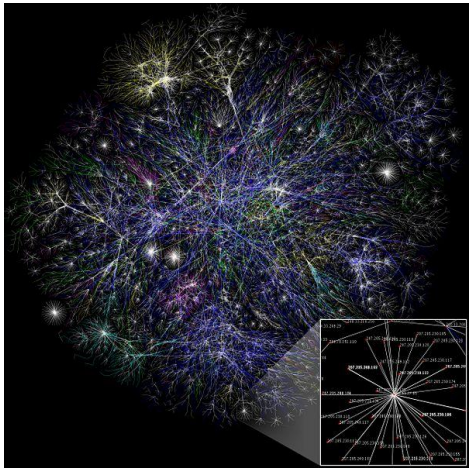


National Power Grid
http://encarta.msn.com/media_701509077/the_national_power_grid.html

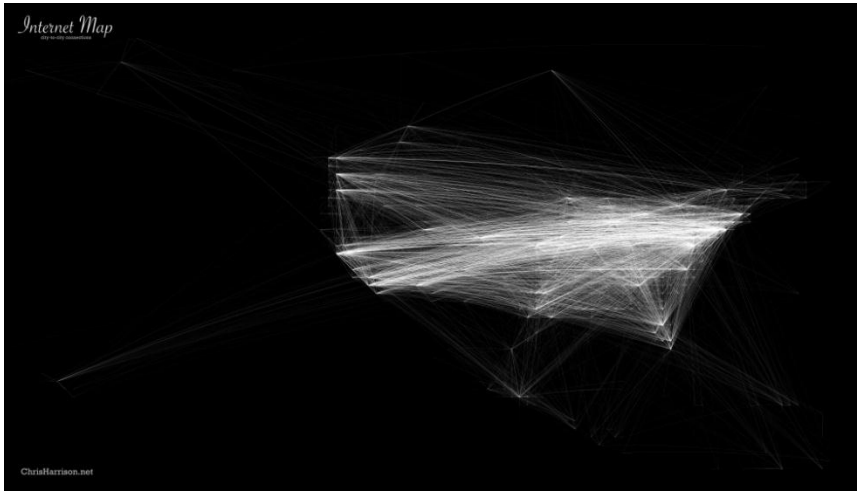




Continental Airlines route network
http://www.airlineroutemaps.com/USA/Continental_Airlines_usa.shtml



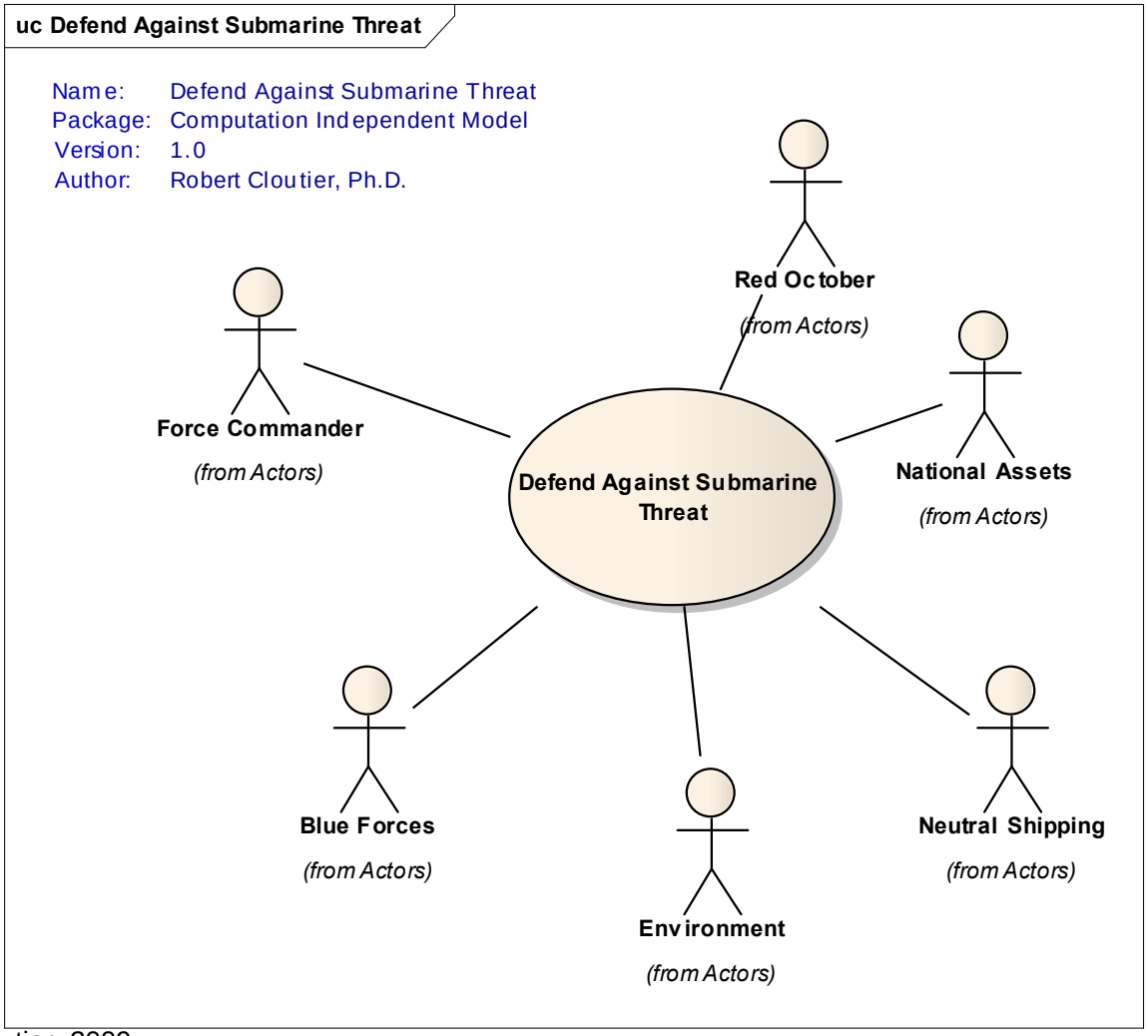
Visualization of the various routes through a portion of the Internet. (wikicommons)
http://janeporter.files.wordpress.com/2008/07/600px-internet_map_1024.jpg



Internet Map
http://www.chrisharrison.net/projects/InternetMap/

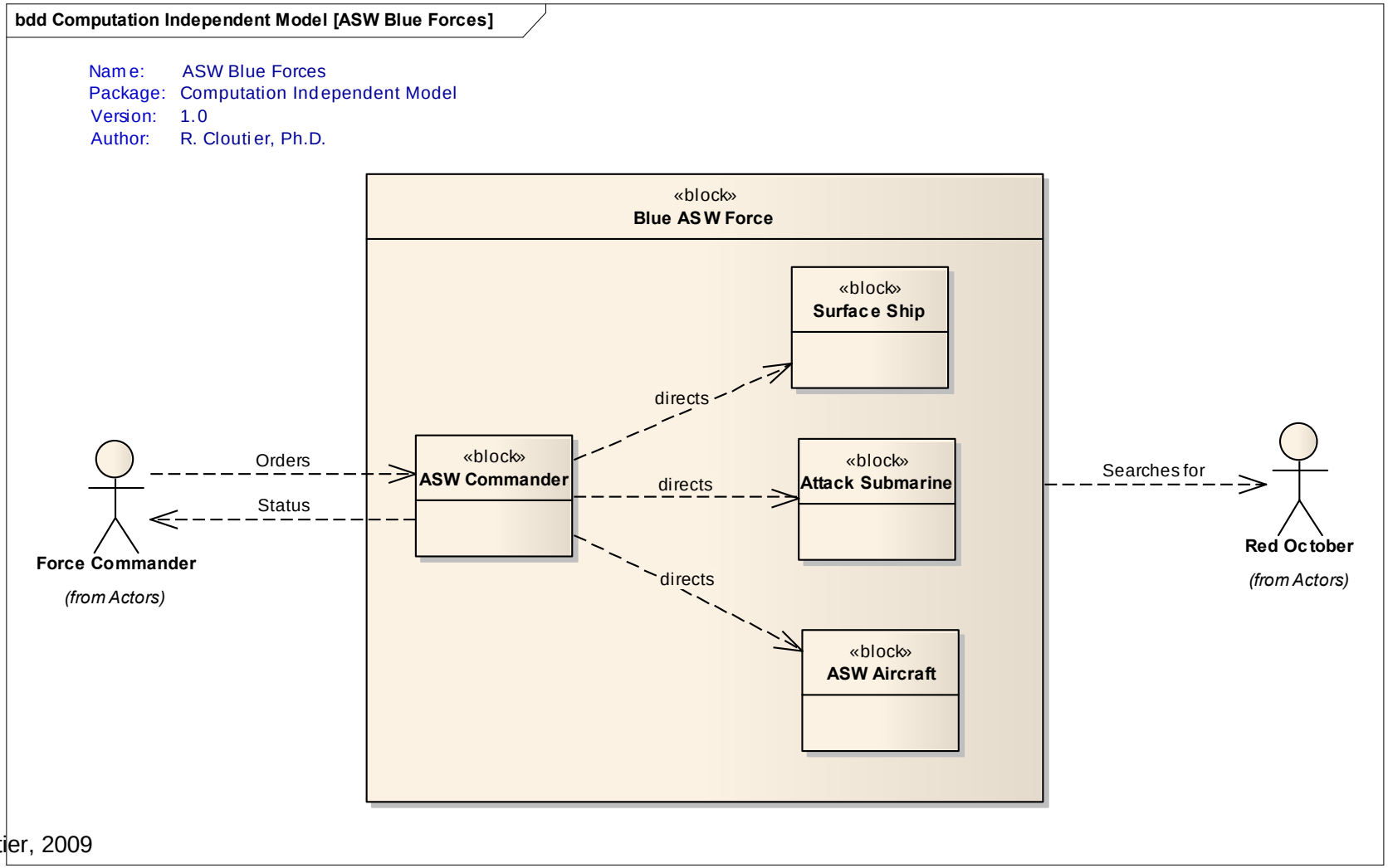
© Robert Cloutier, 2009

An Overly Simplified Example
Hunt for Red October Pattern



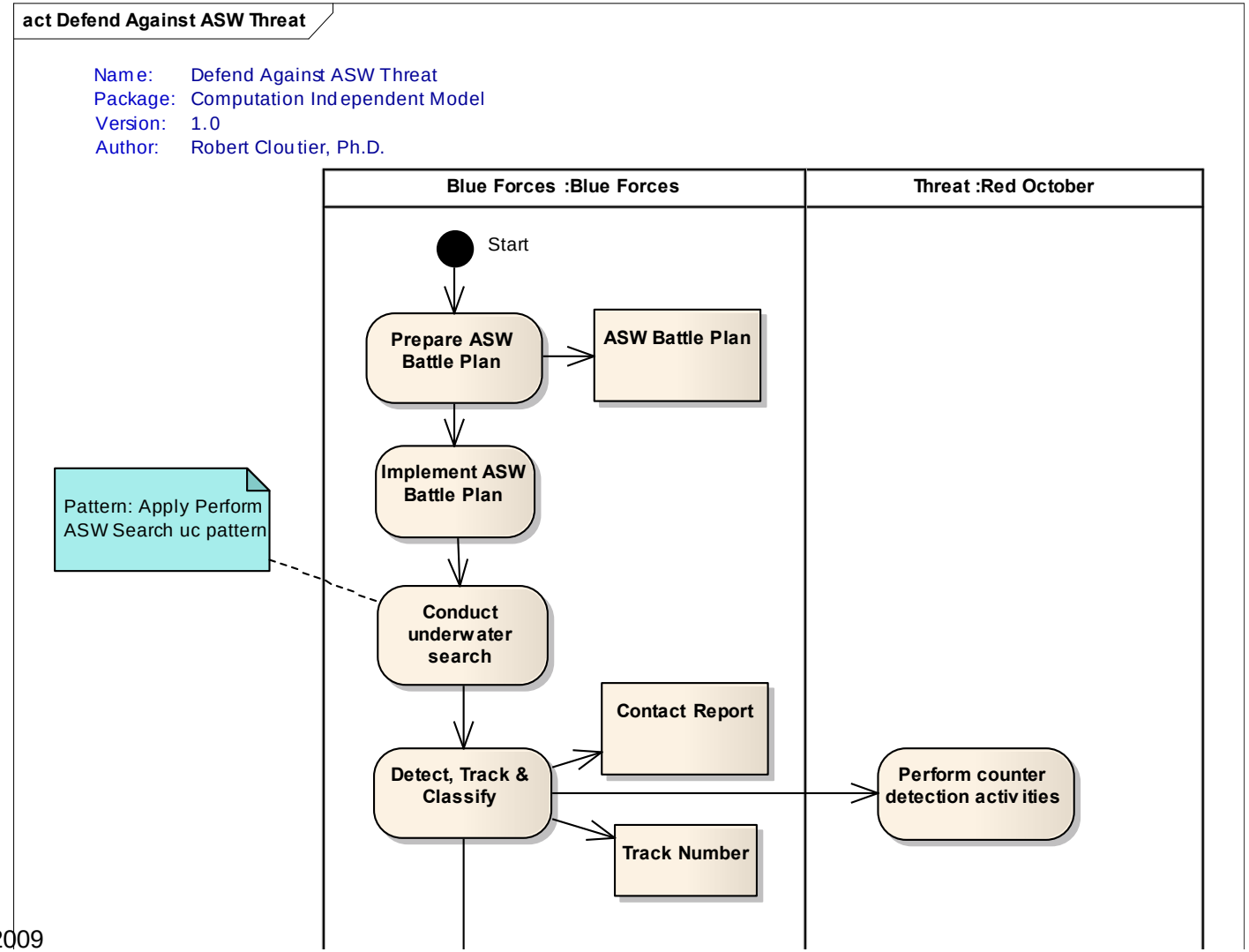
© Robert Cloutier, 2009

Good Guy Block Definition Diagram



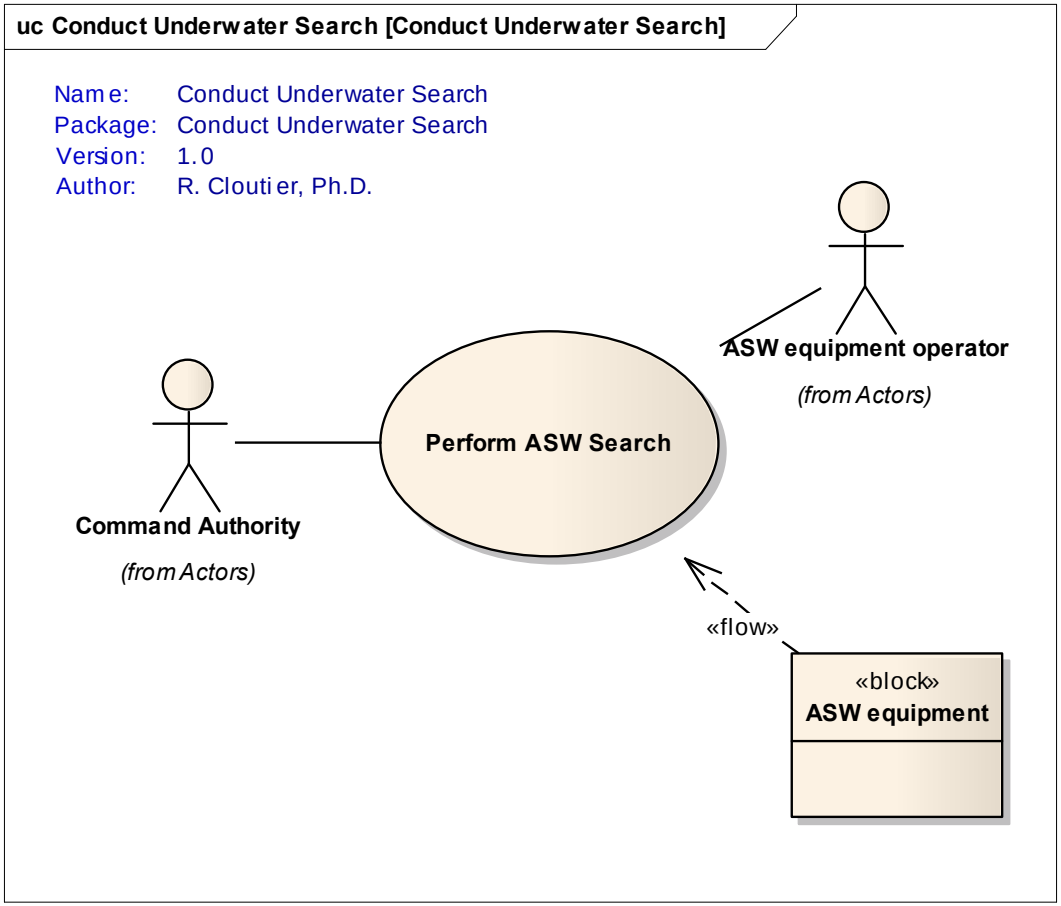
© Robert Cloutier, 2009

Defend Against Submarine Threat with Pattern Tag



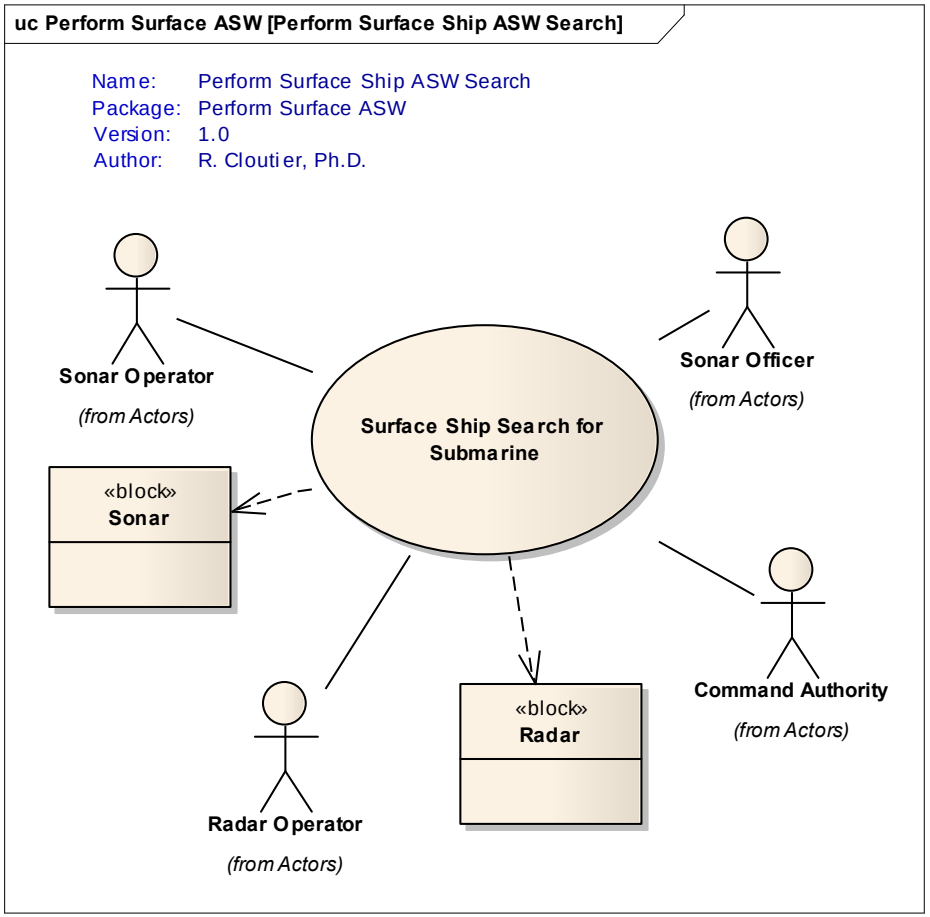
© Robert Cloutier, 2009

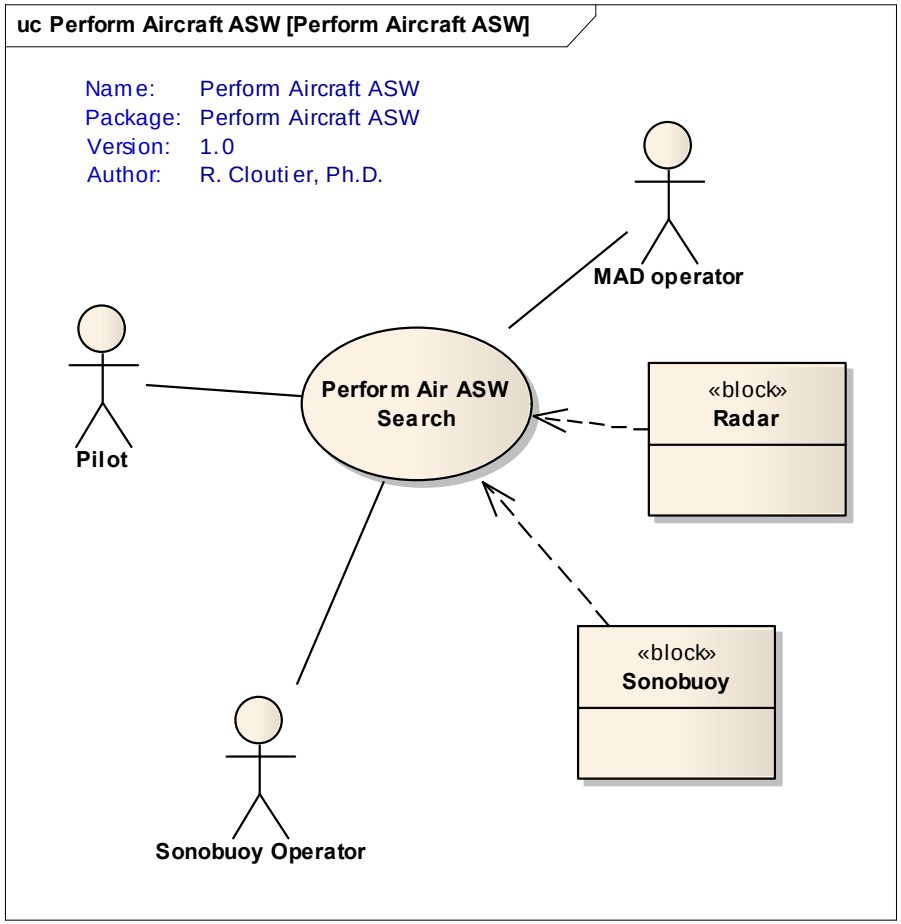
Perform ASW Pattern



© Robert Cloutier, 2009

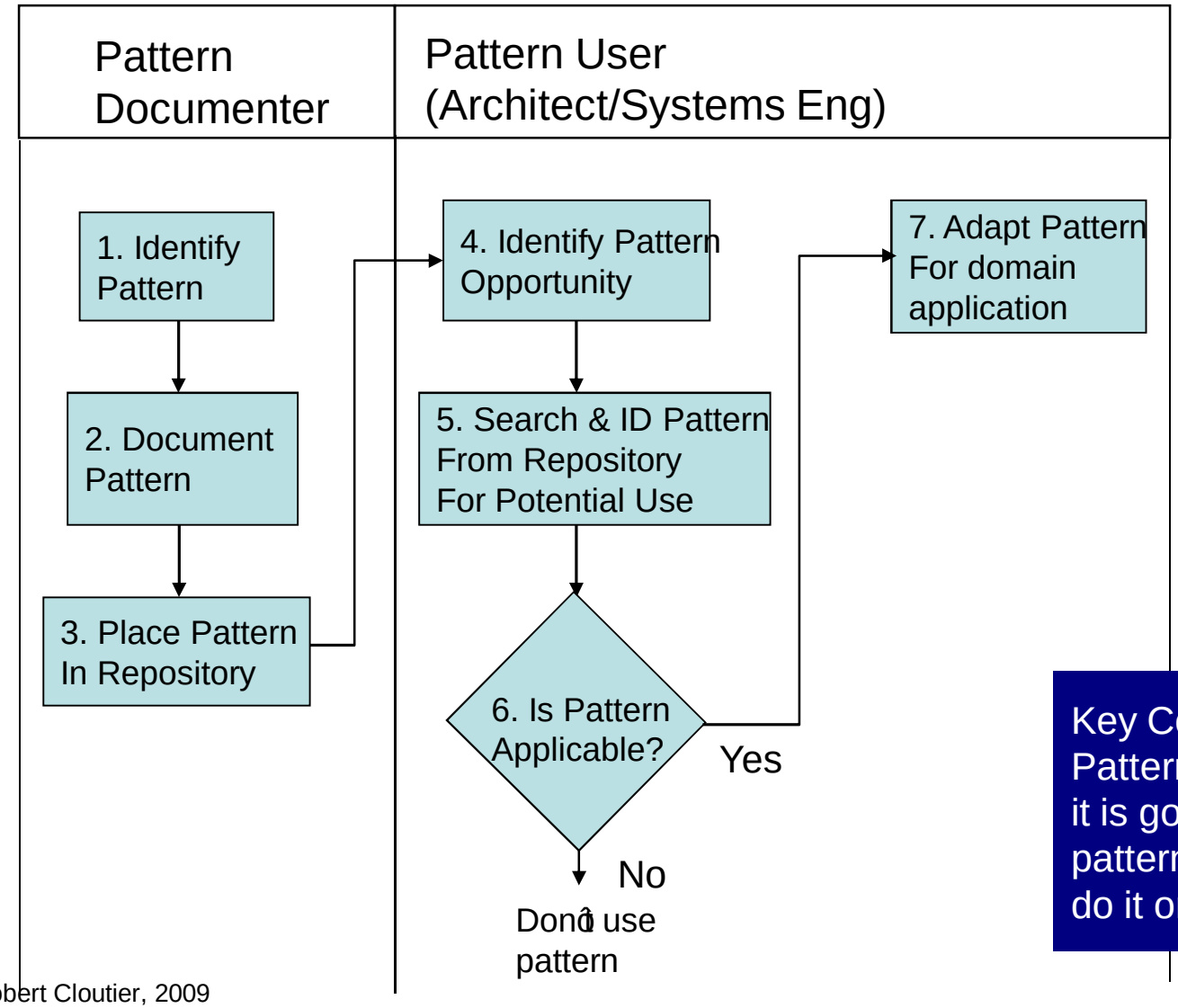
Applied Perform ASW Search Pattern





© Robert Cloutier, 2009

Thoughts on a Pattern Utilization Process



Key Concept:
Patterns are optional ï while it is good to consider using a pattern, do not force the use ï do it only if it makes sense

© Robert Cloutier, 2009

Challenges in Using Patterns

- Mining patterns
  - As one moves up the complexity spectrum of systems, there are fewer that can be mined for patterns
    - Usually represent systems from different companies ï IP issues
    - Acquisition timeline so long, technology changes by the time a system is fielded
  - System Documentation differs, if one can even locate ité
    - System Engineering tool of choice ï PowerPoint
    - IDEF0 -predominately Telelogic (Popkin) SA and Vitech CORE
    - UML ï predominately IBM (Rational) Rose, Telelogic TAU, Telelogic (iLogix) Rhapsody
    - SysML emerging (from a host of vendors)
- Methodology Wars
  - Some systems developed using Functional decomposition, some using Object Oriented decomposition
  - Will result in different ñlookingò solution at high levels
- Correct level of documentation to satisfy the intent of patterns

- Recommend you not develop your own pattern lexicon
  - will make it difficult to talk about patterns in other parts of your companies
  - Will make it difficult to share patterns across broader patterns community

© Robert Cloutier, 2009

http://www.patterns4se.com

HØGSKOLEN i Buskerud

Center for Patterns & Systems Engineering

SSE School of Systems&Enterprises

STEVENS Institute of Technology

Home | About Patterns & SE | Research | Publications | People | Contact Us

Pattern Repository

Patterns Blog

Contact Us

Robert Cloutier, Ph.D.
Associate Professor
School of Systems and Enterprises
Stevens Institute of Technology
Hoboken, NJ 07030
email me

The Center for Patterns and Systems Engineering is focused on promoting and extending the application and use of patterns in system architecting and systems engineering. While the work is based on the research of Dr. Cloutier at Stevens Institute of Technology, it is meant to be a source and repository for all things relating to patterns and systems architecting and systems engineering.

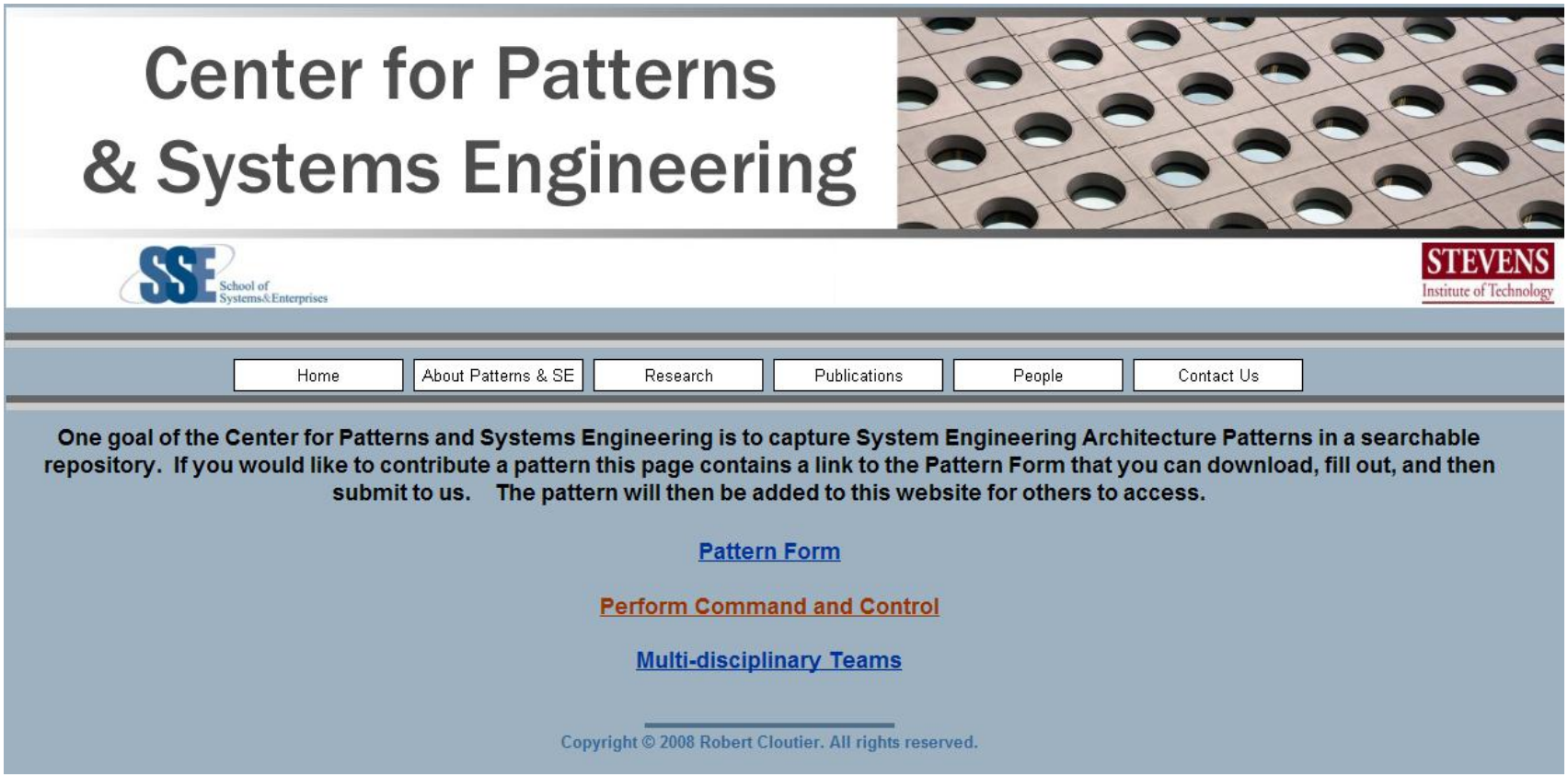
The goal is to present both industry best practices and ongoing research associated with systems engineering patterns. To this end, we want to provide a repository for systems engineering patterns. We are looking to you, the practioners, to submit your patterns using the form provided on the Patterns Repository tab and we will post them for all to see.

If you have recommendations for this site, or would like to contribute to the research, please feel free to contact us.

Copyright © 2008 Robert Cloutier. All rights reserved.

© Robert Cloutier, 2009

The Beginnings of a Patterns Repository



© Robert Cloutier, 2009

http://www.sepatterns.typepad.com

HØGSKOLEN i Buskerud

Patterns & Systems Engineering

SSE School of Systems & Enterprises

STEVENS Institute of Technology

ABOUT THIS BLOG

Why patterns for Systems Engineering? What are Systems Engineering patterns? These are a few topics that will be discussed on this blog. This blog is directed by Dr. Robert Cloutier of Stevens Institute of Technology. Read More >>

SYSTEMS ENGINEERING PATTERNS LINKS

Patterns and Systems Engineering Home Page

The School of Systems and Enterprises

Robert Cloutier Faculty Profile

Enter your email address:

Subscribe

Delivered by FeedBurner

Subscribe in a reader

BOOKMARK

Blog powered by TypePad

« Interesting Pattern Books | Main

February 18, 2009

TRIZ and Systems Engineering

Seems anytime I begin to talk about patterns and systems engineering, someone in the audience will ask me if I have looked at TRIZ (pronounced Trees). While I am aware of what TRIZ is, I have never really done an indepth look at what it could offer - until now. TRIZ is an approach to inventive problem solving - that is to say, it is a methodology with 40 principles to help the inventor invent something new to address a problem. Sounds like a similar selling point for patterns - a collection of proven approaches to solving a problem, that might be useful in solving a similar problem. So now I have begun looking at TRIZ deeper to see what it offers. I have found two books that seem to be pretty good at helping one learn TRIZ, without attending a dedicated seminar. The first was written by the inventor of TRIZ, Genrich Altshuller, and has been translated into english (did I mention this was invented by a Russian?). First published in 1997, and updated in 2005, it is simply titled "40 Principles". The second book caught my attention because I am a fan of Da Vinci. The book is "Da Vinci and the 40 Answers". While both look like "lightweight" books, both require thought - real thought - to understand the approach. However, I firmly believe this is worth the effort. I will keep you posted. Finally, I would be remiss if I did not mention the Altshuller Institute, as they are a wealth of information, including a monthly eJournal.

Posted at 08:00 PM | Permalink

TrackBack

TrackBack URL for this entry:

http://www.typepad.com/services/trackback/6a00e5536796128834011278faf15c28a4

Listed below are links to weblogs that reference TRIZ and Systems Engineering:

ARCHIVES

February 2009

September 2008

August 2008

July 2008

CATEGORIES

Patterns Related Topics Research

FEBRUARY 2009

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

RECENT POSTS

TRIZ and Systems Engineering

A Perform Command and Control Pattern

A Recommended Pattern Form for Systems Engineers

What is a Pattern Form?

Patterns for systems architecting and systems engineering

© Robert Cloutier, 2009

My Four Pattern Take Aways

1. Patterns are natural to think about, but harder to capture in useful form
2. Patterns are an enabler to allow systems engineers and project managers to have time to focus time and dollars to push other technical boundaries
3. Patterns are a learning tool to help others share our experience
   - But they must be captured in a manner in which they are useful to others - formally
     - ***The greatest obstacle to discovery is not ignorance – it is the illusion of knowledge*** ***David. J. Boorstin***
4. If you or your company are going to do anything with patterns in SE, I want to play!

© Robert Cloutier, 2009

Contact Info:

Dr. Robert (Rob) Cloutier

robert.cloutier@hibu.no

Robert.cloutier@stevens.edu

© Robert Cloutier, 2009