

Software Architecture Knowledge Representation

Philippe Kruchten
JA00
October 7, 2009

2

Philippe Kruchten, Ph.D., P.Eng., CSDP

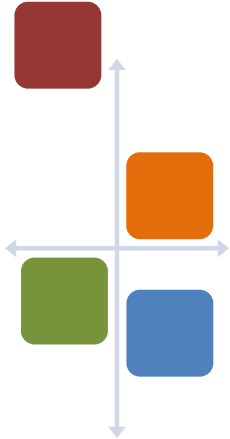


Professor of Software Engineering
NSERC Chair in Design Engineering
Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, BC Canada
pbk@ece.ubc.ca
+1 604 827-5654



Founder and president
Kruchten Engineering Services Ltd
Vancouver, BC Canada
philippe@kruchten.com
+1 604 418-2006

3



Outline

- Software architecture
- Knowledge management
- Motivation
- Architecture representation
- $AK = AD + ADD$
- Processes
- Tools
- Summary

Slides at: pkruchten.wordpress.com/talks/

4

Software Architecture

Software architecture encompasses the set of **significant decisions** about

- the **organization** of a software system,
- the selection of the **structural** elements and their **interfaces** by which the system is composed together with their **behavior** as specified in the collaboration among those elements,
- the **composition** of these elements into progressively larger **subsystems**,

*Grady Booch, Philippe Kruchten, Rich Reitman, Kurt Bittner; Rational, circa 1995
(derived from Mary Shaw)*



5

Software Architecture (cont.)

- the architectural **style** that guides this organization, these elements and their interfaces, their collaborations, and their composition.

Software architecture is not only concerned with structure and behavior, but also with usage, functionality, performance, resilience, reuse, comprehensibility, economic and technological constraints and tradeoffs, and aesthetics.



6

Software architecture...

- architecture = { elements, form, rationale } *

Perry & Wolf 1992

- A skeleton
- More than structure
- Embodies or addresses many “ilities”
- Executable, therefore verifiable
- Emergent? Sometimes...



7

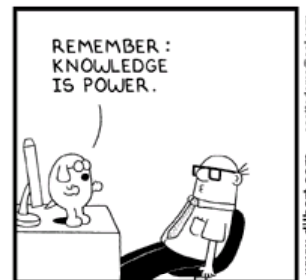
Knowledge

- Expertise, and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject
- What is known in a particular field or in total; facts and information
- Awareness or familiarity gained by experience of a fact or situation
- Plato: knowledge is "justified true belief"

8

Knowledge as an asset

- “Intellectual capital”
- Knowledge workers
- “Knowledge is power”
- Knowledge management:
 - sharing, distributing, creating, capturing and understanding the knowledge of an organization



9



“The major problem with intellectual capital is that it has legs and walks home every day.”

Rus & Lindvall 2002

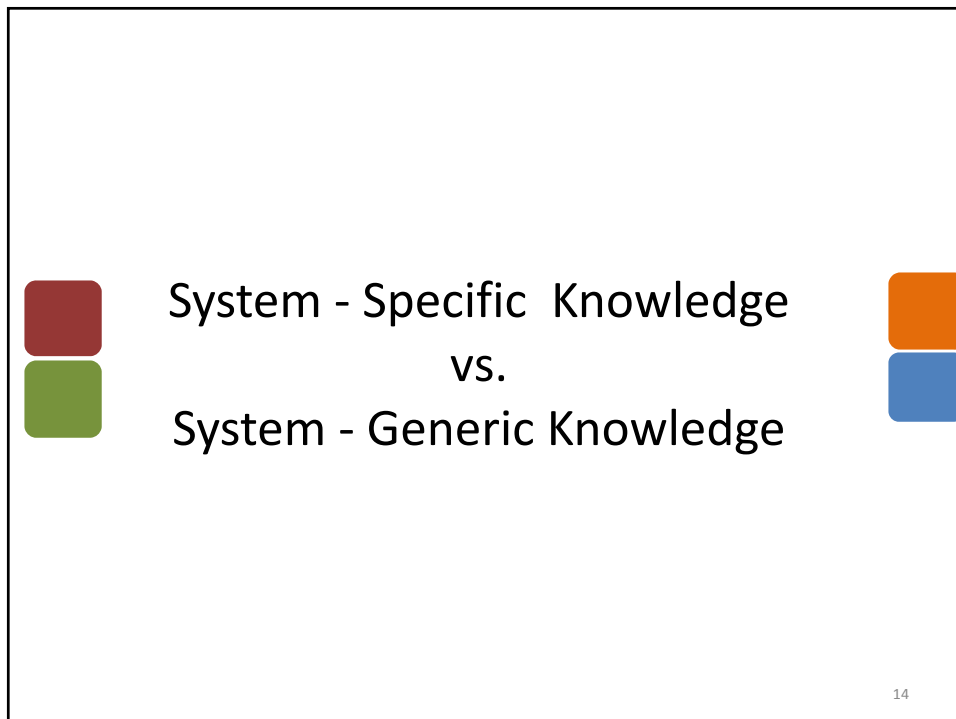
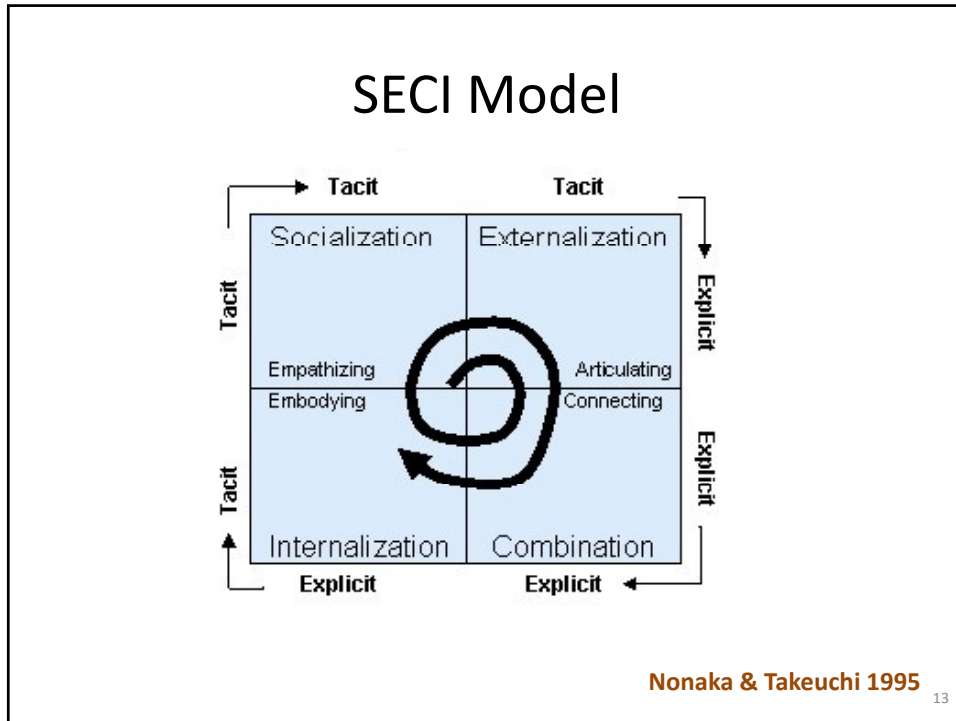
11

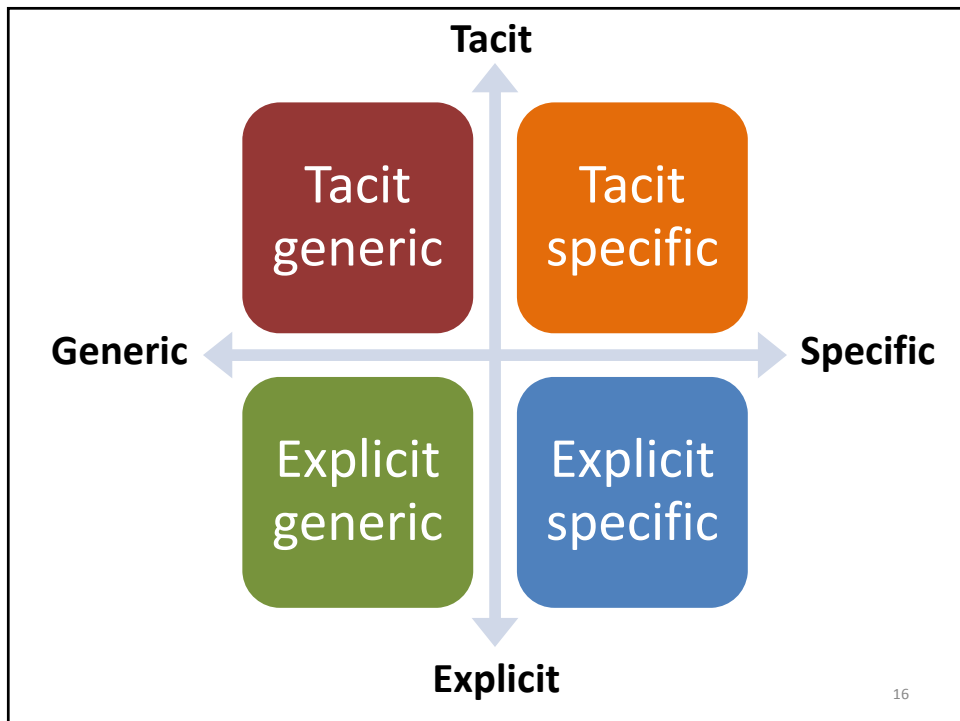
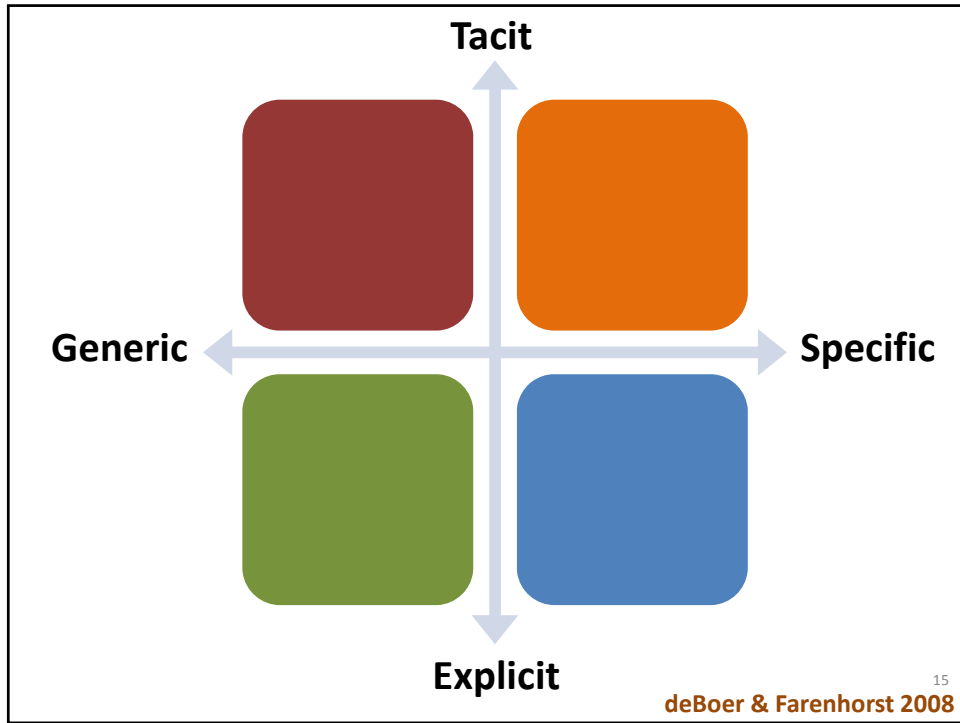


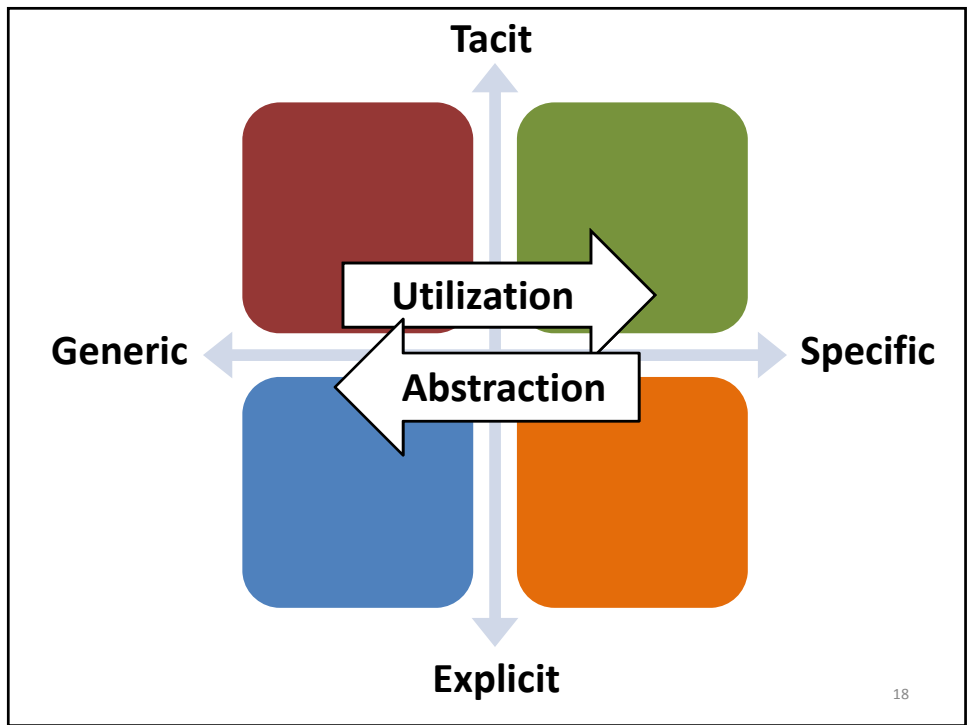
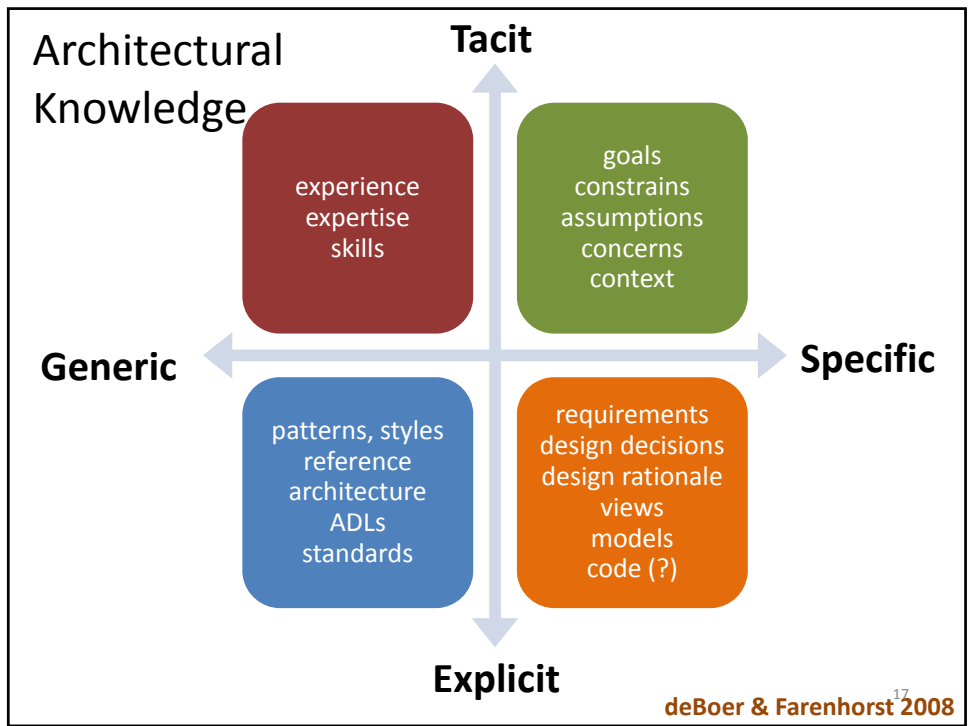
Tacit Knowledge
vs.
Explicit Knowledge

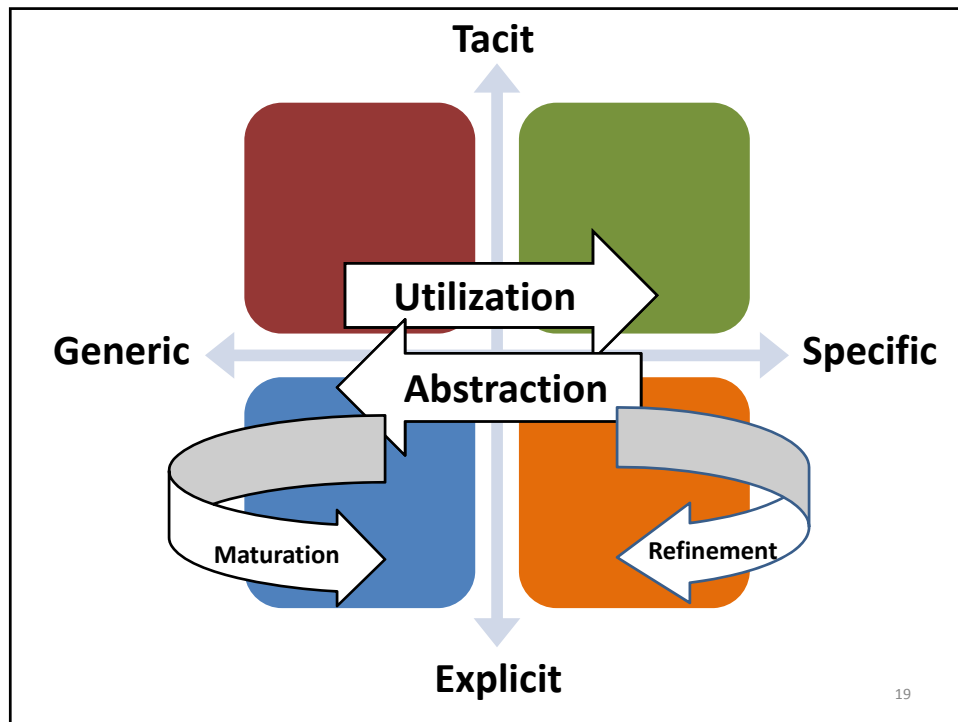


12







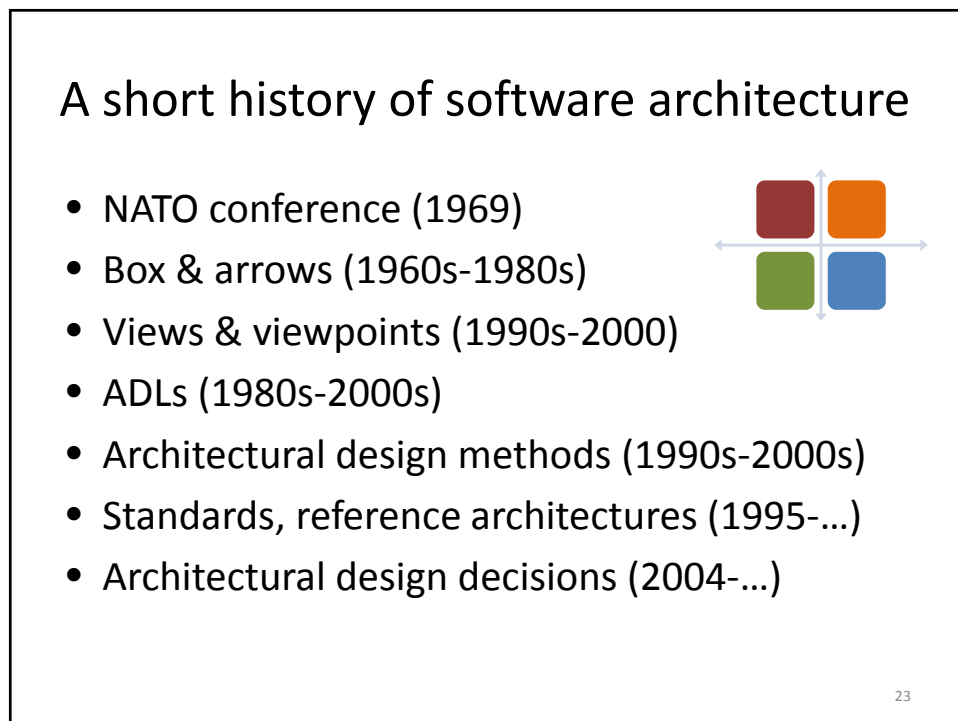
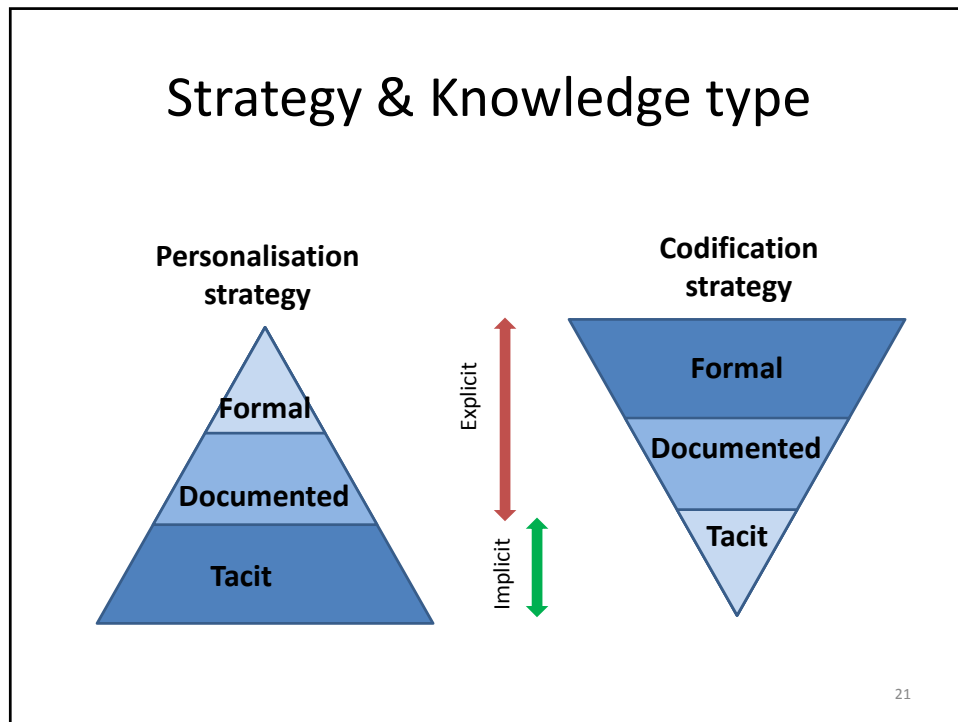


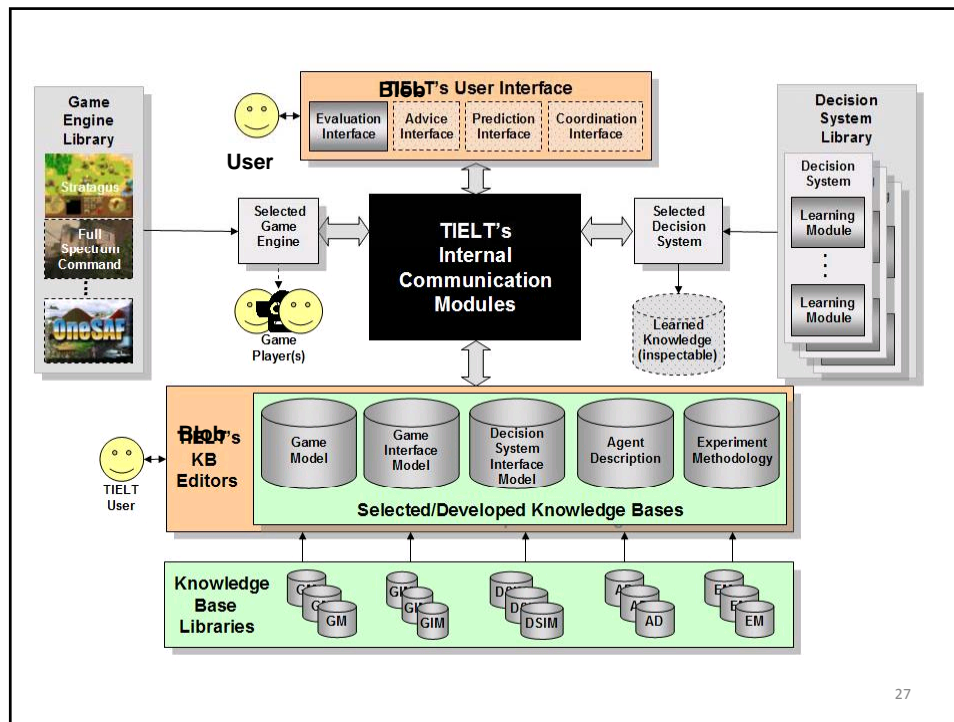
Knowledge management strategies

- Codification
 - Capture information, store it, retrieve it
- Personalisation
 - Define who knows what, “yellow pages”

Hansen et al. 1999

20





Issues

- General “message” or metaphor OK
- Fuzzy semantics:
 - What does a box denote?
 - Function, code, task, process, processor, data
 - What does an arrow denote?
 - Data flow, control flow, semantic dependency, timing
- Diverging interpretation
- Many distinct concerns or issues addressed in one diagram

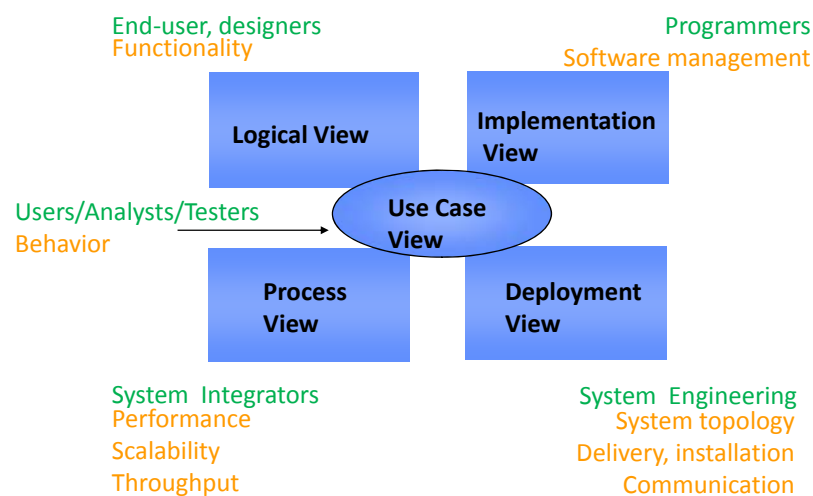
28

Views & Viewpoint

- S4V at Siemens
- BAPO/CAFR at Philips
- IEEE Std 1471:2000 Recommended practice for software architecture description
- ISO/IEC 42010: 2007 Recommended practice for architectural description of software-intensive systems
- ISO/IEC 42010: 2010 (?) Architectural description
- Clements et al. 2005, Documenting Sw Arch
- Rozanski, N., & Woods, E. (2005). *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Boston: Addison-Wesley.

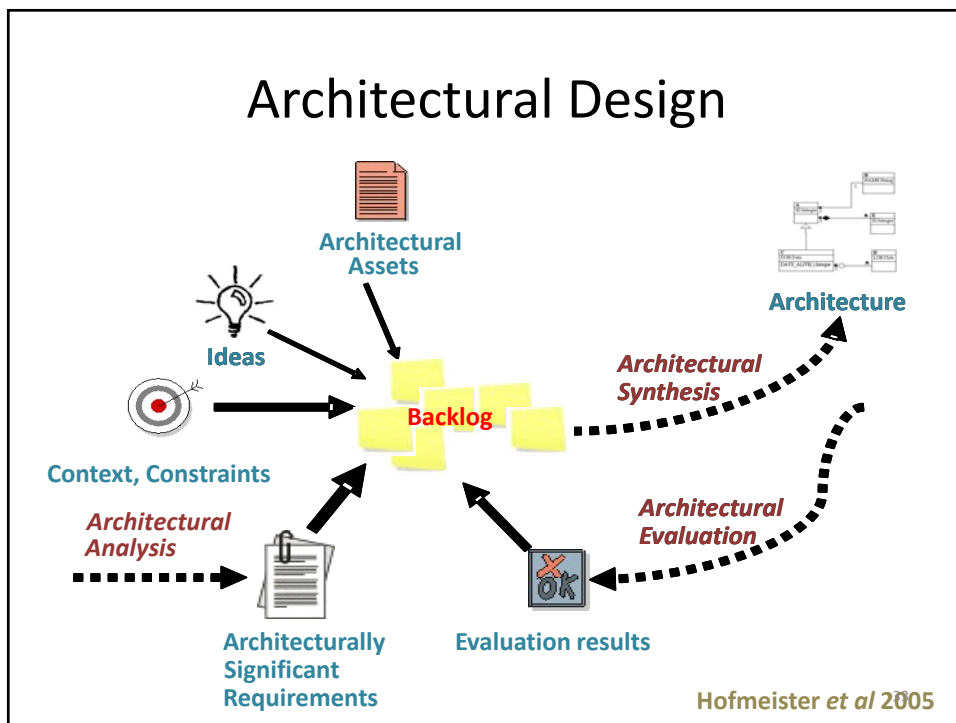
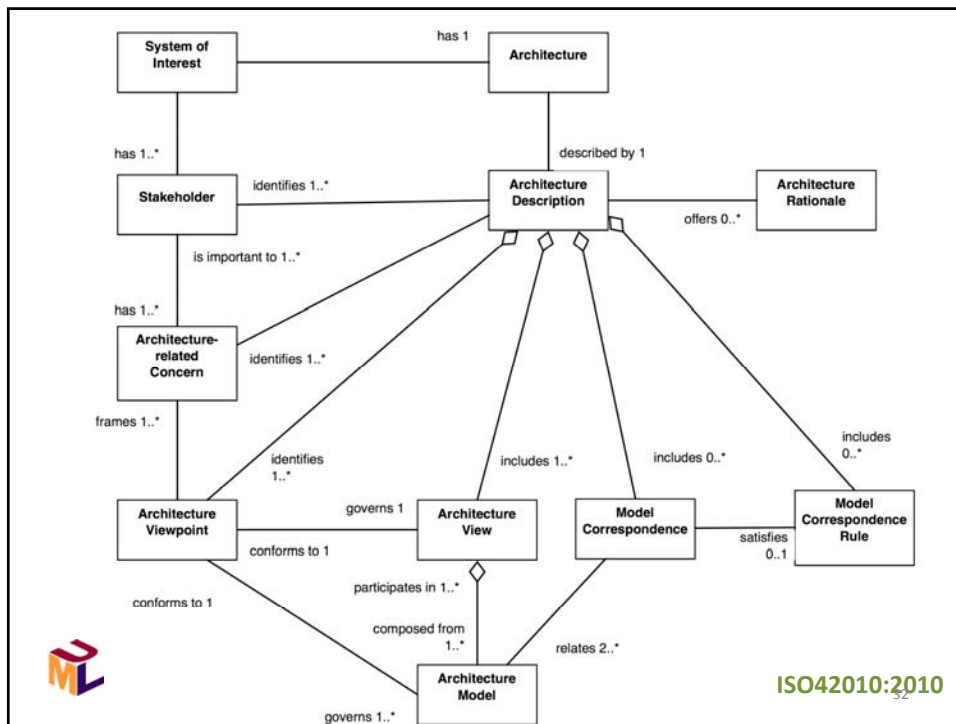
30

The 4+1 view model of architecture



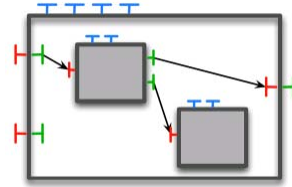
31

Kruchten 1995



Architecture description languages

- Rapide (Stanford)
- ACME (CMU)
- Wright (CMU)
- C2 (UC Irvine)
- Darwin (Imp. Coll.) -> Koala
- Archimate
- AADL (based on metaH)



34

UML 2.0

- A notation
- Better “box and arrows”
- Crisper semantics
- Almost an ADL ?
- Model-driven design,
- Model-driven architecture.



35

Use-Case Diagram

Class Diagram

Statechart Diagram

UNIFIED MODELING LANGUAGE

Collaboration Diagram

Component Diagram

Deployment Diagram

Sequence Diagram

Many kinds of diagrams, but not always very adapted to architecture

36

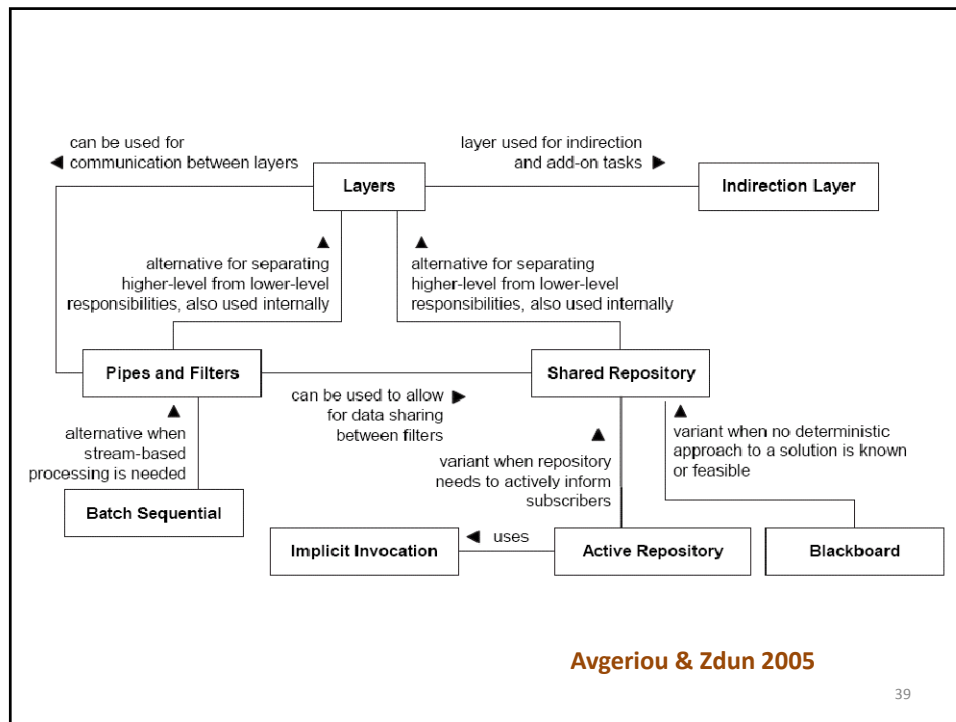
Patterns

- Common solution to a recurring problem...
- Architectural patterns
 - Buschmann F., Meunier R., Rohnert H Sommerlad P. & Stal M. (1996). *Pattern-Oriented Software Architecture: A System of Patterns*. John Wiley & Sons

```

classDiagram
    class KnowledgeSource {
        +updateBlackboard()
        +execCondition()
        +execAction()
    }
    class Blackboard {
        +solutions
        +controlData
        +inspect()
        +update()
    }
    class Control2 {
        +loop()
        +nextSource()
    }
    KnowledgeSource "1" -- "1..*" Blackboard : +operates
    KnowledgeSource "1" -- "1..*" Control2 : +activate
    
```

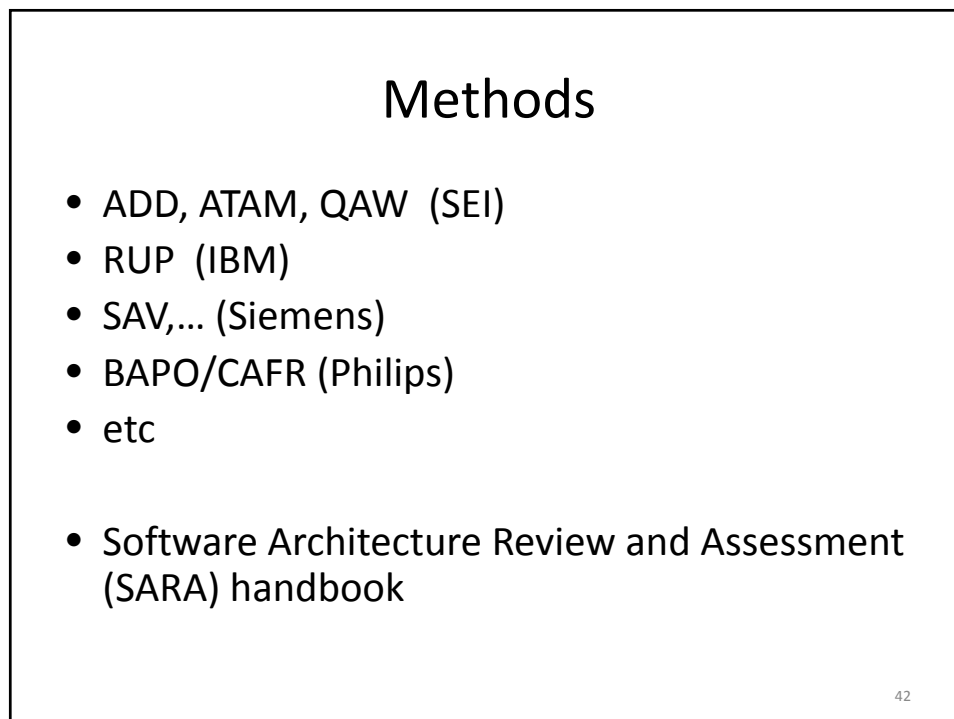
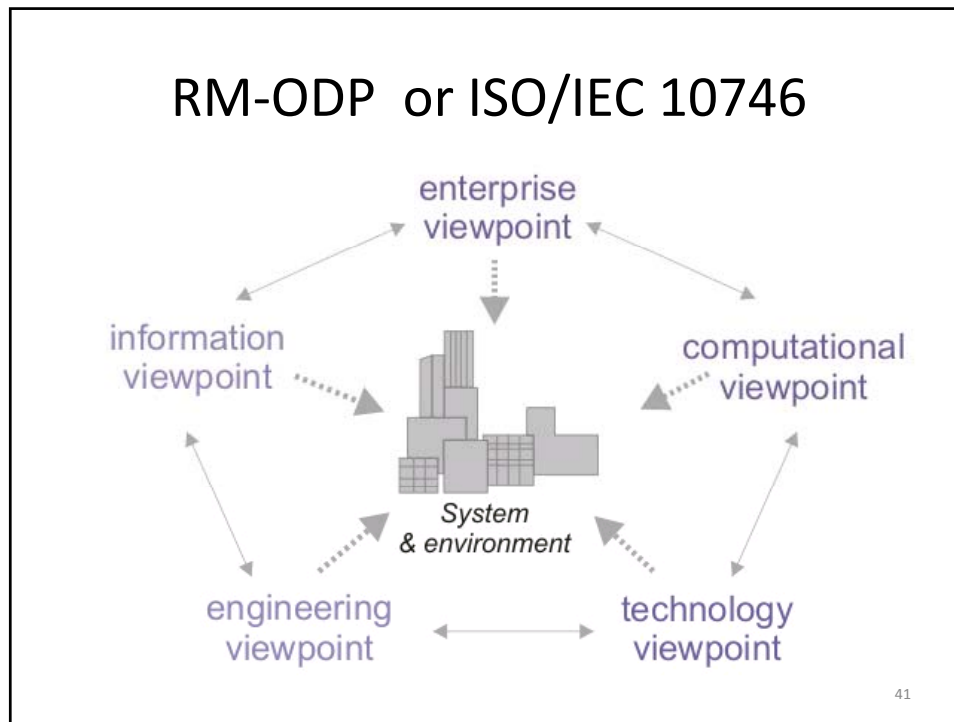
37



Standards, reference architectures

- Codified generic knowledge
- IEEE 1471, ISO 42010 architecture representation
- RM-ODP = ISO 10746
- TOGAF (The Open Group)
- MoDAF, DoDAF, <xyy>AF
- ISO 19439 Framework for enterprise modelling





Metaphors

- Metaphors give meaning to form, help ground our conceptual systems.
- Cognitive transfer: source domain to target domain
 - the <target> is the <source>

*Lakoff and Johnson (1980) **Metaphors we live by***

43

Metaphors

- Ontological metaphors:
 - Clients and servers, layers, pipes and filters, shopping carts
- Structural metaphors
 - Spatial: on top of, parallel to, aligned with, foreground/background
 - Networks, web, hierarchy
 - Containers: packages, repositories, library, volume...

44

Beyond metaphors: Blends

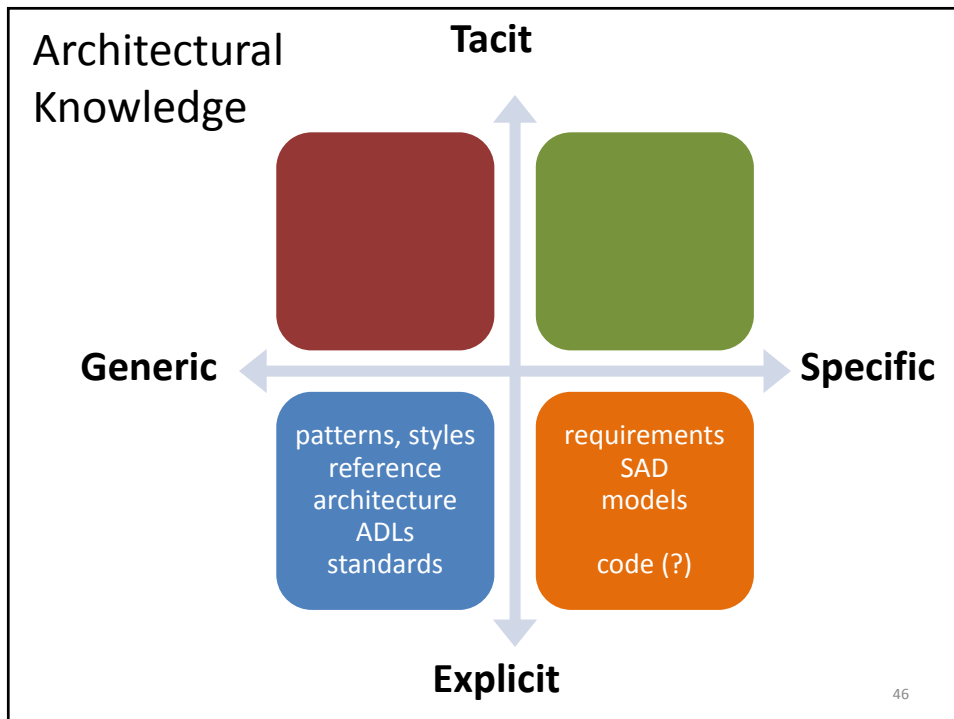
- “Super metaphor” using 2 source spaces.
- Desktop metaphor is actually a blend
 - Computer command
 - Office elements

Imaz & Benyon 2007



“Good news.
The test results show it’s a metaphor.”

45



46

Something's missing

- Software architecture document
- Transferring knowledge
 - To another system
 - To another person
 - To another organization
- Rationale: why?
- Decisions.... ?

circular reasoning works

47

Software Architecture

Software architecture encompasses the set of significant decisions about

- the organization of a software system,
 - the selection of the structural elements and their interfaces by which the system is composed together with their behavior as specified in the collaboration among those elements,
 - the composition of these elements into progressively larger subsystems,
- etc etc etc

*Grady Booch, Philippe Kruchten, Rich Reitman, Kurt Bittner; Rational, circa 1995
(derived from Mary Shaw)*

48





$$AK = AD + DD$$

Architectural Knowledge

=

Architectural Design

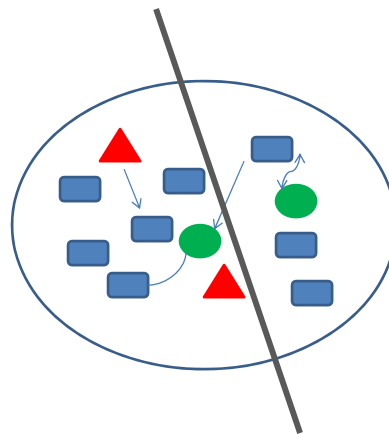
+

Design Decisions

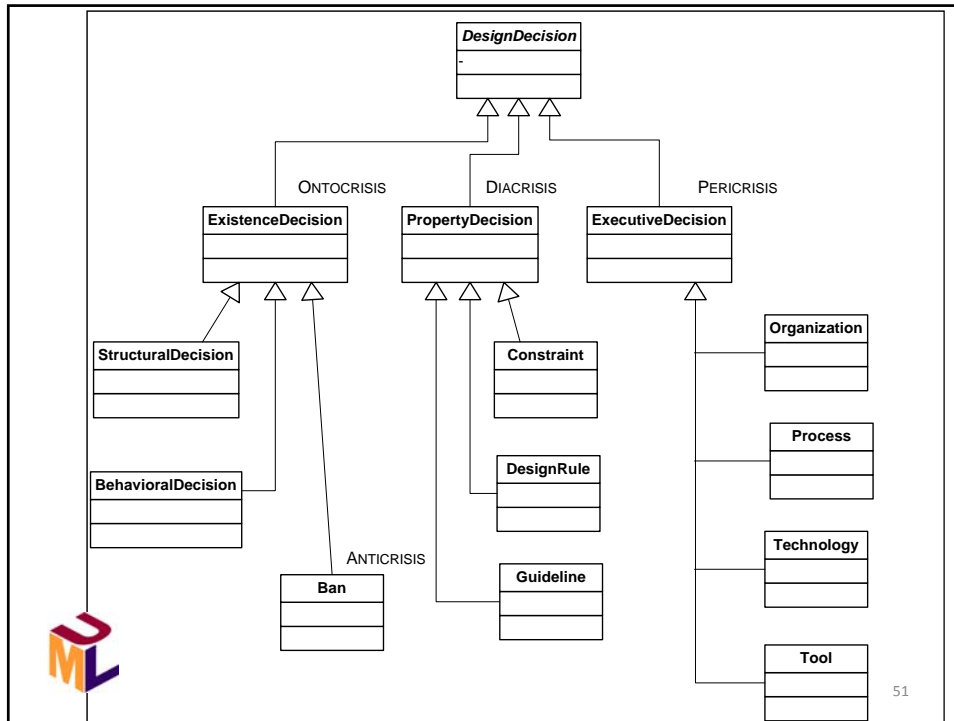
49

Kinds of decisions

- Ontocrises
 - Existence decisions
 - Anticrises
- Diacrisis
 - Property decisions
- Pericrisis
 - Executive decisions



50



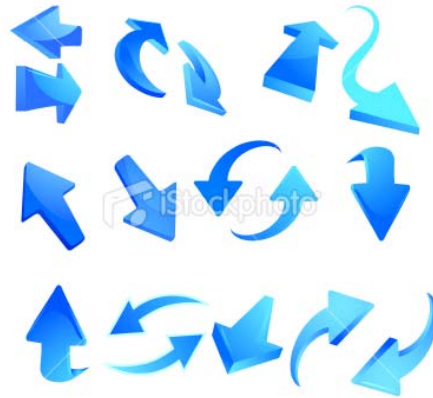
Attributes of a decision

- Epitome Text
- Rationale Text or Pointer
- Scope Text
- State Enumeration
- History List of
(time stamp + author + change)
- Cost Value
- Risk Exposure level

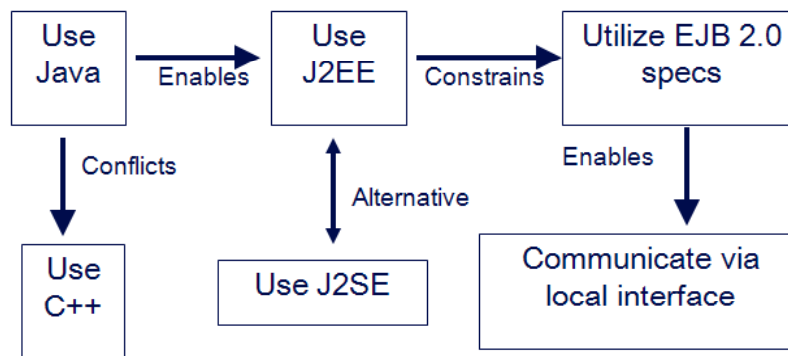


Relationship between decisions

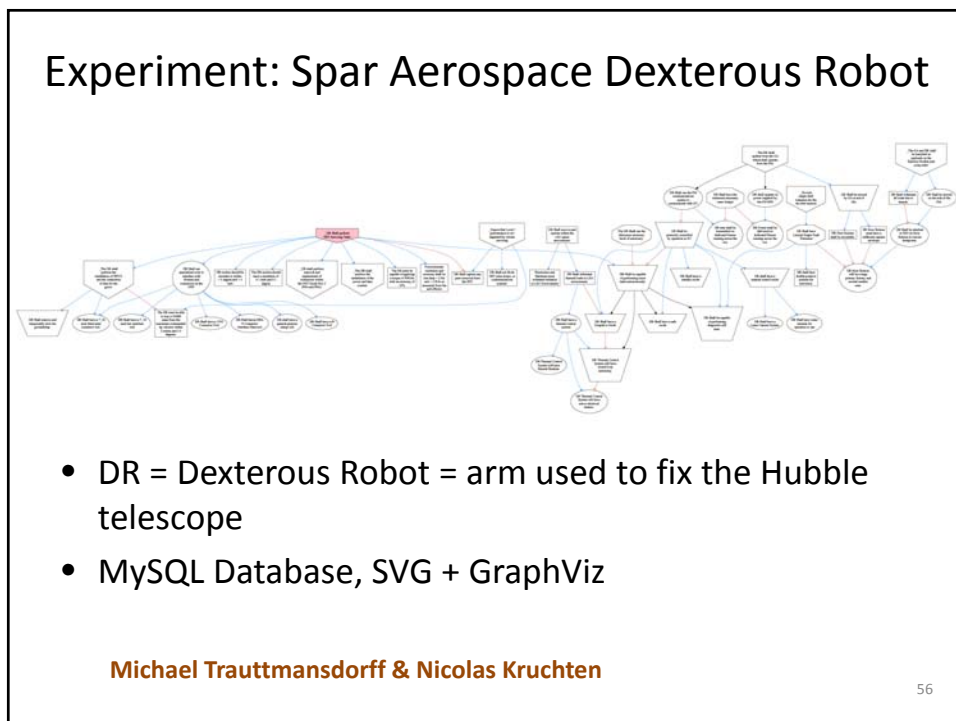
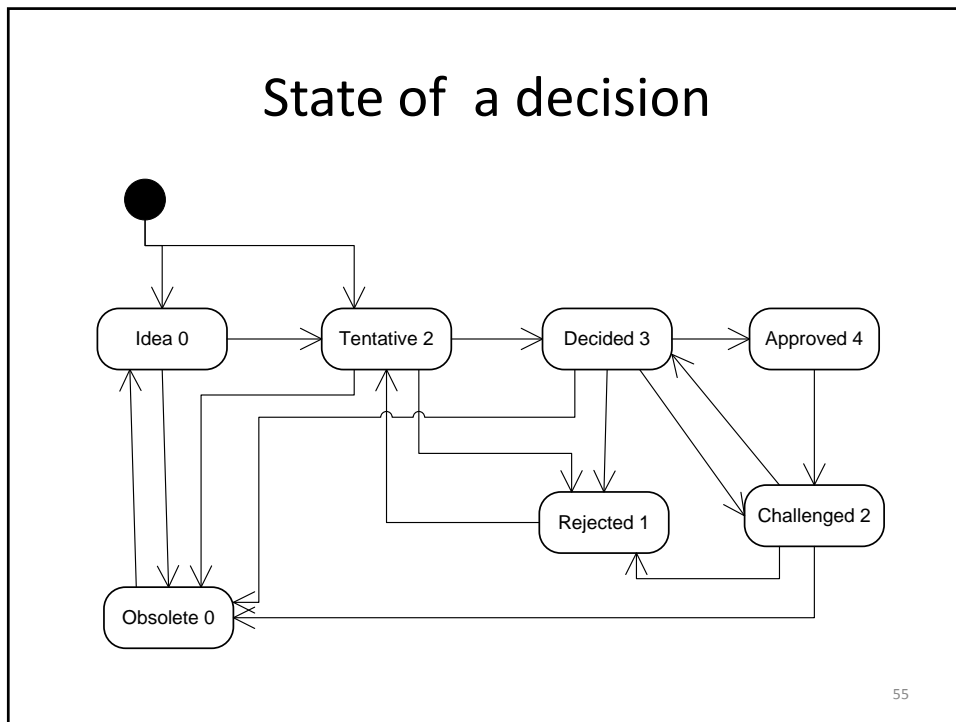
- Constrains
- Forbids
- Enables
- Subsumes
- Conflicts with
- Overrides
- Comprises (is made of)
- Is bound to
- Is an alternative to
- Is related too
- Traces to
- Does not comply with

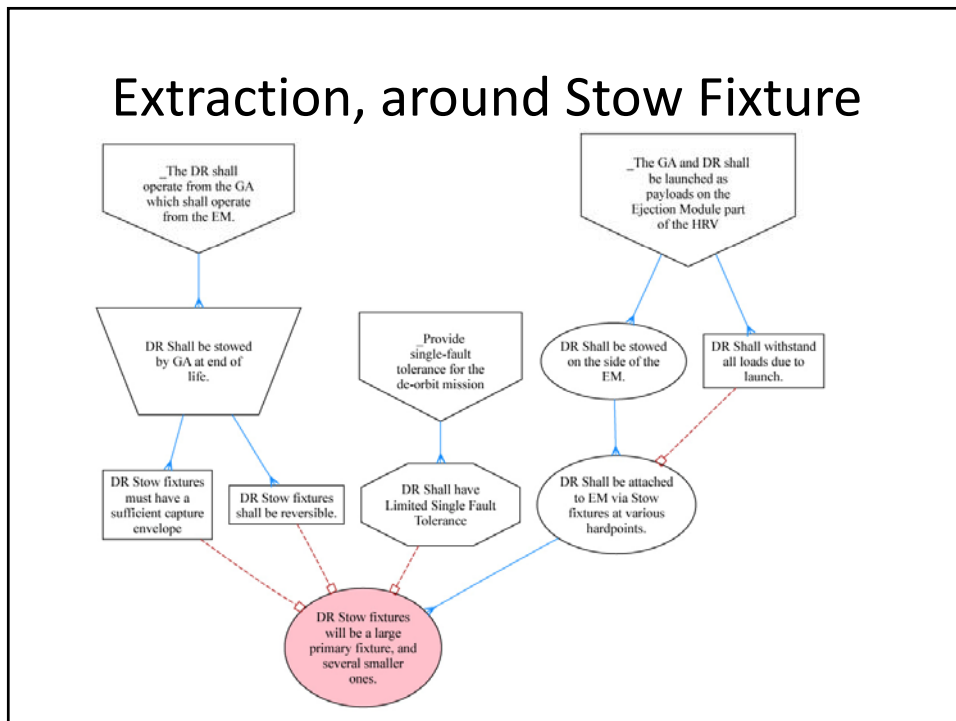
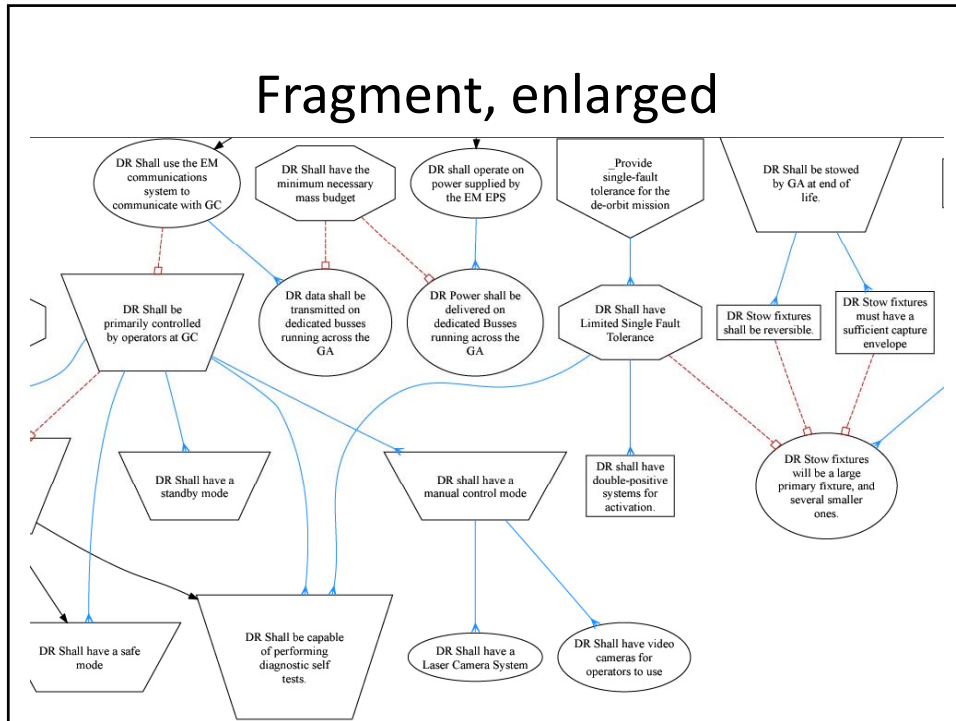


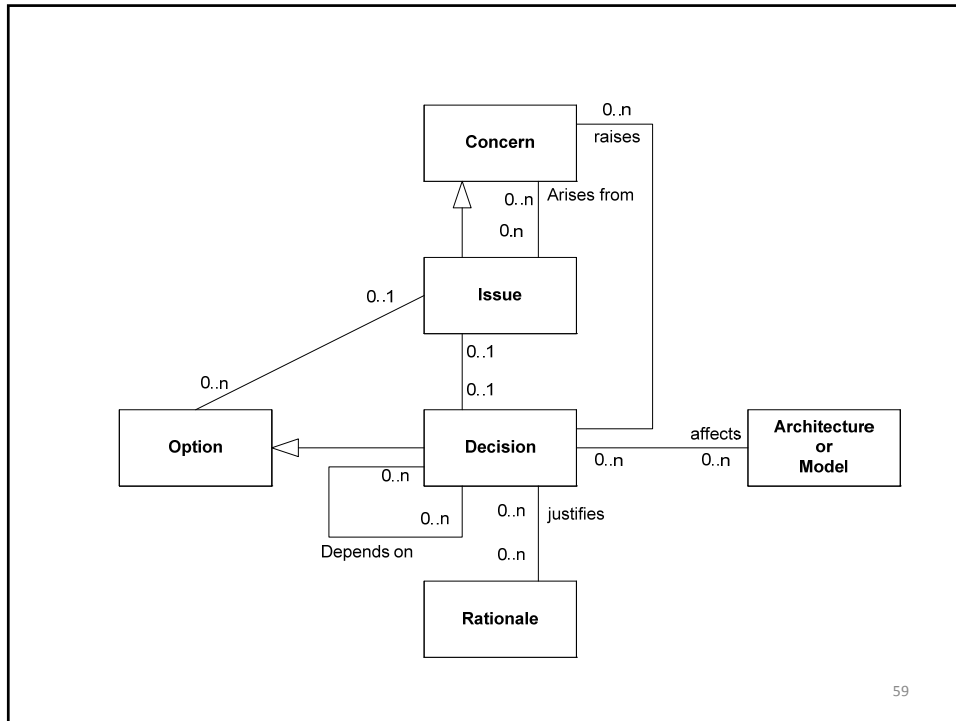
53



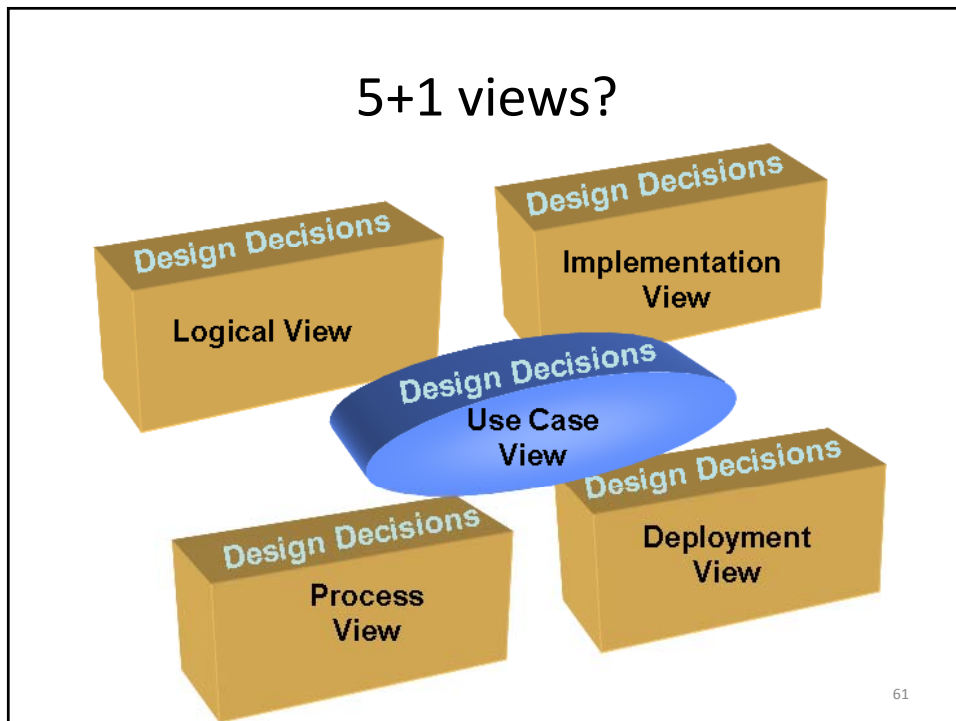
54







59



61

Capturing design decisions

- “Design rationale support systems have failed to gain any level of support in the software industry because of overhead of **capture**”(Jintae Lee)
 - QOC, DRL, InfoRAT, IBIS etc.
 - not enough immediate value, therefore no incentive to capture
 - tedious process, static diagrams

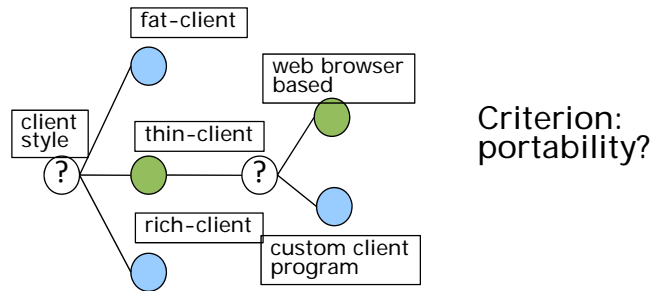
62

Aside: decision support, decisions process

- Many of these systems were design to support a rational decision process
 - Establish issue
 - Enumerate alternatives, find Pros & Cons
 - Rank, prioritize
 - Chose
 - Document reason of choice
- In reality: gut feeling, and first idea to mind
 - Then rationalization, or scrap and rework

63

Rational choice



64

Tackling capture differently...?

- Automating capture with daemons (agents)
 - Instrument the source of decision:
 - Design tool
 - Requirement management tool
 - Defect tracking tool
 - Configuration and change management tool
 - Management tool (task allocation, issue/action items)
- Waypointing
- Capture now, sort out later

65

Tools for Architectural Knowledge Management

- Codification or personalization or both?
- The myth of the “central repository”
 - Bureaucratic school
- The myth of the additional tool to solve a new problem
 - Tool vendors and/or grad students
- “Feeding the beast”
- Time shift: Production – Need
 - no incentive, no ‘stickyness’

66

Tools

- For whom?
 - Architect
 - Reviewers, auditors
 - Requirement eng., analysts
 - Maintainers
- To do what?
 - Producing AK
 - Retrieving AK
 - Assessing architecture
 - Making decisions
 - Educating others
 -



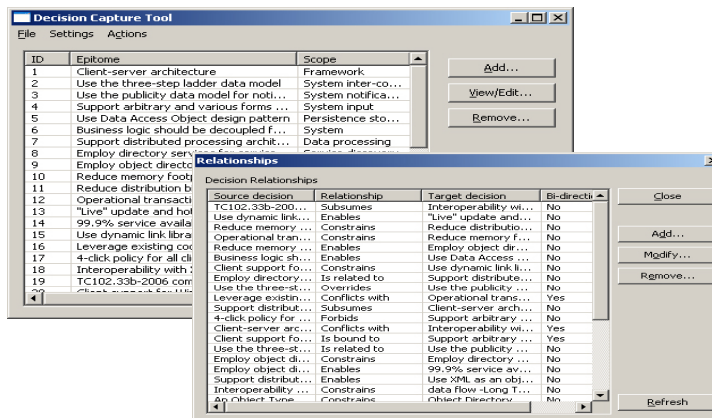
67

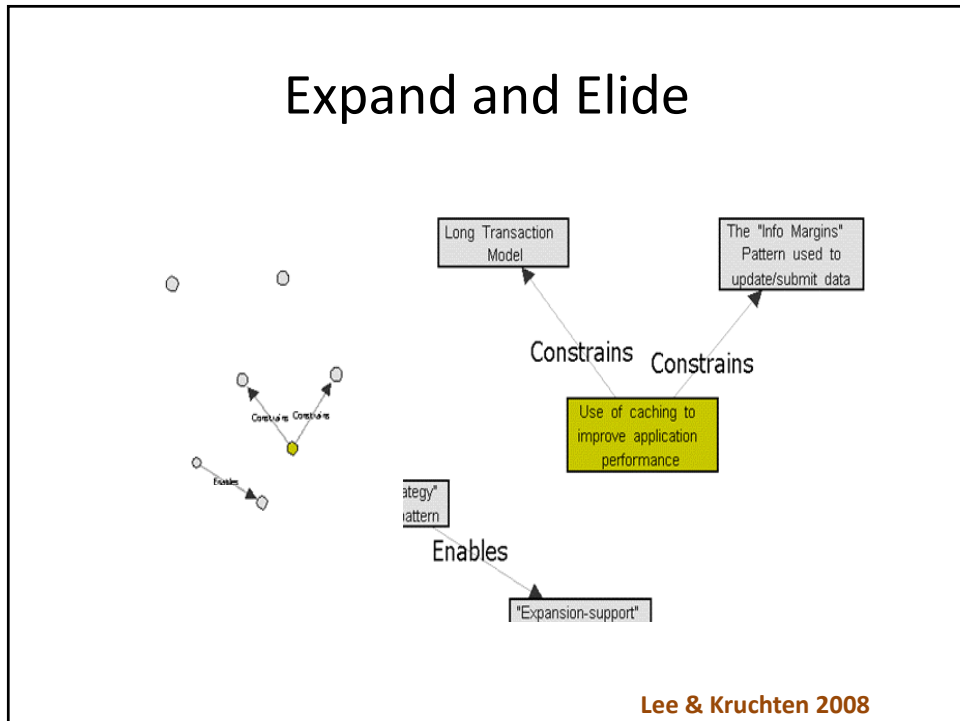
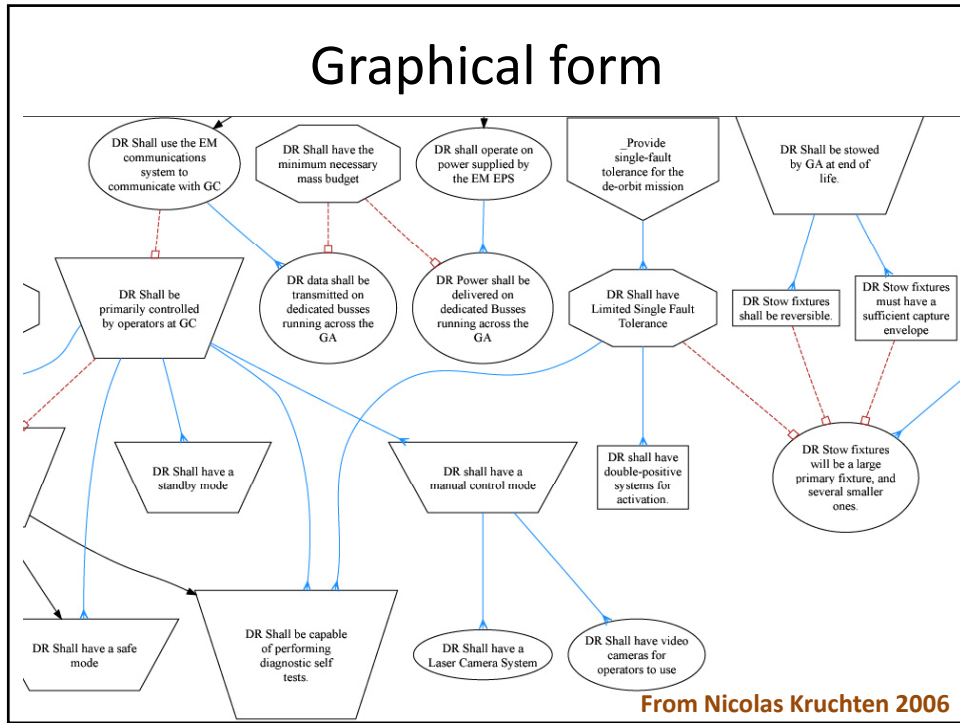
Tools

- ADDS, Rafael Capila
- Archium, Jansen & Bosch
- AREL, Tony Tang
- Knowledge architect
- IBM's Architect's Workbench
- SEURAT, Burge

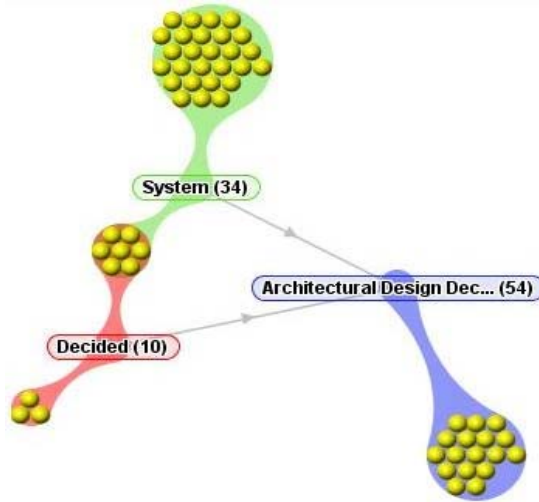
68

Tabular form



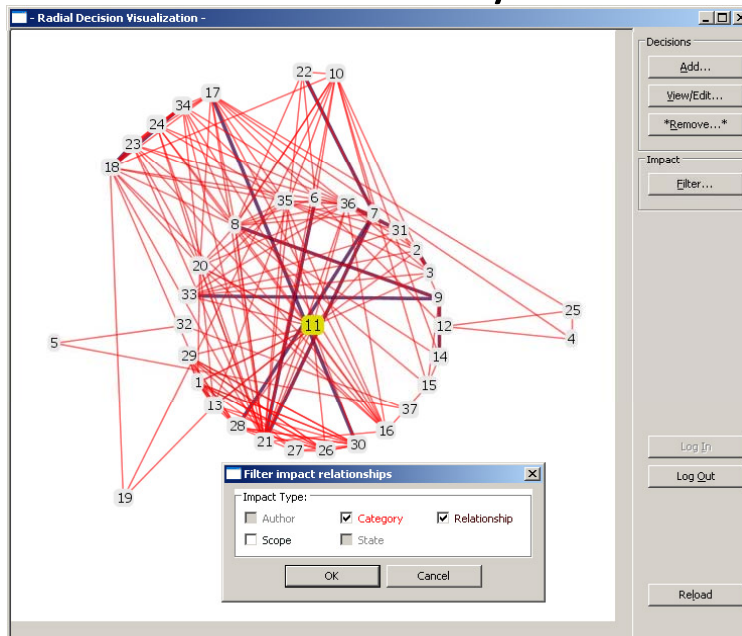


Clustering (with Aduna)



Kruchten et al 2005

What if analysis



Personalisation (& hybrid)

- Knowledge sharing networks
- WIKIs
- Web 2.0

- Semantic web, semantic wikis...
- EAGLE at VU
- PAKME

74

PAKME

- M. Ali Babar, LERO, Limerick Ireland

75

Using a Design Decision Case

The screenshot shows a web-based interface for searching and listing design options. On the left, there are search criteria fields and a 'Display Results and Sort by' dropdown set to 'All Design Opt'. Below this, a table lists design options with columns for Name, Description, and actions like 'Modify' and 'Delete'. The 'Application Server' option is highlighted. On the right, a detailed view of the 'Application Server' option is shown, including its description and a 'Number of Results found from search: 15'.

Design Option Name	Description	Count	Filter	Priority	Percentage	Actions
Database Server	Introduce a dedicated server as a database service provider. This reduces the workloads on other systems and offers a centralized database. [more...]	1	BCS Project	research	100 %	Modify
Application Server	Have a dedicated Application Server to provide application service to the clients. Hence reduces the workload to other parts of the system. [more...]	0	None	None	100 %	Modify
Multiple Server System	Introduce different servers to provide different services for the client. Hence would greatly reduce the workload the current servers. [more...]	0	None	None	100 %	Modify

Date 10/7/2009

78

Navigating the Knowledge Base

The screenshot displays a multi-pane interface for navigating a knowledge base. The left pane shows search filters for 'Advanced Search' and 'Pattern Search'. The middle pane shows a list of results under 'Architecturally Significant Requirement Listing'. The right pane shows a detailed view of a 'Business Delegate' pattern, including its description, purpose, and related information.

Date 10/7/2009

79

Template for Capturing and Representing Patterns

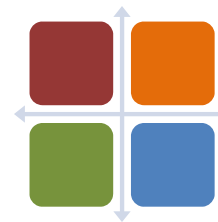
View Pattern	
Name	Business Delegate
Type	Design pattern
Description	This pattern reduces coupling between tiers and provides an entry point for accessing the services that are provided by another tier. It may also provide results caching for common requests to improve performance. It typically uses a Service Locator to locate a service.
Context	In a distributed system, clients may be exposed to the complexity of dealing with the distributed components that provide services.
Problem	Presentation-tier components interact directly with business services, which exposes the implementation details of the services to the clients. Such a direct interaction makes the clients vulnerable to any changes in the business services.
Solution	Use Business Delegate to reduce coupling between presentation-tier clients and business services. The Business Delegate hides the underlying implementation details of the business service.
Parent	<i>No Parent Available</i>
Forces	1) Business Service
Tactics	1) Delegate Proxy 2) Delegate Adapter
Affected Attributes	<i>Positively</i>
	1) Performance
	<i>Negatively</i>
	1) Complexity 2) Introduce new layer
General Scenario	1) BD-S6 2) BD-S2
Usage Examples	1) E-Commerce

Date 10/7/2009

80

Summary

- Architecture is more than just the resulting design of architecting
- Tacit, explicit knowledge
- Generic, specific knowledge
- Codification, personalisation
- Power of metaphors
- Decisions as first class citizen
- Tool support (need more work)



82

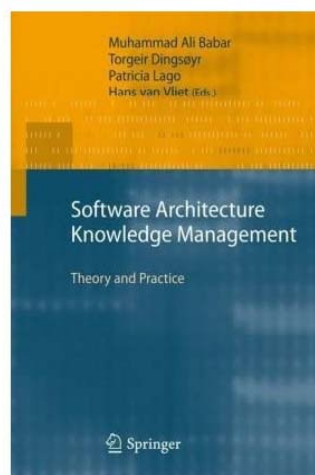
Slides at: pkruchten.wordpress.com/talks/



Questions?

83

Shameless self-promotion



M. Ali Babar, T. Dingsøyr, P. Lago, H. van Vliet, *Software Architecture Knowledge Management*, Springer Verlag, 2009

I wrote chapter 3 !



References (1)

- Avgeriou, P., & Zdun, U. (2005). Architectural patterns revisited: a pattern language. Paper presented at the 10th European Conference on Pattern Languages of Programs (EuroPlop 2005), Irsee, Germany.
- Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice (2nd ed.)*. Reading, MA: Addison-Wesley.
- Buschmann F., Meunier R., Rohnert H. & Sommerlad P. & Stal M. (1996). *Pattern-Oriented Software Architecture: A System of Patterns*. John Wiley & Sons.
- Hofmeister, C., Kruchten, P., Nord, R., Obbink, H., Ran, A., & America, P. (2007). A General Model of Software Architecture Design derived from Five Industrial Approaches. *Journal of Systems & Software*, 80(1), 106-126.
- Imaz, M., & Benyon, D. (2007). *Designing with blends: conceptual foundations of human-computer interaction and software engineering*. Cambridge, MA: The MIT Press.
- Kruchten, P., Capilla, R., & Dueñas, J. C. (2009). The role of a decisions view in software architecture practice. *IEEE Software*, 26(2).
- Kruchten, P. (1995). The 4+1 View Model of Architecture. *IEEE Software*, 12(6), 45-50.



85

References (2)

- Kruchten, P. (2004, December 3-4). An Ontology of Architectural Design Decisions. Paper presented at the 2nd Groningen Workshop on Software Variability Management, Groningen, NL.
- Kruchten, P. (2009). Documentation of Software Architecture from a Knowledge Management Perspective--Design representation. In: M. Ali Babar, T. Dingsøyr, P. Lago & H. van Vliet (Eds.), *Software Architecture Knowledge Management: Theory and Practice* (pp. 29-58). Berlin: Springer-Verlag.
- Lakoff, G., & Johnson, M. (1980). *Metaphors we live by*. Chicago: The University of Chicago Press.
- Nonaka, I., & Takeuchi, H. (1995). *The knowledge creating company: how Japanese companies create the dynamics of innovation*. New York: Oxford University Press.
- Perry, D. E., & Wolf, A. L. (1992). Foundations for the Study of Software Architecture. *ACM Software Engineering Notes*, 17(4), 40-52.



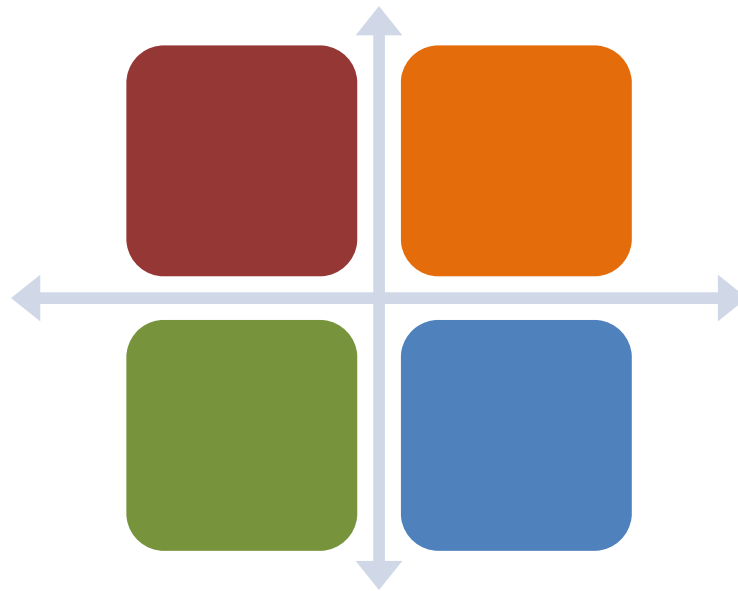
86

References (3)

- Rozanski, N., & Woods, E. (2005). *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Boston: Addison-Wesley.
- Vitruvius Pollio, M. (25 B.C.). *De Architectura*.



87



88