THE UNIVERSITY OF BRITISH COLUMBIA

Software Architecture: Perspectives on a Maturing Discipline

Philippe Kruchten

Roma, June 19-20, 2007

ITALIA

1

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

Professor of Software 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

Cofounder IFIP WG2.10 Software architecture
Assoc. Editor of *IEEE Software* for architecture and design

2

Mary Shaw & David Garlan (1996).
Software Architecture: Perspectives on an Emerging Discipline.
Upper Saddle River: Prentice-Hall.

3

Outline

- Part 1
  - A brief history of software architecture
  - Software architecture in one easy lesson
    - What, who, how, when, why
- Break
- Part 2:
  - Architectural knowledge
  - Architectural decisions

4

THE UNIVERSITY OF BRITISH COLUMBIA

Part 1: Software Architecture

5

A Brief History of Software Architecture (1/3)

- Before 1995:
  - 1969 NATO Conf
  - Conceptualization:
    - Brooks
    - Modules [Parnas]
  - Architecture = {Elements, Forms, Rationale} [Perry & Wolf]
  - ADLs – Components and Connectors
  - Styles [Shaw & Garlan]

6

A Brief History of Software Architecture (2/3)

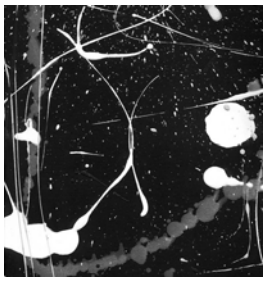
- 1995-2005
  - Architectural descriptions: views and beyond
  - Architectural methods
  - More ADLs
  - Many books (SEI, Clements and co.)
  - IEEE 1471-2000 on Architectural description
  - ISO RM-ODP (X.901-904)
  - Software Product Lines (SPL)
    - Variability

A Brief History of Software Architecture (3/3)

- 2005 – present
  - Broadening of the concepts (system, enterprise)
  - Quality of software architecture
  - Architectural decisions
  - Architectural knowledge
    - Capture, transfer
  - Architectural competence
    - Education
  - Aspects

Software Architecture

- What … is it ?
- Who … does it ?
- When … does it appear ?
- How … to do it ?
- Why … bothering at all ?



What is software architecture?

- Architecture = Structure ?
- Architecture = Design ?
- Architecture = Infrastructure ?
- System architecture vs. Software Architecture
- Enterprise Architecture ??

Software Architecture: A Definition

Software architecture encompasses the 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)

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.

RUP 1998

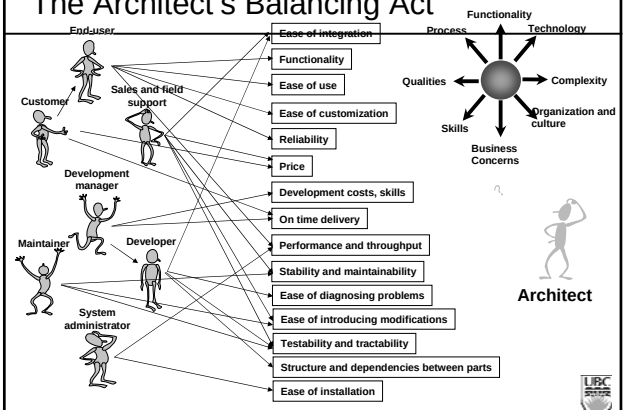
So software architecture…

- … is a part of Design
  But not all design is architecture
  … *which part of design, then?*
- … includes Structure, and much more
  Behaviour, style, tools & language
- … includes Infrastructure, and much more
- … is part of System architecture

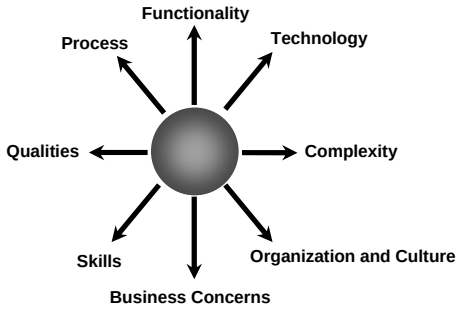
All systems have an architecture

- How much effort should you put into it varies greatly
- 75%: architecture is implicit
  - Choice of technology, platform
  - Need to understand the architecture
- Novel systems:
  - Much more effort in creating and validating an architecture
- Key drivers are mostly non-functional:
  - Capacity, performance, availability, evolvability, security, …

The Architect's Balancing Act



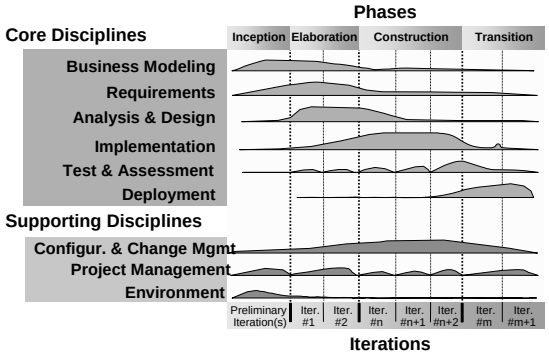
Tradeoffs Between Forces



Architecture is about making decisions

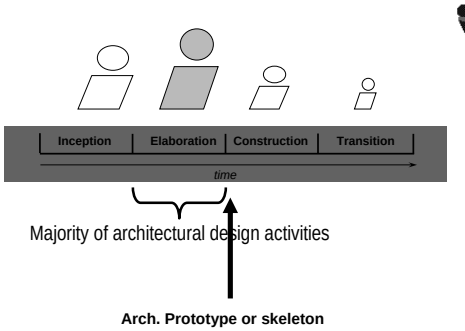
The life of a software architect is a long (and sometimes painful) succession of suboptimal decisions made partly in the dark.

Rational Unified Process® (RUP®)



Source: IBM, 2003

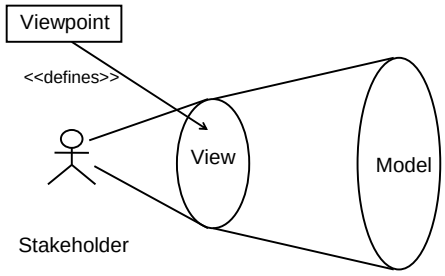
Architectural Effort During the Lifecycle



Representation of architecture

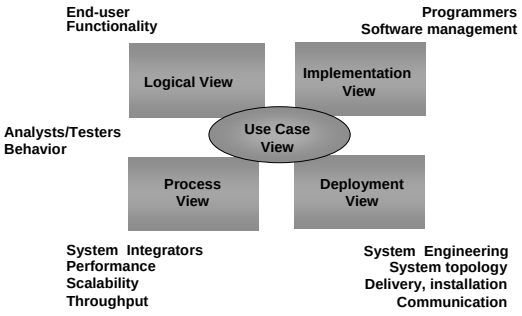
- Software Architecture Document
  - Multiple views
- Architectural prototype, or skeleton
  - Result of early iteration
    - RUP's elaboration phase
  - Or almost "ready made" (out of the box)

Of Views, Viewpoints and Models



Views are projections of a model for a particular stakeholder

RUP's 4+1 Views



Source: Kruchten 1995

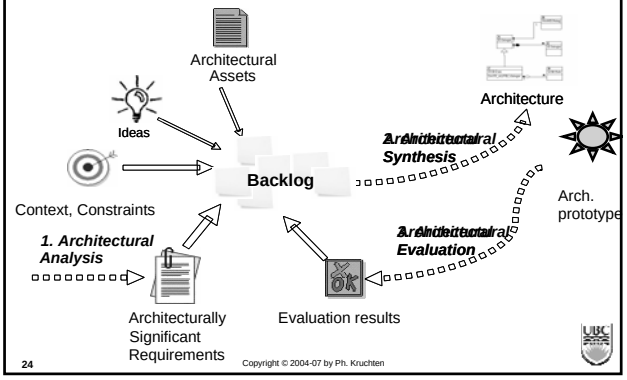
Architectural methods

Source: Hofmeister et al 2005

- SEI: SAAM, ATAM
  - Clements, Bass, Kazman, Barbacci
- Philips: BAPO + CAFCR
  - Obbink, America, …
- Siemens: S4V
  - Nord, Hofmeister, …
- Nokia: ASC
  - Ran…
- IBM Rational: RUP
  - Kruchten, Booch, Wood, Eeles, …

General Workflow

Source: Hofmeister et al 2005



VRAPS: Dealing with hidden realities

- Vision
  - Provide a coherent, compelling picture of the future to all major stakeholders (in and out)
- Rhythm
  - Iteration, "tempo", keep things moving, not analysis-paralysis
- Anticipation
  - Link to risk mitigation: business risk, long-term
- Partnering
  - Cooperation, teamwork, get buy-in of all parties
- Simplification
  - Both at the design level and at the organization level

Source: Dikel et al

Benefits of an Architectural Focus

- Support Project & Product Management
  - Partitioning between teams (large system)
  - Buy vs. build: contracting, outsourcing
  - Reuse, product line
- Communication
  - Between all stakeholders, across teams
- Stability, predictability
- Separate:
  - Long-term, strategic, global decisions
  - Short-term, tactical, local decisions
- Documentation

Software Architecture: A Basis…

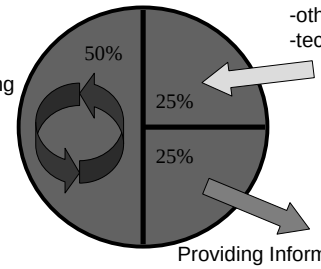
- for Component-base development
- for a reuse policy and strategy
- for Software Product Line
- for COTS-Based development

Software Architecture and Agile Methods

- XP: Metaphor
- Scrum, DSDM, Lean, FDD
- *"No BUFD !", "YAGNI !"*
- They assume relatively stable architecture
  - Go straight into RUP construction phase
- Architecture will not emerge by simple successions of "refactoring"
- Complex, large or novel systems may "hit a wall"

What do architects actually do?

Architecting:
-design
-validation
-prototyping
-documenting
-etc….



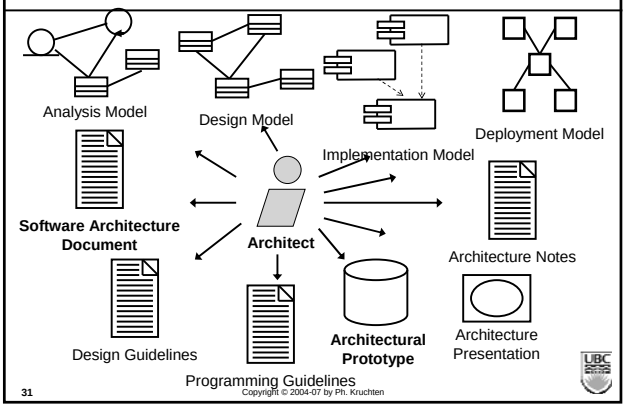
Getting input:
-user, requirement
-other architecture
-technology

Providing Information
-communicating architecture
-assisting other stakeholders

Software Architect (or Team) Charter

- Defining the architecture of the software
- Maintaining the architectural integrity of the software
- Assessing technical risks related to the software design
- Proposing the order and contents of the successive iterations
- Consulting services
- Assisting marketing for future product definition
- Facilitating communications between project teams

Architectural Artifacts



Research on software architecture

- From Architectural Design
  to Architectural Design Decisions
- Architectural Knowledge =
  Architectural Design + Architectural Decisions
- Decisions = making a binding choice:
  - Relationships
  - Rationale
  - Tacit knowledge

More research directions

- ADL = Architecture Description Languages
- ADN = Architectural Design Notation
- SPL: Software Product Lines
  - A common architecture, several products
- Model Driven Architecture / Design
- Service-Oriented Architecture
- Aspect-oriented architecture

Summary

- Architecture is…
  - a subset of design
  - more than just structure
  - the place of resolution of major tensions
  - the concern of many stakeholders

Summary

- All systems have an architecture
- Project manager understands the role of architecture and exploits it:
  - to define an organization
  - to drive early phases (inception, elaboration)
  - to mitigate most dramatic technical risks
  - to define a rhythm for iterative development
  - to articulate reuse
  - to instill a common culture
- Architecture team, for large projects, or a family
- Project manager <–> Chief architect

Break

THE UNIVERSITY OF BRITISH COLUMBIA

Part 2: Architectural Knowledge and Design Decisions

37

Software Architecture in RUP

Software architecture encompasses the 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,

38

Software Architecture in RUP (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.

39

Architectural decisions as 1st class elements

- Software architecture encompasses the set of significant decisions about the organization of a software system

  …

- Design = Making choices
  - Leonardo da Vinci : disegno
    Giambattista Vico : ingenio
  - in French: un Dessin avec un Dessein
    = A drawing with an intent

40

Getting at the architecture

- Architecture document
  - Views, UML, text, PowerPoint slides, xP's Metaphors
- Reverse engineering

- But the decisions only partially emerge
- Part of architectural knowledge is lost

Architectural Knowledge =
Design Decisions + Design

41

Modeling the decisions

- Not the design
- But the decisions

For what purpose?

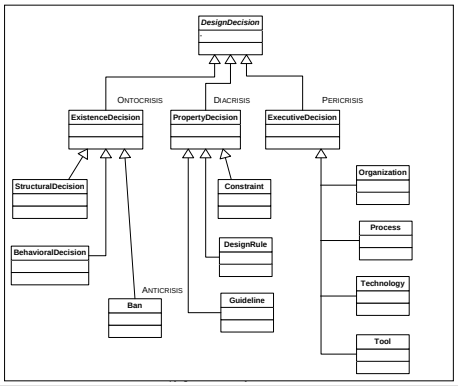
- Rationale
- What if
- Cross-cutting concerns
- Political and organizational issues

42

Types of architectural design decisions

- Ontocrises: existential decisions
  - state what element will exist in the design
  - also anticrises (ban, things that will not exist)
- Diacrises: property decisions
  - associate predicate to set of elements
  - Cross cutting concerns, “aspects”
- Pericrises: executive decisions
  - constrain the above, the organization, etc.

Taxonomy of decisions



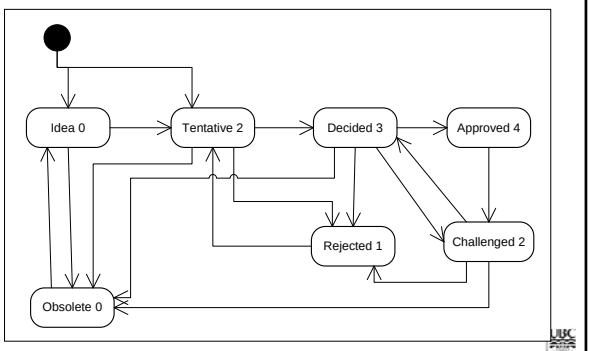
Attributes of a decision

- 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

States of decision



What is the use of a decision graph

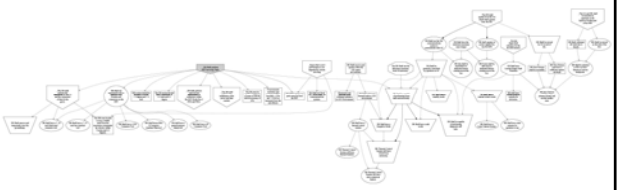
- Rationale
  - Connection between requirements, defect and implementation
- Support for review
  - Completeness, consistency
  - Incremental reviews
- Support for evolution
  - Change analysis, “what if”
- Support for reuse, for learning the architectural process itself, for extracting patterns of decisions

Capturing design decisions

- Design Rationale support systems failed because of overhead of capture
  - not enough immediate value, no incentive to capture
  - tedious process, static diagrams
  - QOC, DRL, InfoRAT, IBIS etc.
- Automating capture with daemons (agents)
  - Instrument source of decision:
    - Design tool
    - Requirement management tool
    - Defect tracking tool
    - Configuration and change management tool
    - Management tool (task allocation, issue/action items)

49

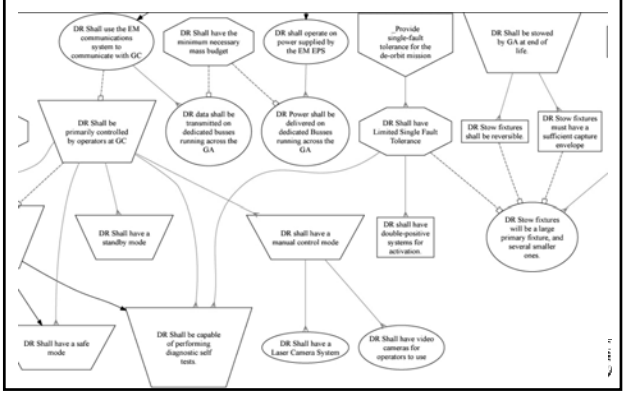
Experiment: Spar Aerospace Dexterous Robot



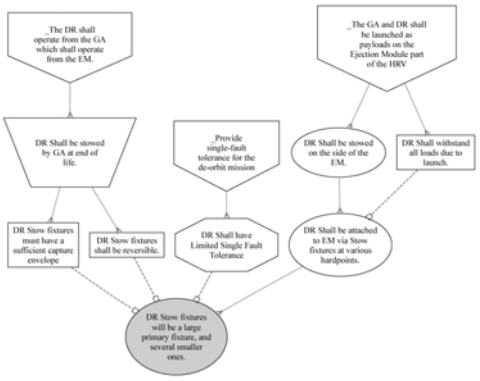
- Dexterous Robot = arm used to fix the Hubble telescope
- Michael Trauttmansdorff & Nicolas Kruchten
- MySQL Database, SVG + GraphViz

50

Fragment, enlarged

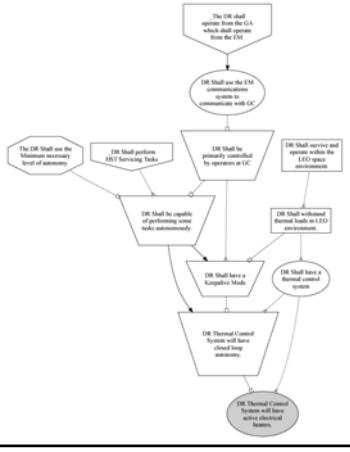


Extraction, around Stow Fixture



52

Heaters



53

Difficulties

- Design rationale: research peaked in the late 1997
  - hard to do consistently; tedious capture
  - no immediate benefits for architects
  - not very useful for software
- Focus shifts
  - from Rationale and Decision Support,
  - to Decision and connections between decisions

  (rationale becomes secondary)

54

Software architecture?

- May apply to wider area than just software architecture
- Systems engineering

Future work

- Refine model
  - Develop more case studies
    - Raytheon: air traffic control,
    - CapitalOne: Banking…
    - Genomics tools at BC Genome science corporation
  - Develop mathematical model, find patterns
- Investigate visualization
  - Icons, arrows, arrowhead, color, position
  - Automatic layout
- Design decision capture mechanism
  - Aim at Eclipse platform: EMF and Rational tool suite

Thanks

- Hans van Vliet and Patricia Lago from the Vrije Universiteit in Amsterdam
- Davide Falessi, U. di Roma Tor Vergata
- Larix Lee (Master student at UBC)
- Shark community (Sharing Architectural Knowledge)
- IBM Canada

Resources

Web sites, portals
- SEI's Software Architecture for Software-Intensive Systems http://www.sei.cmu.edu/architecture/
- The Gaudí System Architecting page http://www.gaudisite.nl/
- The Bredemeyer Architecture page http://www.bredemeyer.com/
- Software Product Lines http://softwareproductlines.com/
- SoftwareArchitectures.com http://www.softwarearchitectures.com
- Grady Booch's Handbook of Software Architecture www.booch.com/architecture/index.jsp

Conferences
- WICSA http://www.softwarearchitectureportal.org/WICSA/conferences/
- EWSA http://www.arch-ware.org/ewsa
- SPLC http://softwareproductlines.com/

Associations and working groups
- IFIP WG 2.10 Software Architecture
- http://www.softwarearchitectureportal.org
- WWISA World-wide Institute of Software Architects http://www.wwisa.org
- IASA International Association of Software Architects http://www.iasarchitects.org

Research and Education
- Nenad Medvidovic, at the University of Southern California: http://sunset.usc.edu/research/software_architecture/index.html
- Gert Florijn, at the Software Engineering Research Centre in the Netherlands: http://www.serc.nl/people/florijn/interests/arch.html

Short Bibliography

Len Bass, Paul Clements, and Rick Kazman. Software Architecture in Practice. Reading, MA: Addison-Wesley, 1998.

Jan Bosch. Design and Use of Software Architecture: Designing and Evolving a Product-Line Approach. Boston: Addison-Wesley, 2000.

Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. Pattern-Oriented Software Architecture: A System of Patterns. New York: John Wiley & Sons, 1996.

Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford. Documenting Software Architectures: Views and Beyond. Boston: Addison-Wesley, 2002.

Paul Clements, Rick Kazman, and Mark Klein. Evaluating Software Architecture. Boston: Addison-Wesley, 2002.

Paul Clements and L. Northrop. Software Product Lines: Practice and Patterns. Boston: Addison-Wesley, 2002.

David M. Dikel, David Kane, and James R. Wilson. Software Architecture: Organizational Principles and Patterns, Upper Saddle River, NJ: Prentice-Hall, 2001.

Christine Hofmeister, Robert Nord, and D. Soni. Applied Software Architecture. Boston: Addison-Wesley, 2000.

IEEE 1471:2000. "Recommended Practice for Architectural Representation." Los Alamitos, CA: Software Engineering Standards Committee of the IEEE Computer Society, 2000.

ISO/IEC 10746:1995. Reference Model of Open Distributed Processing (RM-ODP) (ITU Rec. X901). Geneva, Switzerland: ISO, 1995.

Ivar Jacobson, Martin Griss, and Patrik Jonsson. Software Reuse: Architecture, Process and Organization for Business Success. Reading, MA: Addison-Wesley, 1997.

Philippe Kruchten. "The 4+1 View of Architecture," in IEEE Software 6 (12), 1995.

Philippe Kruchten. "The Tao of the Software Architect," in The Rational Edge, March 2001.

Philippe Kruchten. "Common Misconceptions about Software Architecture," in The Rational Edge, April 2001.

Eberhardt Rechtin and Mark Maier. The Art of System Architecting 2/ed. Boca Raton, FL: CRC Books, 1997.

Bran Selic, Garth Gullekson, and Paul Ward. Real-Time Object-Oriented Modeling. New York: John Wiley & Sons, 1994.

Mary Shaw and David Garlan. Software Architecture: Perspectives on an Emerging Discipline. Upper Saddle River, NJ: Prentice-Hall, 1996.

More

D.E. Perry and A.L. Wolf, "Foundations for the Study of Software Architecture," ACM Software Engineering Notes, vol. 17, no. 4, 1992, pp. 40-52.

D.L. Parnas, "On the Criteria To Be Used in Decomposing Systems into Modules," Comm. ACM, vol. 15, no. 12, 1972, pp. 1053-1058.

D.L. Parnas, "On the design and development of program families," IEEE Transactions on Software Engineering, vol. SE-2, no. 1, 1976, pp. 1-9.

D.L. Parnas, P. Clements, and D.M. Weiss, "The Modular Structure of Complex Systems," IEEE Transactions on Software Engineering, vol. SE-11, no. 3, 1985, pp. 259-266.

F. DeRemer and H. Kron, "Programming-in-the large versus programming-in-the-small," in Proceedings of International Conference on Reliable software, Los Angeles, California, 1975, ACM, pp. 114-121.

D. Soni, R. Nord, and C. Hofmeister, "Software Architecture in Industrial Applications," in Proceedings of 17th International Conference on Software Engineering (ICSE-17), 1995, ACM Press, pp. 196-207.

P. Kruchten, "The 4+1 View Model of Architecture," IEEE Software, vol. 12, no. 6, 1995, pp. 45-50.

B.W. Lampson, "Hints for Computer System Design," Operating Systems Review, vol. 15, no. 5, 1983, pp. 33-48. Reprinted in IEEE Software, Vol. 1, no. 1, 1984, pp 11-28.

J.A. Mills, "A Pragmatic View of the System Architect," Comm. ACM, vol. 28, no. 7, 1985, pp. 708-717.

W.E. Royce and W. Royce, "Software Architecture: Integrating Process and Technology," TRW Quest, vol. 14, no. 1, 1991, pp. 2-15.

M. Shaw and P. Clements, "A Field Guide to Boxology: Preliminary Classification of Architectural Styles for Software Systems," in Proceedings of COMPSAC97, 21st Int'l Computer Software and Applications Conference, 1997, pp. 6-13.

M. Shaw, "The Coming-of-Age of Software Architecture Research," in Proceedings of 23rd International Conference on Software Engineering (ICSE), Toronto, 2001, IEEE.

More, on design decisions

Bosch, J. (2004, May). Software Architecture: the Next Step. Paper presented at the First European Workshop on Software Architecture (EWSA 2004), St Andrews, Scotland.

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.

Tyree, J., & Akerman, A. (2005). Architecture Decisions: Demystifying Architecture. IEEE Software, 22(2), 19-27.

Dueñas, J. C., & Capilla, R. (2005, June 13-15, 2005). The Decision View of Software Architecture. Paper presented at the 2nd European Workshop on Software Architecture, Pisa, Italy.

Tang, A., Babar, M. A., Gorton, I., & Han, J. (2005, Nov. 6-10). A Survey of Architecture Design Rationale. Paper presented at the Working IEEE/IFIP Conference on Software Architecture (WICSA 5), Pittsburgh, PA, USA.

Falessi, D., Cantone, G., & Becker, M. (2006). Documenting design decision rationale to improve individual and team design decision making: an experimental evaluation. Paper presented at the 2006 ACM/IEEE international Symposium on international Symposium on Empirical Soft. Eng. ISESE '06.

Jansen, A., & Bosch, J. (2005). Software Architecture as a Set of Architectural Design Decisions. Paper presented at the International Conference on Software Engineering ICSE 2005

Kruchten, P., Lago, P., van Vliet, H., & Wolf, T. (2005, Nov. 6-9). Building up and exploiting architectural knowledge. Paper presented at the 5th Working IEEE/IFIP Conference on Software Architecture (WICSA 5)

Jansen, A., Van der Ven, J. S., Avgeriou, P., & Hammer, D. (2006). Tool support for architectural decisions. Paper presented at WICSA 2007, Mumbai.

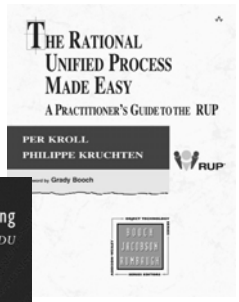
Lee, L., & Kruchten, P. (2007). Capturing Software Architectural Design Decisions. Paper presented at the 20th Annual Canadian Conference on Electrical and Computer Engineering (CCECE'07), Vancouver.

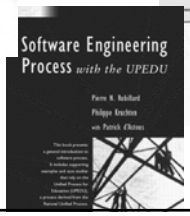
Falessi, D., Cantone, G., & Kruchten, P. (2007). Value-Based Design Decision Rationale Documentation: Principles and Empirical Feasibility Study. Paper presented at the 11th European Conference on Software Engineering, Dubrovnik.

De Boer, R. C., Farenhorst, R., Lago, P., van Vliet, H., & Clerc, V. (2007). Architectural Knowledge: Getting to the Core. Paper presented at the QoSA 2007

61

Shameless self-promotion





62