In Search of 'Architectural Knowledge'

Remco C. de Boer
Department of Computer Science
VU University Amsterdam
the Netherlands
remco@cs.vu.nl

Rik Farenhorst
Department of Computer Science
VU University Amsterdam
the Netherlands
rik@cs.vu.nl

ABSTRACT

The software architecture community puts more and more emphasis on 'architectural knowledge'. However, there appears to be no commonly accepted definition of what architectural knowledge entails, which makes it a fuzzy concept. In order to obtain a better understanding of how different authors view 'architectural knowledge', we have conducted a systematic review to examine how architectural knowledge is defined and how the different definitions in use are related. From this review it became clear that many authors do not provide a concrete definition of what they think architectural knowledge entails. What is more intriguing, though, is that those who do give a definition seem to agree that architectural knowledge spans from problem domain through decision making to solution; an agreement that is not obvious from the definitions themselves, but which is only brought to light after careful systematic comparison of the different studies.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures

General Terms

Design

Keywords

Architectural knowledge, systematic review

1. INTRODUCTION

Architectural knowledge and architectural knowledge management gain increasing interest from researchers and practitioners alike, evidenced among others by the continuing success of the SHARK workshop series. In this workshop series various topics explicitly devoted to architectural knowledge are identified, such as communicating and sharing architectural knowledge, ontologies and domain models for architectural knowledge, and tools to manage architectural knowledge. However, since the focus on architectural knowledge is relatively new, there is no commonly accepted definition of what it entails. This makes 'architectural knowledge' a fuzzy concept, and may evidently introduce misunderstandings when different authors use the same words with different meaning.

In order to obtain a better understanding of the concept of architectural knowledge, we have conducted a systematic review. In this systematic review, we searched for papers defining or discussing 'architectural knowledge'. We found 115 such papers, 14 of which provide an actual definition. We examined the relations between those definitions through reciprocal translational analysis. This analysis shows that most of the definitions focus on design decisions, although some refer to solution fragments or elements from the problem domain as well. A subsequent analysis of concepts related to those definitions shows that, contrary to the focus of the definitions, most authors seem to agree that architectural knowledge spans all these areas. We also encountered two surprising definitions: one, used by two authors, from the area of systems design and one almost a decade old but remarkably similar to current ones.

In the remainder of this paper we present our research and findings as follows. In Section 2 we describe the two main parts of the research methodology followed: the systematic selection of primary studies that discuss or define architectural knowledge and the synthesis of those studies through meta-ethnography, more in particular reciprocal translational analysis. Section 3 discusses the actual execution of our research. In Section 4 we provide a narrative summary of the 115 studies selected, followed in Section 5 by a synthesis of the 14 definitions of architectural knowledge found in those studies. In Section 6 we go beyond synthesis of the definitions, and perform synthesis at the level of the concepts used in the studies. Section 7 concludes our paper.

2. RESEARCH METHODOLOGY

Our study intends to give an overview of the current understanding of architectural knowledge; we do not intend to present in this paper our own definition of architectural knowledge. It is therefore a typical example of secondary research (or 'desk research') in which the results of different primary studies are combined through some form of meta-analysis. For the primary studies, we rely only on studies that have been published in the literature.

In a literature review such as ours, there are two important considerations: how to select the primary studies, and how

SHARK'08, May 13, 2008, Leipzig, Germany.

to perform the required meta-analysis. We have chosen to perform a systematic review for selecting studies and to use a meta-ethnographic approach as part of that systematic review for synthesizing the gathered data.

2.1 Studies Selection

There are basically two ways of performing a literature review: (1) an ad hoc review, and (2) a systematic review. The main difference between the two is the formality and a priori planning of the systematic approach. In a systematic review, a protocol is defined that specifies the research questions to be answered, as well as the manner in which data will be gathered. This contrasts with the ad hoc manner of conventional literature reviews. Systematic reviews are a heavily used instrument in (evidence-based) medicine. Work by various authors (e.g. [5, 9, 14]) has led to guidelines for applying systematic reviews to the domain of software engineering as part of the evidence-based software engineering paradigm.

Biolchini et al. propose a protocol template for systematic reviews in software engineering [5]. Their template combines review protocols from the medical field with earlier work on evidence based research and systematic reviews in software engineering. We have used Biolchini's template to develop a protocol for our systematic review of definitions of 'architectural knowledge'. The main elements of this protocol are given below.

- **Problem.** The problem we address is that there appears to be no commonly accepted definition of what architectural knowledge entails. This makes it a fuzzy concept, and makes the current understanding of what architectural knowledge entails unclear. This may evidently introduce misunderstandings when different authors use different definitions for the same concept.
- **Research Questions.** Particular research question our systematic review should answer are:
  - 1. What are the different definitions of 'architectural knowledge', and how are they related?
  - 2. What concepts are deemed related to 'architectural knowledge'?
- **Sources list.** Since we are interested in definitions of architectural knowledge in the context of software-intensive systems, we include the major publishers of and indexes that contain software engineering related publications:
  - ACM Digital Library
  - IEEE Xplore
  - ISI Web of Science
  - SpringerLink
  - ScienceDirect
  - Wiley Inter Science Journal Finder
- **Search string.** We are interested in architectural knowledge in the context of software-intensive systems (and not in the context of, for example, civil architecture). Furthermore, we are aware that some authors prefer to use the phrase 'architecture knowledge' over 'architectural knowledge'. We assume that a definition of 'architectural knowledge' is not present in a study if the phrase itself is not present. We therefore want to search for publications that contain either the word 'software' or 'system' and the phrase 'architecture knowledge' or 'architectural knowledge'. Therefore, the search string we use is:

  ('architectural knowledge' OR 'architecture knowledge') AND ('software' OR 'system')
- **Studies inclusion/exclusion criteria** Studies to be included in our review:
  - Must have 'architecture' as topic (and not e.g. present an architecture for a particular system)
  - Must be about architecture of software-intensive systems (and not about e.g. civil architecture)
  - Must have 'real' content (hence not a TOC, cover page, advertisement, etc.)
  - Must be in English
  - Must discuss 'architectural knowledge'

2.2 Data Synthesis

A potential problem with common meta-analytic approaches in systematic reviews is their focus on integration of quantitative data. Dybå et al. address the limited applicability of this type of meta-analysis to software engineering research [9].

Our review in particular is not very well suited for quantitative techniques, since the primary studies we address can be expected to be mostly grounded in qualitative research. Instead of a quantitative approach we therefore need a qualitative approach to data synthesis. Based on the experiences from Dybå et al., we chose meta-ethnography as the approach to synthesize the selected studies.

Meta-ethnography was proposed by Noblit and Hare [20] as a form of interpretive synthesis. Meta-ethnography relies on the translation of key concepts and metaphors of different studies into one another, for which there are three strategies. In reciprocal translational analysis, concepts from different studies are directly translated into each other. In refutational analysis, contradictions between studies are characterized. In lines-of-argument synthesis, a general interpretation is grounded in the findings of separate studies. For our review, we performed reciprocal translational analysis of the concepts that different authors use in defining and characterizing 'architectural knowledge'.

3. RESEARCH EXECUTION

The execution of our research commenced early January 2008. As a consequence, our review includes studies that were published and/or indexed before that date. We cannot be certain that we have covered all studies with a publication date in 2007, since especially the studies from Q4 may not have been indexed yet at the time we conducted our review.

3.1 Studies Selection

Since one cannot expect a definition of 'architectural knowledge' always to be found in the abstract of a study, we run the risk of missing relevant studies when we limit our search to abstracts only. We therefore conducted a full-text search in the six identified sources, which resulted in a list of 751 studies that matched the search string.

3.1.1 First Iteration: Scanning Titles

We evaluated compliance of the studies found with the inclusion criteria in four iterations. In the first iteration, we read only the titles of the identified studies and indicated (independently of each other) whether we considered the study out of the scope of our review, i.e. we could indicate at least one inclusion criterion for which it was clear from the paper's title that it would not be met. In this phase, only if we both agreed on at least one exclusion criterion the paper was excluded from our review. If, for example, we both determined from the title of the study that it was not in English we would exclude it. If, however, one of us found for instance that the paper was not about architecture and the other concluded that the study was in fact about architecture but not about architecture of software-intensive systems, the study would remain in the set of studies to be analyzed in the second iteration.

3.1.2 Second Iteration: Reading Abstracts

At the start of the second iteration we had 396 papers left, which means that we were able to exclude more than 45% of the studies by reading the title alone. In this iteration we followed the same procedure as in the first iteration, but instead of reading only the titles of the studies we now read the abstracts as well. After this iteration we were able to eliminate another 140 studies, which left us with 256 studies.

3.1.3 Third Iteration: Scanning Full-text

In the third iteration we did a full-text scan of the studies. Based on this scan we again assessed the studies' compliance with our inclusion criteria. Because of the amount of work involved, half of the 256 studies was independently assessed by each researcher. In this third iteration, we particularly focused on the phrases 'architectural knowledge' and 'architecture knowledge', which were part of the search string and therefore must be present in the text. Apart from various studies that were out of scope because they discussed for example the field of civil architecture, we found that a number of papers did not contain the phrase 'architectural knowledge' as we had intended it. For example, some studies only referred to other studies that happened to have 'architectural knowledge' in their title, or the full-text query that we executed had registered a match on only the keywords (e.g. 'software architecture; knowledge engineering'). Both researchers gathered evidence from the full text of the studies to support their assessment of in- or exclusion. This evidence (mainly literal quotes from the full text) was thereafter discussed and only when both researchers agreed that the study did not meet the inclusion criteria it was excluded. Whenever necessary, the full text of the study itself was consulted during this discussion. In this way, we were able to exclude another 80 studies, leaving us with 158 studies.

3.1.4 Fourth Iteration: Reading Full-text

In the fourth, and final, iteration the full text of the studies was not scanned, but thoroughly read. It was only in this phase that we could really assess whether a study did or did not discuss 'architectural knowledge'. In some studies, while the study did use the phrase 'architectural knowledge' it remained completely unclear what the authors considered to be architectural knowledge. For example, a study could talk about 'architectural knowledge present in the organization' without the definition or scope of such architectural knowledge further being discussed in the text. Those studies were excluded from the review. After this iteration we were left with a total of 115 studies in which the authors talk about architectural knowledge in more or less concrete terms, i.e. the authors either provide a clear definition, or discuss several concepts related to 'architectural knowledge'. As it turns out, the latter is much more common. Quite a lot of papers address the notion of architectural knowledge, but many of them refrain from explicitly stating what architectural knowledge entails. Out of the 115 publications we found, only 14 studies define what the authors consider architectual knowledge.

3.2 Data Synthesis

The high number of studies (115) that matched our inclusion criteria made it infeasible to translate all studies found into each other. Since we were particularly interested in how architectural knowledge is defined, we decided to limit the meta-ethnographical synthesis to the 14 papers that provide a definition of architectural knowledge. Since we read the full text of all selected publications, we obviously gained an understanding of those other 101 studies as well. A short narrative summary of those studies is presented in Section 4.

To the 14 papers that define architectural knowledge, we applied reciprocal translational analysis in two ways. We first focused on the concepts explicitly used in the different definitions. By translating those concepts into each other, we were able to identify four areas from which – according to the 14 definitions – architectural knowledge originates: problem domain, design decisions, solution fragments, and systems design. This synthesis of architectural knowledge definitions is presented in Section 5.

After synthesis of the definitions, we extended the reciprocal translational analysis to find concepts that – although not part of their definition of architectural knowledge – the various authors discussed in terms of elements that contain, provide, or are part of architectural knowledge. We were able to classify most of those concepts in the four areas identified in the definition synthesis. Nevertheless, one new architectural knowledge area emerged from this concept synthesis: implementation. The synthesis of architectural knowledge concepts is presented in Section 6.

4. NARRATIVE SUMMARY OF PRIMARY STUDIES

One of the first striking observations from the 115 selected primary studies it the very clear upward trend in the number of publications that discuss architectural knowledge. There is undoubtedly an increasing interest in the topic.

Fig. 1 shows the number of publications per year that discuss and/or define architectural knowledge. The figure clearly shows that in 1992 the concept was coined for the first time and that from that date on every year 'architectural knowledge' crops up in at least one publication. However, until the early 2000s architectural knowledge received fairly little attention. From 2001 onward, the number of publications per year has seen uninterrupted and rapid growth which continues to this date.

After reading the selected studies, we gained the impression that there are different communities that address architectural knowledge in their own specific way.

The **patterns community** has since the early nineties

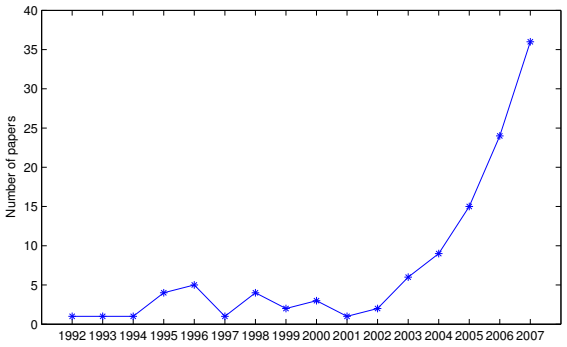

Figure 1: Papers that discuss architectural knowledge

explored and published patterns and pattern languages of software development in an ever growing body of literature. The community stresses the advantages of having reusable solutions to specific problems in software design. In several publications such solutions are related to architectural knowledge, usually without much further definition. Typically, such publications would present patterns as a means to communicate architectural knowledge, for example by stating that *"patterns disseminate modelling and architectural knowledge among developers."* [13].

In the **requirements engineering community**, researchers relate functional and quality requirements to on the one hand problems that need to be addressed, and on the other hand solutions that meet those requirements. Architecture, then, is related to the structure of those solutions. Within the requirements engineering community architectural knowledge seems to particularly play a role in discussions of the reciprocal relation between problem domain and solution space, for example in Pohl and Sikora's work on the co-design of requirements and architectural artefacts. There, the authors stress the role of architectural knowledge in satisfying the need for knowledge about the (coarse) solution when defining (detailed) system requirements: "*We illustrate the important role that [this] architectural knowledge plays in requirements engineering by means of a simplified example from the automotive domain*" [21].

We also observed a **model-oriented community** that takes a rather formal stand when it comes to software architecture. This is also visible in how architectural knowledge is treated. In such contexts as dynamically adaptable systems, agent systems, or embedded systems, architectures are often modeled using architecture description languages (ADLs) and emphasis is put on architecture in terms of its components and connectors. Formal reasoning over this knowledge is used for example to allow for dynamic runtime changes to the system configuration. In the work of Georgas and Taylor, for instance, an architecture-centric knowledge-based approach is presented to specify and enact architectural adaptation policies for self-adaptive software which involves among others *"the leveraging of existing knowledge-based techniques for dynamic reasoning over the space of architectural knowledge"* [11].

Researchers in the **software architecture community** are engaged in studying different aspects pertaining to software architectures. Until recently this was often based on a view on architecture as components and connectors, but over the past few years more and more studies consider not only the resulting architecture itself, but also the design decisions that have led to this result. In those studies, architectural knowledge is often linked to or even equated to architectural design decisions. In this sense, the phrase 'architectural knowledge' is often used to denote the knowledge *about* a solution instead of the solution itself, for example in work by Van der Ven et al., where the authors note that *"evaluated alternatives, made tradeoffs and rationale about the made decision [often] remain in the heads of the designers"* [23].

5. SYNTHESIS OF ARCHITECTURAL KNOWLEDGE DEFINITIONS

Although we found 115 studies that discuss architectural knowledge, only 14 of them give a clear definition of what architectural knowledge entails. These definitions are listed in Table 1, ordered chronologically by publication date. This table shows that almost all definitions were proposed over the past three years, which corresponds to the increasing interest in architectural knowledge observed earlier.

Chronologically, a clear outlier in Table 1 is the work by Ran and Kuusela [22]. Already in 1996, they are the first to give a definition of architectural knowledge. Moreover, their definition is remarkably similar to the view on architectural knowledge currently prevalent in the software architecture community. However, somehow this work seems unrelated to the stir that started in the early 2000s and brought more attention to architectural knowledge, especially in terms of design decisions. On the contrary, as far as we know Ran and Kuusela's work is hardly referenced in current discussions.

In the 14 definitions from Table 1, there are 15 concepts that play a major role. Not every definition relies on unique concepts, but similar concepts are used by different authors in different definitions. In Table 2, we have made this relation clear by translating each definition of architectural knowledge into the concepts it uses.

If we carefully examine the concepts used in definitions of architectural knowledge, we can identify four higher-level constructs to which those definitions relate: decision, problem domain, solution fragment, and systems design. The relation between the definitions and the four constructs is represented by the four gray areas in Table 2. To a certain extent, the high-level constructs correspond to the different communities that address architectural knowledge (cf. Section 4), but there are also differences:

1. **Definitions that center around design decisions**
   By far the largest percentage of the definitions we found focus on design decisions [1, 3, 8, 10, 12, 15, 16, 17, 18, 19, 22]. Some are limited to design decisions and accompanied rationale, some others are a bit broader and also explicitly refer to the resulting design. The decision construct to a large extent corresponds to the view of the software architecture community on architectural knowledge.

2. **Definitions that contain elements from the problem domain**
   Several definitions contain elements from the problem domain [1, 4, 12, 15, 16, 18]. Those elements are typi-

Author(s)	Architectural Knowledge Definition
Ran and Kuusela (1996) [22]	To avoid replication when representing variations and alternatives DDT structures architectural knowledge hierarchically into fine-grain elements we call design decisions. [22]
Carayannis and Coleman (2005) [6]	The architectural innovation is dependent on the system designers' knowledge of the components in the system and their knowledge of the configuration of the components. Henderson and Clark (as cited in Afuah, 1998) show the knowledge as Component Knowledge (CK) and the latter Architectural knowledge (AK).
Chen (2005) [7]	A distinction that is particularly significant in the product innovation context is the distinction between component-specific knowledge and "architectural" knowledge (Henderson and Clark, 1990). Component knowledge is knowledge that concerns a particular aspect of an organization's product, process or operation. Architectural knowledge, on the other hand, relates to the various ways in which the components are integrated and linked together into a complete system.
Kruchten et al. (2005) [16]	Architectural knowledge consists of architecture design as well as the design decisions, assumptions, context, and other factors that together determine why a particular solution is the way it is.
Kruchten et al. (2006) [15]	Architectural Knowledge = Design Decisions + Design, derived from 'Architectural knowledge consists of architecture design as well as the design decisions, assumptions, context, and other factors that together determine why a particular solution is the way it is.'
Kruchten et al. (2006) [17]	Some researchers are looking into architectural knowledge – that is, architectural design decisions and their rationale.
Babar et al. (2006) [3]	We propose a framework for managing design rationale to improve the quality of architecture process and artifacts. This framework consists of techniques for capturing design rationale, and approach to distill and document architectural information from patterns, and a data model to characterize architectural constructs, their attributes and relationships. These collectively comprise Architectural Design Knowledge (ADK) to support the architecting process.
SHARK workshop (2006,2007) [18, 1]	Architectural Knowledge (AK) is defined as the integrated representation of the software architecture of a software-intensive system or family of systems along with architectural decisions and their rationale external influence and the development environment.
Lee and Kruchten (2007) [19]	Software architectural knowledge is composed of the design and the set of decisions made to arrive at the design.
De Boer and Van Vliet (2007) [8]	Following a recent trend in software architecture research we refer to the collection of architectural design decisions and the resulting architectural design as 'architectural knowledge'.
Farenhorst et al. (2007) [10]	[..] not only the architecture design itself is important to capture, but also the knowledge pertaining to it.Often, this so-called architectural knowledge is defined as the set of design decisions, including the rationale for these decisions, together with the resulting architectural design.
Habli and Kelly (2007) [12]	Architectural Knowledge = {drivers, decisions, analysis}
Babar and Gorton (2007) [2]	[The knowledge management component] provides services to store, retrieve, and update artifacts that make up architectural knowledge.
Bahsoon (2007) [4]	We anticipate the architectural knowledge to constitute architectural artifacts such as deployable components and associated specification of what the components provide and require, quality requirements, scenarios corresponding to specific dependability requirements, and possibly dependable styles and patterns.

Table 1: Definitions of 'architectural knowledge'

cally factors that influence the architect's work. None of the definitions solely contains problem domain concepts, but about half of the definitions does take the problem domain into account. The problem domain construct relates to the view of the requirements engineering community on architectural knowledge, especially since the problem domain elements do not stand on their own but are – through the definitions – linked to either decisions or solutions.

3. **Definitions that specify solution fragments**
   Closely related to the view of the pattern community on architectural knowledge, some definitions specify solution fragments as key aspects of architectural knowledge [2, 3, 4]. But, as with elements from the problem domain, those solution fragments are never considered to fully encompass architectural knowledge.

4. **Definitions from the realm of systems design**
   Two studies apply a surprising definition of architectural knowledge, referred to particularly in studies from systems design and product development, to the world of software-intensive systems [6, 7]. This definition is completely orthogonal to other definitions used, and talks about components in a less technical sense than the usual component-connector meaning. One of the authors refers to components as "a particular aspect of an organization's product, process, or operation" [7], whereas the other defines components of a technical system as "products, processes, people, services, and technologies" [6]. Nevertheless, one of the authors uses this definition to assess knowledge creation in different software development settings. This topic would probably interest those who approach architectural knowledge from an orthogonal point of view as well. The systems design view on architectural knowledge does not seem to fit any of the communities we identified in Section 4.

Possibly apart from the concept 'architectural design', none of the concepts from Table 2 seems to particularly relate to the model-oriented community. We also found two concepts (architectural construct and architectural artifact) to be too broad to fit one particular construct. From the definition alone, it remains unclear to which construct those concepts would or would not belong. In order to obtain a

		Kruchten et al. (2006) (IEEE)	Farenhorst et al. (2007)	Ran and Kuusela (1996)	Lee and Kruchten (2007)	De Boer and Van Vliet (2007)	SHARK workshop (2006 / 2007)	Kruchten et al. (2006) (QoSA)	Kruchten et al. (2005)	Habli and Kelly (2007)	Bahsoon (2007)	Babar et al. (2006)	Babar and Gorton (2007)	Carayannis and Coleman (2005)	Chen (2005)
Decision	Architectural Design		x		x	x	x	x	x						
	Design decisions	x	x	x	x	x	x	x	x	x					
	Rationale	x	x				x	(x)	x			x			
Problem Domain	Analysis									x					
	Non-funct. requirement										x				
	Assumption							(x)	x						
	Context						x	(x)	x						
	Driver									x					
	Scenario										x				
Solution Fragment	Patterns										x	x			
	Styles										x				
	Deployable component										x				
Systems Design	Component configuration													x	x
	Architectural constructs											x			
	Architectural artifact										x		x		

Table 2: Reciprocal translational analysis of architectural knowledge definitions

more detailed understanding of the various definitions, we need to go beyond synthesis of the definitions, and perform synthesis at the level of the concepts used in the studies.

6. SYNTHESIS OF ARCHITECTURAL KNOWLEDGE CONCEPTS

In order to synthesize all concepts related to the various definitions of architectural knowledge, we first of all need to determine which concepts the various authors deem related. The concepts directly used in the definitions are obviously related to those definitions, and form the basis for recursive synthesis and identification of other related concepts.

For each of the 14 studies that provide a definition, we have carefully read the text to determine which concepts are used in an explanation or further discussion of that definition. Whenever we encountered a new concept, we added it to the grid shown in Table 2[1]. We then studied the text for possible discussions on other concepts not directly related to the definition from that particular study, but used in or related to other definitions. In other words, whenever an additional concept was found in one of the publications, we re-examined all other studies to determine whether that concept was discussed there as well.

Figure 2 shows all concepts identified using this approach. The mapping of those concepts to the studies in which they are discussed is shown in Table 3, where the numbers in the grid refer to the numbers of the concepts in Fig. 2. As in the synthesis of architectural knowledge definitions, we tried to derive higher-level constructs from the identified concepts and grouped the concepts accordingly. Besides stronger support for the constructs identified in the synthesis of definitions in Section 5, we obtained another construct 'implementation' that remains hidden when one looks at the definitions alone. Apparently, some authors consider the implementation to be related to architectural knowledge, but their definitions do not tell.

Another interesting observation comes from a comparison of Tables 2 and 3. Table 3 clearly shows that almost

Problem Domain
1. Arch. significant requirement
2. Analysis
3. Assumption
4. Concern
5. Constraints
6. Context
7. Design problem
8. Driver
9. Finding
10. Forces
11. Goal
12. Need
13. Non-funct. requirement
14. Non-risk
15. Quality Attributes
16. Quality Factor
17. Requirement
18. Risk
19. Scenario

Decision
20. Tradeoffs
21. Alternative
22. Architectural Design
23. Design decisions
24. Design option
25. Rationale
26. Design History

Solution Fragment
27. Tactics
28. Patterns
29. Reference architectures
30. Solution
31. Styles
32. Deployable component

Implementation
33. Documentation
34. Data models
35. Implementation
36. Product Artifacts
37. Software component
38. Source code

Systems Design
39. Services
40. Component
41. Component configuration
42. Operation
43. People
44. Process
45. Product
46.Technologies

Unclassified
47. Architectural construct
48. Architectural artifact

Figure 2: Categories of related architectural knowledge concepts

[1]For bookkeeping purposes, the grid we used did not contain 'x'es to mark relations, but literal quotes from the original text that showed those relations. Unfortunately, space limitations prevent us from showing the resulting grid in this paper. However, the complete mapping table is available online from http://www.cs.vu.nl/~remco/akconceptmapping.html

	Problem Domain	Decision	Solution Fragment	Implementation	Systems Design
Ran and Kuusela (1996)	3,5,10,11,17	21,22,23,25	28,30		
Carayannis and Coleman (2005)					39,40,41,43,44,45,46
Chen (2005)					40,41,42,44,45
Kruchten et al. (2005)	3,5,6,10,12,17	22,23,25		35	
Kruchten et al. (2006) (QoSA)	3,4,5,6,10,12,13,15,17	22,23,25	30	34,38	
Kruchten et al. (2006) (IEEE)		23,25			
Babar et al. (2006)	1,6,13,15,19	23,24,25	28,32		
SHARK workshop (2006/2007)	6,15	21,22,23,25	27,28,29,30		
Lee and Kruchten (2007)	3,5,13	22,23,25		36,37	
De Boer and Van Vliet (2007)	5,10,15	21,22,23,25	28,30	33,34,36,38	
Farenhorst et al. (2007)	4,12	21,22,23,25	27,28,30	36	
Habli and Kelly (2007)	2,3,4,5,7,8,13,15,17,18	21,22,23,25	27,28,30	36	
Babar and Gorton (2007)	1,5,9,13,14,15,16,17,18,19	21,23,24,25,26	27,28,30		
Bahsoon (2007)	2,13,19	22,23	27,28,31,32	36,37	

Table 3: Reciprocal translational analysis of concepts related to definitions of architectural knowledge

all authors who define architectural knowledge (outside the scope of systems design) consider architectural knowledge to span from problem domain through decision making to solution fragments, and possibly even implementation. Table 2, however, shows that none of the definitions span this entire space. This begs the question whether the definitions are too narrowly scoped, or whether the authors consider certain concepts deducible from others.

7. CONCLUSIONS AND OUTLOOK

Conducting a systematic review is obviously more enduring and time-consuming than a more ad hoc overview of related work, but we feel it is well worth the extra effort. By systematically scrutinizing available literature, we obtained an understanding of the domains in which the concept is used, the different definitions of architectural knowledge, and how these are related. A potential and obvious drawback of this approach is the reliance on the search string used. Communities that discuss topics related to architectural knowledge but never use that phrase itself are per definition not included in our analysis. However, given our goal to investigate current definitions of architectural knowledge we do not see the exclusion of communities that do not use the literal phrase – let alone define it – as a shortcoming.

Based on the number of publications that discuss architectural knowledge, we see an increasing interest in the subject matter. However, only a small percentage of those studies (14 out of 115 publications) propose or refer to a definition of architectural knowledge. Most of those definitions focus on design decisions, although some refer to solution fragments or elements from the problem domain as well.

When we look beyond the definitions, we see that almost all of the authors who provide a definition of architectural knowledge consider that knowledge to span from problem domain through decision making to solution fragments and sometimes also implementation. This contrasts with the provided definitions, none of which spans this complete space, and begs the question whether the definitions are too narrowly scoped, or whether the authors consider certain concepts deducible from others and therefore leave them out of their definition. We have not addressed this question in our present work, but find it interesting enough to revisit it in future work.

We found two surprising definitions of architectural knowledge. One surprise was a definition of architectural knowledge often used in the broad area of systems design, applied to the development of software intensive systems by two authors. This definition is orthogonal to the definitions of architectural knowledge in software development, but at the same time seems related given its application to characterize different software development projects by one of the authors. The other surprise was a definition of architectural knowledge coined in 1996, remarkably similar to recently proposed definitions that focus on design decisions. Between 1996 and 2005, no definitions or references thereto were published in the literature. It seems as if this early definition was forgotten and almost reinvented after nearly a decade.

We were somewhat disappointed by the low number of actual definitions (or references thereto) in the current literature. We attribute this low number of definitions to the relatively recent focus on architectural knowledge. The fact that 7 out of the 14 definitions come from studies that were published last year (2007) leads us to believe that we will see more definitions being coined and discussed in the near future. We are unsure whether the perfect definition of architectural knowledge that everyone agrees upon will ever be found, but we do want to urge researchers to be precise and concrete in defining the concepts they consider part of architectural knowledge. In this way, ambiguity can be prevented and the community as a whole can work toward a better common understanding of the scope and span of 'architectural knowledge'.

Acknowledgements

This research has been partially sponsored by the Dutch Joint Academic and Commercial Quality Research & Development (Jacquard) program on Software Engineering Research via contract 638.001.406 GRIFFIN: a GRId For inFormatIoN about architectural knowledge.

The authors would like to thank Torgeir Dingsøyr for sharing his experiences in conducting a systematic review.

8. REFERENCES

[1] P. Avgeriou, P. Kruchten, P. Lago, P. Grisham, and D. Perry. Sharing and Reusing Architectural Knowledge–Architecture, Rationale, and Design

Intent. In *29th International Conference on Software Engineering - Companion (ICSE)*, pages 109–110, 2007.

[2] M. A. Babar and I. Gorton. A Tool for Managing Software Architecture Knowledge. In *Second Workshop on Sharing and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent, (SHARK/ADI)*, 2007.

[3] M. A. Babar, I. Gorton, and B. Kitchenham. A Framework for Supporting Architecture Knowledge and Rationale Management. In *Rationale Management in Software Engineering*, pages 237–254. 2006.

[4] R. Bahsoon. Defining Dependable Dynamic Data-Driven Software Architectures. In *IEEE International Conference on Information Reuse and Integration, (IRI)*, pages 691–694, 2007.

[5] J. Biolchini, P. G. Mian, A. C. C. Natali, and G. H. Travassos. Systematic Review in Software Engineering. Technical Report ES 679 / 05, May 2005.

[6] E. Carayannis and J. Coleman. Creative System Design Methodologies: the Case of Complex Technical Systems. *Technovation*, 25(8):831–840, 2005.

[7] S. Chen. Task Partitioning in New Product Development Teams: A Knowledge and Learning Perspective. *Journal of Engineering and Technology Management*, 22(4):291–314, 2005.

[8] R. C. de Boer and H. van Vliet. Constructing a Reading Guide for Software Product Audits. In *6th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, pages 11–20, Mumbai, India, 2007.

[9] T. Dybå, T. Dingsøyr, and G. K. Hanssen. Applying Systematic Reviews to Diverse Study Types: An Experience Report. In *First International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 225–234, Madrid, Spain, 2007.

[10] R. Farenhorst, P. Lago, and H. van Vliet. Effective Tool Support for Architectural Knowledge Sharing. In *1st European Conference on Software Architecture (ECSA)*, pages 123–138, Aranjuez (Madrid), Spain, 2007.

[11] J. C. Georgas and R. N. Taylor. Towards a Knowledge-based Approach to Architectural Adaptation Management. In *1st ACM SIGSOFT Workshop on Self-managed Systems*, Newport Beach, California, 2004.

[12] I. Habli and T. Kelly. Capturing and Replaying Architectural Knowledge through Derivational Analogy. In *Second Workshop on Sharing and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent, (SHARK/ADI)*, Minneapolis, USA, 2007.

[13] E. A. Kendall. Utilizing Patterns and Pattern Languages in Education. *Annals of Software Engineering*, 6(1):281–294, 1998.

[14] B. Kitchenham. Procedures for Performing Systematic Reviews. Technical Report TR/SE-0401 and NICTA Technical Report 0400011T.1., Keele University, 2004.

[15] P. Kruchten, P. Lago, and H. van Vliet. Building Up and Reasoning About Architectural Knowledge. In *Second International Conference on the Quality of Software Architectures (QoSA)*, pages 43–58, Stockholm, Sweden, 2006.

[16] P. Kruchten, P. Lago, H. van Vliet, and T. Wolf. Building up and Exploiting Architectural Knowledge. In *5th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, pages 291–292, Pittsburgh, Pennsylvania, USA, 2005.

[17] P. Kruchten, H. Obbink, and J. Stafford. The Past, Present, and Future for Software Architecture. *IEEE Software*, 23(2):22–30, 2006.

[18] P. Lago and P. Avgeriou. First ACM Workshop on SHAring and Reusing architectural Knowledge (SHARK). *SIGSOFT Software Engineering Notes*, 31(5):32–36, 2006.

[19] L. Lee and P. Kruchten. Capturing Software Architectural Design Decisions. In P. Kruchten, editor, *Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 686–689, 2007.

[20] G. Noblit and R. Hare. *Meta-Ethnography: Synthesising Qualitative Studies*. Sage, Newbury Park, California, 1988.

[21] K. Pohl and E. Sikora. Structuring the Co-design of Requirements and Architecture. In *Requirements Engineering: Foundation for Software Quality*, pages 48–62. 2007.

[22] A. Ran and J. Kuusela. Design Decision Trees. In *8th International Workshop on Software Specification and Design*, pages 172–175, 1996.

[23] J. S. van der Ven, A. Jansen, J. Nijhuis, and J. Bosch. Design decisions: The Bridge between Rationale and Architecture. In A. Dutoit, editor, *Rationale Management in Software Engineering*, pages 329–346. Springer-Verlag, 2006.