Experiences with Semantic Wikis for Architectural Knowledge Management

Remco C. de Boer
ArchiXL
Amersfoort, the Netherlands
rdeboer@archixl.nl

Hans van Vliet
VU University
Amsterdam, the Netherlands
hans@cs.vu.nl

***Abstract*—In this paper, we reflect on our experiences with using semantic wikis for architectural knowledge management in two different contexts: e-government and distributed software development. Whereas our applications of semantic wikis in e-government focus on organizing and structuring architectural knowledge for reuse, the applications in distributed software development focus on searching and querying architectural knowledge. Yet, the emerging research challenges – alignment of knowledge models, knowledge versioning, change acknowledgements – are very similar.**

***Keywords*-architectural knowledge management; semantic wiki; experience report**

I. Introduction

From 2005 to 2009, the GRIFFIN research project addressed the - at that time fairly new - research topic of ‘architectural knowledge management’. The project has delivered several concrete results, including PhD theses ([1], [2]), the SHARK workshop series on Sharing and Reusing Architectural Knowledge, and a book dedicated to theoretical and practical aspects of architectural knowledge management [3]. More importantly, the GRIFFIN project contributed to a different mindset in the field of software architecture in which more attention is put on rationale and decisions as first class entities, and the ensuing architecural design is seen more as the result of an architectural decision making process.

Architectural knowledge is increasingly regarded as an organizational asset that should be managed. We may distinguish different types of architectural knowledge along two orthogonal axes: tacit vs. explicit, and generic vs. organization-specific [2]. Explicit architectural knowledge can be further divided into documented (unstructured) and formal (structured) architectural knowledge (cf. [1]).

A great deal of explicit architectural knowledge encountered in practice falls in the category of ‘documented’ and ‘unstructured’ knowledge. Architecture descriptions usually consist of textual descriptions and graphical depictions of the architectural structure, both of which have no formal structure attached to them. Architectural knowledge may also be found in other documents, emails, meeting minutes, et cetera, but these are all unstructured forms of explicit architectural knowledge.

Formal representations of explicit architectural knowledge one may encounter include architecture description languages (ADLs) as well as instances of design decision ontologies such as the one proposed by Kruchten [4]. This architectural knowledge conforms to a well-defined semantics, which makes it possible to reason over the codified knowledge.

Architectures are not created in isolation. Intra- and interorganizational collaboration, e.g., between different stakeholders in an organization-specific architecture or between organizations that collectively maintain and/or use a generic reference architecture, requires that architectural knowledge is shared within or between organizations. In previous work [5], we have outlined the requirements for collaboration in such environments: Manage architectural decisions, Codify architectural knowledge, Search architectural knowledge, Support community building, Provide intelligent support, and Enrich architectural knowledge.

Satisfying these requirements in an integrated environment is non-trivial, since the requirements combine aspects of supporting management of formal architectural knowledge (e.g., intelligent support; insight into completeness, correctness, and consistency of architectural decisions), documented architectural knowledge (e.g., codification of architectural knowledge in documents and models) and even tacit architectural knowledge (e.g., community building). Structured knowledge management environments, based on for instance rule bases and/or semantically enhanced modeling environments, may very well support management of formal architectural knowledge at the cost of rigidity and a poor treatment of purely textual information. Unstructured environments on the other hand, such as word processors or drawing tools, provide flexibility at the cost of a lack of ‘intelligence’.

Over the past year, we have experimented with the use of semantic wikis to support architectural knowledge management in different contexts. The VU University has a research collaboration with industries to investigate ways in which semantic wikis can support (distributed) software development. A recurring theme in such environments is the accessibility of documents. ArchiXL, on the other hand, is an IT architecture consultancy that operates primarily at the level of enterprise architecture, i.e., the fundamental organization of an enterprise’s information landscape and the related technology. The company has a specific focus on the governmental, educational, and financial domains. One of the recurring themes ArchiXL deals with is the ongoing introduction of 'e-government'.

In this paper, we reflect on our experiences with using semantic wikis for architectural knowledge management in these different contexts. Whereas our applications of semantic wikis in e-government focus on organizing and structuring architectural knowledge for reuse, the applications in distributed software development focus on searching and querying architectural knowledge. Yet, the emerging research challenges touch upon the same themes.

The remainder of this paper is organized as follows. In Section II, we provide a background on semantic wikis and how they compare to regular wikis. In Section III, we describe the use of semantic wikis to support reuse of reference architecture knowledge within the Dutch e-government domain. In Section IV, we present the use of semantic wikis to capture architectural knowledge in software documentation within a large industrial manufacturer. In Section V, we reflect on our experiences and list lessons learned from the use of semantic wikis for architectural knowledge management. In Section VI, we provide a number of research challenges and research opportunities that we identified over the course of our experiments. Section VII contains our concluding remarks.

II. Semantic Wikis

A wiki is a website where users can easily customize pages and add new pages from within their web browser. The very first wiki, called WikiWikiWeb, was created in the mid-nineties by Ward Cunningham as part of the Portland Pattern Repository. Since then, many different wiki engines have been built and put to use for many different purposes. The main goal of each wiki has remained the same: to allow users to collaborate on the content of the wiki. As such, a wiki is a collaborative platform that strongly invites users to share their knowledge. The best known example of a wiki is probably Wikipedia[1], an online encyclopedia in wiki format which was cocreated by millions of people worldwide. Wikipedia is one of the prime examples of a successful wiki. However, regardless of its success, Wikipedia (like other wikis) has several shortcomings that are intrinsic to the way in which knowledge is codified, namely in the form of interlinked web pages consisting solely of unstructured (textual and/or graphical) information. There is no way to directly ask questions to the wiki engine, even though the wiki might contain all the knowledge that is needed to answer that question. One can see evidence of this in the multitude of "List of..." articles ("List of cities with the most high-rise buildings", "List of actors who played President of the United States", "List of countries with fewer than 100,000 people", et cetera) that are carefully maintained by Wikipedia users in order to satisfy some particular information need. However, these pages - being maintained by hand - always run the risk of getting out of sync with other pages.

[1]http://www.wikipedia.org

A semantic wiki adds to a regular wiki an underlying knowledge model. This model describes the knowledge that the wiki contains. Such a description makes the facts and relationships in the wiki meaningful, for both man and machine. From this meaning (or 'semantics'), new relationships and facts can be derived. Moreover, the wiki can be directly queried for the knowledge it contains, and - based on these queries - reports may be dynamically generated and visualized. In a semantic wiki, we could for instance semantically annotate a page about a particular country with the number of inhabitants that country has. Using these semantic annotations, the wiki engine itself can determine which countries have fewer than 100,000 people and present the result as a list, a table, or even a chart. There is no longer a need to manually keep track of the countries that should be on this list. Whenever the semantically annotated data on a page changes (say, the number of inhabitants of some country rises above 100,000), the list would immediately and automatically be updated to reflect this new fact. Moreover, users of the wiki can formulate queries against the semantically annotated pages.

III. Architectural Knowledge and e-Government

The Dutch government is currently transforming into what is often referred to as 'e-government'. The objective of e-government is to provide better services to citizens and businesses. To reach this goal, government agencies need to work together, align their business processes and use each other's information.

For example, until recently building or renovating a house in the Netherlands required several different permits (e.g., one for demolition, one for construction, one for felling a tree that stands in the way, et cetera), independently issued by different government agencies. As of October 2010, a new law is effective that aggregates a substantial number of permits (about 25 of them) in one "environmental permit". Municipalities act as a single point of entry for this permit, while behind the scenes different government agencies are still involved in judging 'their' part of the permit request. Under the new law, it is the government agencies' responsibility to orchestrate their work and reach a joint judgment on the requested permit.

In order to reach the necessary level of collaboration, the government has issued the Dutch Government Reference Architecture (NORA). This reference architecture primarily consists of a set of principles that, when adhered to, establish processes and systems that ensure interoperability. The NORA is further refined by more domain-specific reference architectures such as those for municipalities (GEMMA), provinces (PETRA), water control boards (WILMA), civil service (MARIJ), and education (ROSA). Each of these reference architectures falls under the authority of - and is maintained by - a particular governmental body. The reference architectures in turn form the basis for organization-specific enterprise architectures, e.g., for a particular municipality or educational institution.

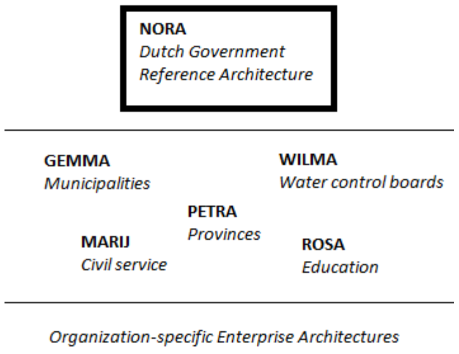


Figure 1. The NORA family of reference architectures

Principles, such as those used by NORA and its domain-specific descendents, may be defined as "general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an organization sets about fulfilling its mission" [6]. Architecture principles lay on the boundary between high level strategic intents and concrete designs; they provide the boundaries within which design decisions are taken [7]. The core attributes of an architecture principle are the principle statement, its motivation and its implications.

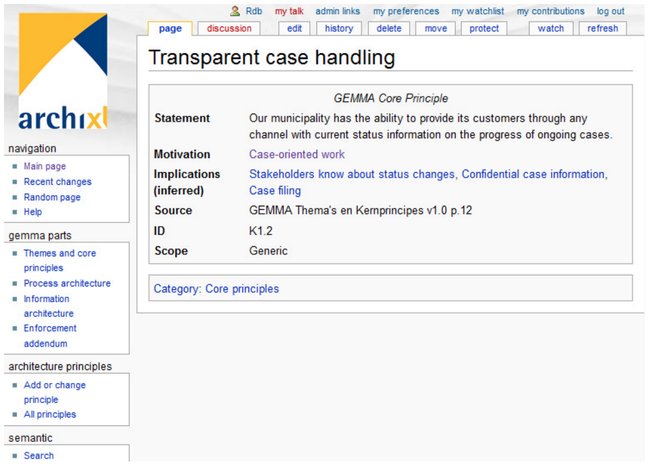
Enterprise architecture models are often drawn with ArchiMate [8]. The ArchiMate language is an enterprise architecture modeling language that defines structure and behavior elements at three layers: the business layer, the application layer and the technology layer. The language uses a service-oriented approach at each of these layers. Business layer concepts include actors, roles, processes, and business services. Application layer concepts include application components and application services. Technology concepts include devices, system software, and infrastructure services. ArchiMate offers different types of relationships that may be used to link elements. For instance, application components may be composed of other application components and may realize application services. Application services, in turn, may be used by business processes which may be assigned to business actors.

We have been using Semantic MediaWiki[2] to capture, relate and disseminate enterprise architecture knowledge especially in the form of architecture principles and ArchiMate models. Semantic MediaWiki is a plugin that provides semantic capabilities, such as annotation and querying, to the MediaWiki wiki engine[3]. The MediaWiki engine is the same engine that runs the Wikipedia family of wikis.

[2]http://semantic-mediawiki.org
[3]http://www.mediawiki.org

Within the semantic wiki, principles are mapped onto wiki pages such that a single principle corresponds to a single page in the wiki. The rationale and implications can either be expressed in plain text or as references to other pages in the wiki. The latter creates a tree- or graph-like structure that allows for traceability between principles as well as to other architectural elements, e.g., ArchiMate model elements. The way in which principles are thus recorded can, by the way, also be used to record other statements that are defined along the lines of "argument-rationale-implications". This pattern also applies to, for instance, strategic goals, policies or design decisions. In other words, traceability can be extended beyond the level of architecture principles, into e.g., strategy and/or design.

Figure 2 shows an example principle as it is shown in the semantic wiki. The page name ("Transparent case handling") coincides with the name by which we refer to the principle. The page further contains several properties that belong to the principle: the principle statement, its motivation and implications, the source of the principle, its ID, and its scope. All of these properties have been semantically annotated, i.e., they can be queried or used in dynamically generated reports.



Transparent case handling

GEMMA Core Principle

Statement	Our municipality has the ability to provide its customers through any channel with current status information on the progress of ongoing cases.
Motivation	Case-oriented work
Implications (inferred)	Stakeholders know about status changes, Confidential case information, Case filing
Source	GEMMA Thema's en Kernprincipes v1.0 p.12
ID	K1.2
Scope	Generic

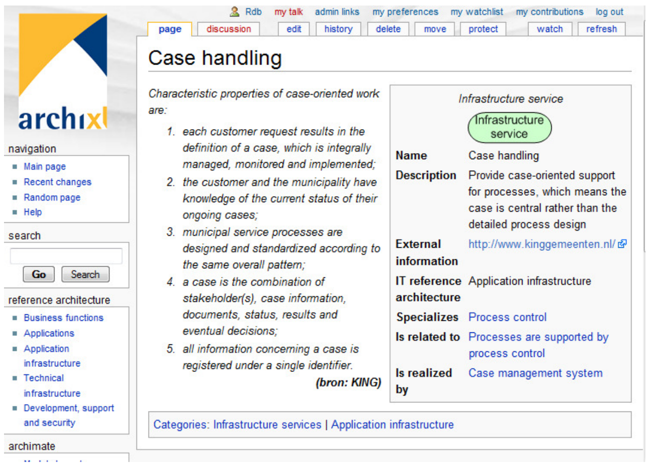
Category: Core principles

Figure 2. Example of a principle in a semantic wiki

The motivation and implications properties are somewhat special in that they consist of links to other pages in the wiki. Moreover, only the 'motivation' property has been recorded directly on this page. The implications listed have been inferred from the information recorded at the corresponding implication pages. Inferred relations (motivations as well as implications) are calculated by the semantic wiki itself. A reciprocal relationship between two principles needs to be recorded only once to be known on both pages. For this, the implication and motivation relations are modeled as each other's inverse. In other words: When principle A implies principle B, then (part of) B's rationale is principle A.

As with principles, every ArchiMate element is mapped onto a dedicated wiki page. Figure 3 shows the page for an

ArchiMate infrastructure service “Case handling”. This page nicely demonstrates how the wiki may be used to combine structured information (the infobox with semantic properties at the right hand side) and unstructured information (the block of text at the left hand side). Here again, the infobox shows relations directly registered on this page (e.g., “specializes”) as well as relations derived from information registered on other pages (e.g., “is realized by”).



Case handling

Characteristic properties of case-oriented work are:

1. *each customer request results in the definition of a case, which is integrally managed, monitored and implemented;*
2. *the customer and the municipality have knowledge of the current status of their ongoing cases;*
3. *municipal service processes are designed and standardized according to the same overall pattern;*
4. *a case is the combination of stakeholder(s), case information, documents, status, results and eventual decisions;*
5. *all information concerning a case is registered under a single identifier.*

(bron: KING)

Infrastructure service

Name	Case handling
Description	Provide case-oriented support for processes, which means the case is central rather than the detailed process design
External information	http://www.kinggemeenten.nl/
IT reference architecture	Application infrastructure
Specializes	Process control
Is related to	Processes are supported by process control
Is realized by	Case management system

Categories: Infrastructure services | Application infrastructure

Figure 3. Example of an ArchiMate element in a semantic wiki

A. *Case: GEMMA - Municipalities*

In the Netherlands, municipalities constitute the government layer that is closest to citizens and businesses. Within a broader vision of a reduction of administrative burdens for and better service provision to citizens and businesses, there is an ongoing trend towards municipalities as a ‘one stop shop’, i.e., a single point of contact for ‘the’ government. As a consequence, the internal organization of municipalities and their information landscape is changing. The Municipal Model Architecture (GEMMA) is one of the instruments in establishing this change.

GEMMA consists of seven topics that correspond to thematic changes and developments within municipalities. Examples of these topics are ‘Case- and process-oriented work’, which implies that logical units of work are treated as a single case, and ‘The municipality becomes the gateway to the government’, which implies among others that municipalities establish customer contact centers.

Within each of these themes, various principles have been defined, distinguishing between so-called core principles, process architecture principles, information architecture principles, and design principles [9], [10], [11]. The design principles are linked to information functions. Each information function can be seen as representing an (implicit) principle, namely the principle that the municipality should arrange facilities to support certain information functions. The identification of municipality-wide information functions such as customer contact management, case management, and registration system management - is a direct implication of one of GEMMA’s information architecture principles which states that “our municipality [...] uses generic facilities for generic information functions”. Additionally, the individual information functions stem from the different themes: the need for setting up facilities for case management, for example, is an implication of the desire to employ case- and process-oriented work (Theme 1 of GEMMA).

Even though the categorization of GEMMA principles suggests some sort of layered ordering, the internal structure of the set of GEMMA principles remains very much implicit. This makes it fairly hard to determine which principles are relevant to – i.e., should be reused and further extended in – a particular IT project within a municipality. Adding to the complexity is the fact that GEMMA contains over 500 principles. To alleviate this problem, we analyzed each of the principles by asking the questions “why?” and “how?”: why should we conform to this principle (in other words, what is its motivation?) and how is this principle realised (in other words, what are its implications?). By explicitly asking these questions, and recording their answers in a semantic wiki, the implicit structure of the set of GEMMA principles emerged as a “network” of principles. For instance, Figure 4 shows the principle network for case management for the “why”-question, and Figure 5 for the “how” question. Nota bene: both figures have been generated by the wiki based on the information recorded on its pages.

Using any principle as a starting point, the network of principles can hence be traversed in two directions: ‘upwards’, towards the underlying reasons (“Why?”) and ‘downwards’, towards the implications (“How?”). Having identified the structure of the set of GEMMA principles, it becomes more apparent where an organization-specific architecture should extend GEMMA: we should be particularly interested in the ultimate implications of GEMMA, i.e., the ‘leaf nodes’ in the principle network. In the figure above, these are the eight design principles ‘Service oriented process support’ to ‘Case information is split in process- and content-related’. For each of these implications, we must determine their impact on our organization-specific architecture.

One of the strengths of Semantic MediaWiki is that it can be used to query other data sources as well. In particular, we can query other instances of Semantic MediaWiki. This establishes a system of semantic wikis that together constitute an inter-organizational architectural knowledge management platform.

Figure 6 shows a link in this system of semantic wikis between a municipality’s wiki and a reference architecture wiki containing GEMMA. The municipality’s wiki queries the reference architecture wiki to determine the ultimate implications of, for instance, case management. One of these implications is the design principle “Every case is stored in a case warehouse”. For each of the ultimate implications,

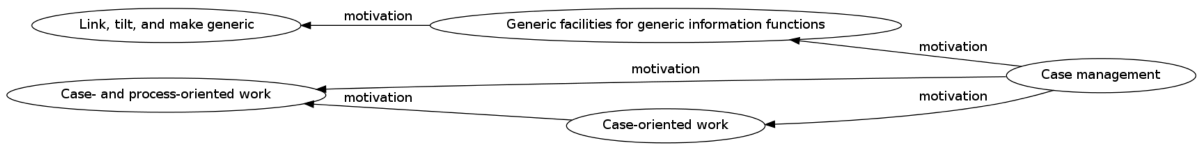

Link, tilt, and make generic
motivation
Generic facilities for generic information functions
motivation
Case management
motivation
Case- and process-oriented work
motivation
motivation
Case-oriented work

Figure 4. Motivation for case management

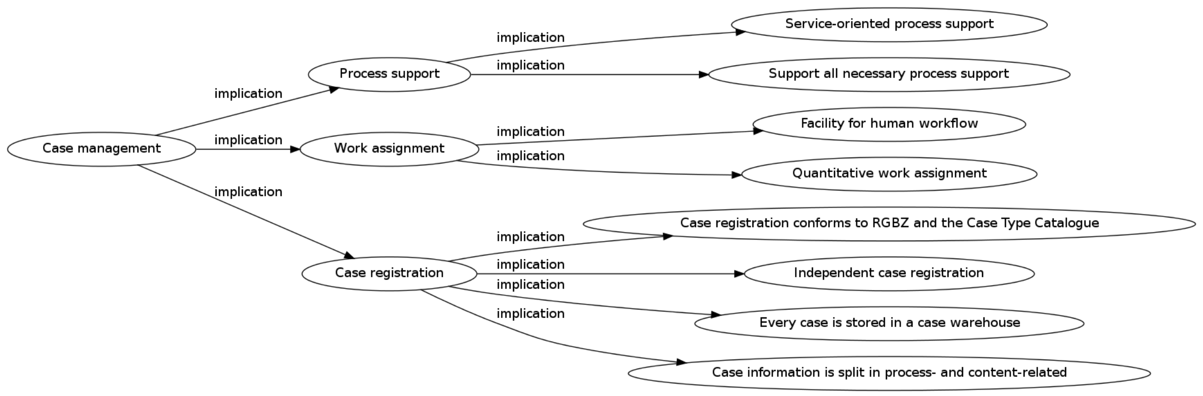

Case management
implication
Process support
implication
Service-oriented process support
implication
Support all necessary process support
implication
Work assignment
implication
Facility for human workflow
implication
Quantitative work assignment
implication
Case registration
implication
Case registration conforms to RGBZ and the Case Type Catalogue
implication
Independent case registration
implication
Every case is stored in a case warehouse
implication
Case information is split in process- and content-related

Figure 5. Implications of case management

a page is generated in the municipality wiki that acts as a proxy to the information in the reference architecture wiki.

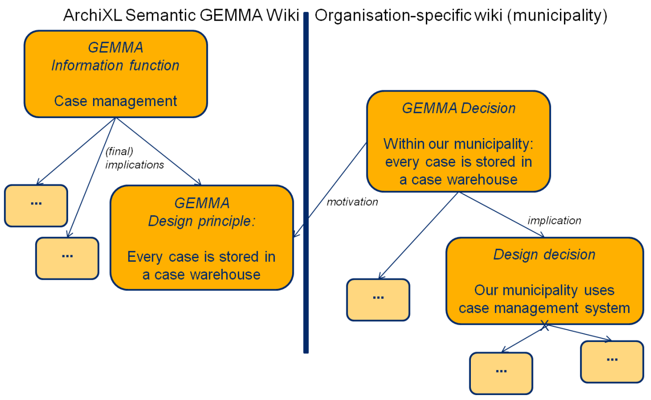
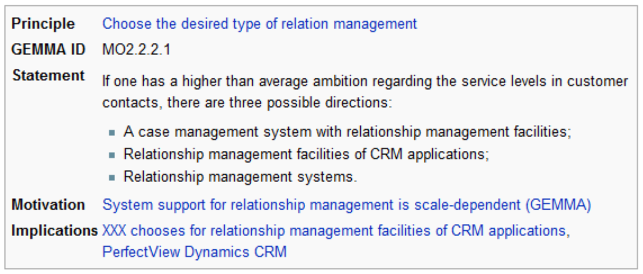

ArchiXL Semantic GEMMA Wiki
Organisation-specific wiki (municipality)
GEMMA Information function Case management
(final) implications
...
...
GEMMA Design principle: Every case is stored in a case warehouse
motivation
GEMMA Decision Within our municipality: every case is stored in a case warehouse
implication
...
Design decision Our municipality uses case management system
...
...

Figure 6. Reuse of reference architecture knowledge

The links between the organization-specific wiki and the generic reference architecture wiki allow us to further control the implementation of GEMMA within the municipality. Figure 7 shows the (anonymized) information box of a decision on one of the GEMMA design principles within a municipality. GEMMA makes clear that each municipality must make a choice for the type of relationship management. The specific choices made by the municipality are registered in their wiki as implications of this design principle.

The status of the implementation of GEMMA in this municipality is displayed in the wiki using overviews such as the one shown in Figure 8. Some implications are formulated as decisions, while others are still in the stage of (research) questions. These overviews are dynamically generated based on the knowledge registered in the wiki, and always offer insight into the current state of affairs.



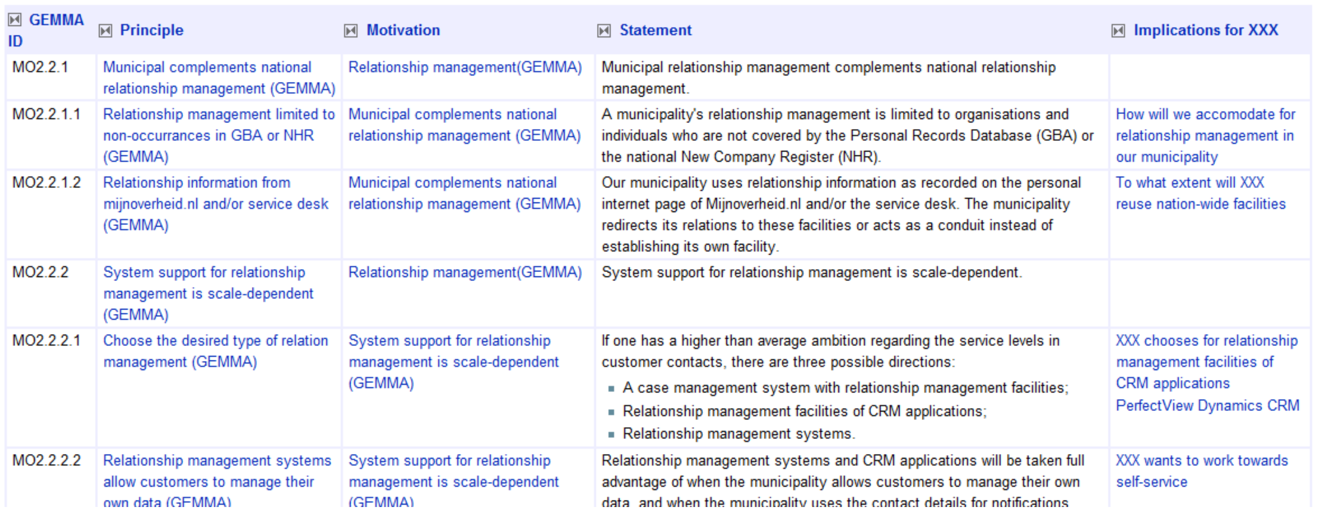
Principle	Choose the desired type of relation management
GEMMA ID	MO2.2.2.1
Statement	If one has a higher than average ambition regarding the service levels in customer contacts, there are three possible directions: ▪ A case management system with relationship management facilities; ▪ Relationship management facilities of CRM applications; ▪ Relationship management systems.
Motivation	System support for relationship management is scale-dependent (GEMMA)
Implications	XXX chooses for relationship management facilities of CRM applications, PerfectView Dynamics CRM

Figure 7. GEMMA Decision

Since organization-specific architectural knowledge is directly linked to the underlying reference architecture knowledge, it is also easier to address change management. As soon as things change in the reference architecture wiki (new principles appear, or existing principles are adapted to new insights) these changes could be identified from the organization-specific wiki. The architect may then be warned that that changes have occurred and be pointed to the area where those changes affect the municipality's enterprise architecture.

B. Case: ROSA - the Education Sector

Like many governmental fields, the education sector is characterized by chained collaborations between various stakeholders. The education chain includes such stakeholders as the ministry of education, executing agencies that implement the ministry's policies, education councils, and educational institutions to name but a few.



GEMMA ID	Principle	Motivation	Statement	Implications for XXX
MO2.2.1	Municipal complements national relationship management (GEMMA)	Relationship management(GEMMA)	Municipal relationship management complements national relationship management.	
MO2.2.1.1	Relationship management limited to non-occurrances in GBA or NHR (GEMMA)	Municipal complements national relationship management (GEMMA)	A municipality's relationship management is limited to organisations and individuals who are not covered by the Personal Records Database (GBA) or the national New Company Register (NHR).	How will we accomodate for relationship management in our municipality
MO2.2.1.2	Relationship information from mijnoverheid.nl and/or service desk (GEMMA)	Municipal complements national relationship management (GEMMA)	Our municipality uses relationship information as recorded on the personal internet page of Mijnoverheid.nl and/or the service desk. The municipality redirects its relations to these facilities or acts as a conduit instead of establishing its own facility.	To what extent will XXX reuse nation-wide facilities
MO2.2.2	System support for relationship management is scale-dependent (GEMMA)	Relationship management(GEMMA)	System support for relationship management is scale-dependent.	
MO2.2.2.1	Choose the desired type of relation management (GEMMA)	System support for relationship management is scale-dependent (GEMMA)	If one has a higher than average ambition regarding the service levels in customer contacts, there are three possible directions: ▪ A case management system with relationship management facilities; ▪ Relationship management facilities of CRM applications; ▪ Relationship management systems.	XXX chooses for relationship management facilities of CRM applications PerfectView Dynamics CRM
MO2.2.2.2	Relationship management systems allow customers to manage their own data (GEMMA)	System support for relationship management is scale-dependent (GEMMA)	Relationship management systems and CRM applications will be taken full advantage of when the municipality allows customers to manage their own data, and when the municipality uses the contact details for notifications	XXX wants to work towards self-service

Figure 8.

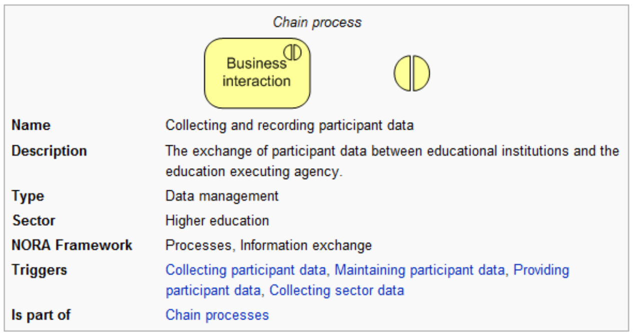
Led by the ministry, stakeholders within the Dutch education sector want to reach mutual agreements to facilitate the exchange of information between organizations and the development of a common ICT infrastructure. To this end, the Education Reference Architecture (ROSA) has been created, based on the principles found in the Dutch Government Reference Architecture (NORA). The first version of ROSA was adopted in 2007, followed by version 1.1 in 2008 [12].

Work on a new version of ROSA is currently in progress. To support architectural knowledge management and enhance the reuse of ROSA reference architecture knowledge by other stakeholders in the field, the ministry has opted to use a semantic wiki for further development and publication of ROSA.

Like GEMMA, ROSA is not an architecture of a single enterprise. Instead, ROSA provides insight into the chain processes that run throughout the education sector as a whole. Organization-specific business processes are only touched upon at the highest level. These processes need to be filled in by individual organizations through their own enterprise architectures. Hence, ROSA does not describe the internal organization of enterprises in the education sector, but instead describes the ways in which these organizations are interrelated and the available building blocks and standards for information exchange between the different parties involved.

Figure 9 shows the knowledge model upon which the ROSA semantic wiki has been based. It consists of four parts: 'why', 'what', 'who', and 'how'. The 'why' is expressed by means of architecture principles which are in turn motivated by drivers for change. The 'what' and 'who' consist of architectural components and stakeholder representations based on ArchiMate elements. The 'how' is represented by projects that involve construction of concrete architectural components, especially building blocks.

In this knowledge model, chain processes – which are modeled as ArchiMate business interactions – lead to business processes which in turn may make use of one or more IT services. These are a superset of ArchiMate's application services and infrastructure services, and represent externally visible units of functionality delivered by application components or system software. The application components and system software that realize the IT services are referred to as 'building blocks'. Some elements (standards, data objects) are maintained externally and imported into the wiki.



Name	Collecting and recording participant data
Description	The exchange of participant data between educational institutions and the education executing agency.
Type	Data management
Sector	Higher education
NORA Framework	Processes, Information exchange
Triggers	Collecting participant data, Maintaining participant data, Providing participant data, Collecting sector data
Is part of	Chain processes

Figure 10. Infobox of a chain process

Figure 10 shows the infobox of a chain process, taken from the page 'Collecting and recording participant data'[4] in the ROSA wiki. The chain process leads to four business processes: educational institutions are responsible for 'Collecting partipant data', 'Maintaining participant data', and 'Providing participant data'; the Ministry, through one of its executing agencies, is responsible for 'Collecting sector data' (which includes participant data) and further processing this data. The way in which the collected data are further processed is not considered to be part of the chain process, however. Likewise, *how* the parties involved should

[4] 'Participants' are students who participate in the provided education.

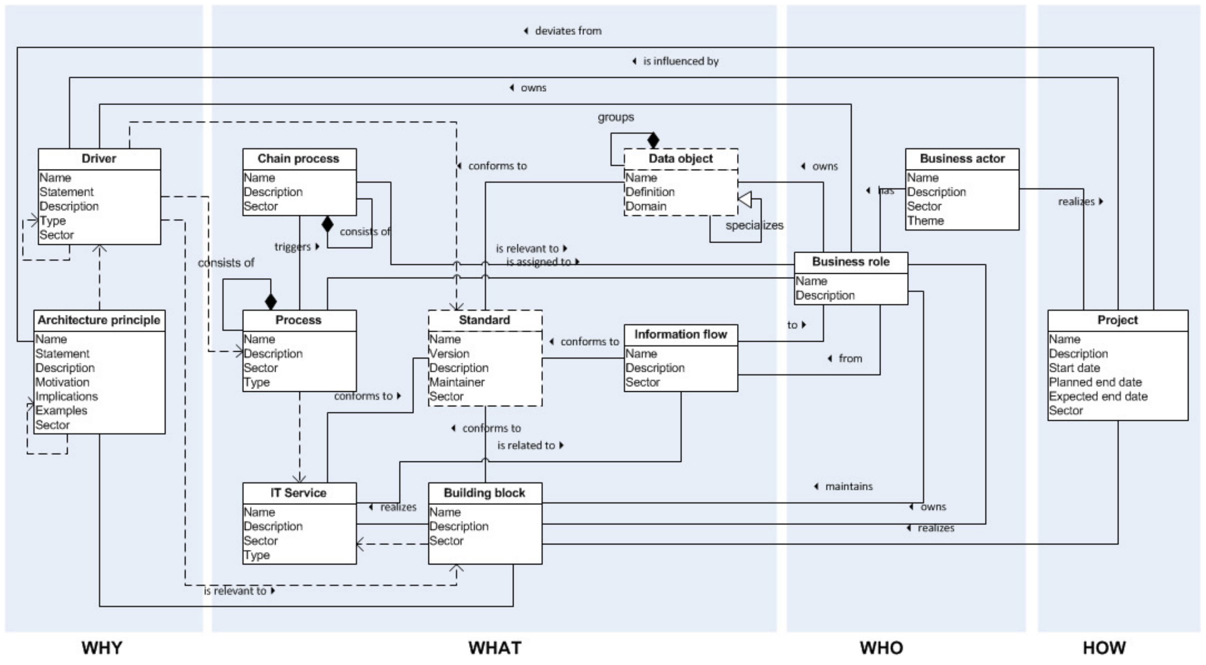


Figure 9. ROSA Knowledge Model

collect, maintain or provide the data is not further detailed in ROSA. Each of the parties involved has complete freedom of (and complete responsibility for) the way in which they implement these processes. There are, however, a number of additional architectural components that ROSA prescribes in order to facilitate the necessary information exchange.

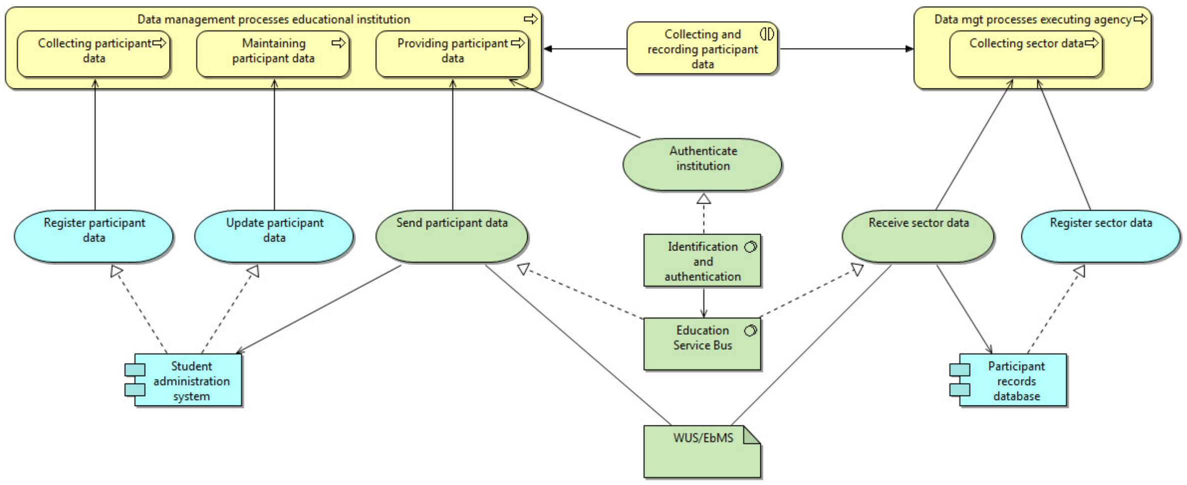
Figure 11 shows an ArchiMate model of the complete chain process and its constituents. Each of the model elements corresponds to a single page in the wiki. The figure shows that four (logical) building blocks are involved in this chain process: a student administration system that realizes services for registering and updating participant data for a single institution; an identification and authentication component that realizes a service to authenticate educational institutions; an education service bus that provides service to send and receive data; and a participant registry that realizes a service to register sector data about education participants.

One of the intended uses of the ROSA semantic wiki is for different stakeholders in the education sector to understand what their responsibilities are in the various chain processes, and what building blocks they are expected to use in order to realize the necessary services. In the above example, educational institutions are expected to use a student administration system that is able to connect to an education service bus using the right data exchange standards. The ministry's executing agency, on the other hand, is expected to be able to receive data through said service bus, and to use a registry that stores participant data for the whole sector. All building blocks are expected to comply with relevant standards and architecture principles.

To support different views on the content of the ROSA wiki, we extended the semantic wiki with a project start architecture (PSA) generator. The PSA generator requires only a limited number of input parameters, namely: a chain process, a sector (primary, secondary, tertiary education for educational institutions or sector-independent for other parties such as executing agencies) and the architecture layer(s) of interest (business, information, technology). Based on these inputs, the semantic annotations according to the knowledge model from Figure 9 are queried to determine which pages in the wiki - i.e., principles, processes, services, building blocks, standards, projects - are relevant to the situation at hand. The PSA generator then presents an overview of the contents of these pages in a single report. This report can be used by the requesting party to align their organization-specific enterprise architecture with ROSA, analoguous to the reuse of GEMMA principles we described in Section III-A. The organization-specific architectural knowledge can again be stored in its own dedicated semantic wiki and through interwiki links be linked to the relevant pages from the ROSA wiki.

IV. ARCHITECTURAL KNOWLEDGE AND SOFTWARE DOCUMENTATION

A problem frequently encountered in software development environments, especially when they are distributed over multiple sites, is the accessibility of documents. Designers do not know what the latest set of requirements is. Architects cannot trace requirements to design documents. Developers cannot find the appropriate information in the myriad of data repositories that typically exist in the organization, ranging from word files to meeting minutes and Requisite Pro



Data management processes educational institution
Collecting participant data
Maintaining participant data
Providing participant data
Collecting and recording participant data
Data mgt processes executing agency
Collecting sector data
Authenticate institution
Register participant data
Update participant data
Send participant data
Identification and authentication
Receive sector data
Register sector data
Education Service Bus
Student administration system
Participant records database
WUS/EbMS

Figure 11. ArchiMate model of the chain process 'Collecting and recording participant data'

databases. One avenue we pursue is to store the information in a semantic wiki, and provide powerful search facilities to query the wiki pages.

We are currently experimenting with capturing requirements and architectural documents in a semantic wiki (viz. OntoWiki), and provide traceability between them. This goal is comparable to the desired traceability between (and within) reference architecture principles and model elements discussed in Section III. The main difference is the type of knowledge elements considered. In this case, each section of a (requirements or architecture) document is captured in a wiki page. The underlying ontology to index the knowledge captured has elements such as "requirement", "component", "decision", and so on. Attributes of a requirement include its id and proposer. Similarly, a decision is characterized by, among others, the issue it addresses, its arguments, and the resulting architectural structure.

Once such information is stored in the semantic wiki, one needs powerful search facilities to be able to query the information in meaningful ways. An important requirement of such search facilities is that it has to cater for transitive relations. For instance, if a person "John" has proposed some subset of requirements, and each requirement relates to certain design decisions, it must be possible to search for the design decisions that are linked to "John", even when this information is not directly captured. We use the semantic search web-server ClioPatria for this purpose [13]. A more elaborate description of our approach is given in [14].

V. Lessons learned

We have presented two e-government cases in which we have gained experience with the use of semantic wikis for architectural knowledge management. We have also briefly discussed our ongoing experiments with the use of semantic wikis to capture and trace software requirements and software architecture documentation. From our experiences in these (and other) cases, we have learned several lessons regarding the use of semantic wikis for architectural knowledge management.

A. *Satisfaction of requirements*

Let us start our discussion by revisiting the requirements we listed in Section I. These are the requirements that an inter-organizational or multi-site architectural knowledge management platform should satisfy. Semantic wikis contribute to the satisfaction of each of these requirements:

Manage architectural decisions: The GEMMA case has shown how a system of semantic wikis can support municipal architects in their decision making process. By connecting a municipality's organization-specific wiki to a generic GEMMA reference architecture wiki, architects can keep track of the implementation of GEMMA within their own municipality. Moreover, the use of semantic links between principle pages in the GEMMA wiki results in an explicit hierarchy of architecture principles that makes it easier for the architect to understand which parts of GEMMA should be further extended in the municipality's enterprise architecture.

Codify architectural knowledge: By their very nature, semantic wikis provide support for the codification of structured information on a wiki page. Moreover, semantic wikis may also support the combination of unstructured and structured information on a single page.

Search architectural knowledge: A huge advantage of semantic wikis over more traditional, document-based codification is the fact that semantic wikis can be queried for information and can contain overviews (lists, tables, graphs, et cetera) that are dynamically updated as soon as the knowledge codified in the wiki changes.

Support community building: Wikis are well-known for having the ability to attract and help establish a community. An interesting side-effect of using semantic wikis for managing - and especially for publishing - architectural knowledge, is that other departments can quite easily find and browse through information about the organization's architecture. We have witnessed how departments that previously had difficulties understanding architectural information, suddenly found they understood "what the architecture was all about". We have also observed situations in which departments whose tasks were sideways related to the work of the architecture team (e.g., project management or service management) found enough valuable information in the wiki that they wanted to add and link their own domain knowledge to the architectural knowledge already codified in the wiki.

Provide intelligent support: The ROSA case has shown how a PSA generator based on the semantic wiki's knowledge model can provide intelligent support to determine the parts of the reference architecture that pertain to a particular situation. Additionally, maintaining explicit links between organization-specific and generic architectural knowledge may be used to warn architects of changes in a reference architecture that impact certain parts of their architecture.

Enrich architectural knowledge: Although originally we primarily meant (semi-)automatic enrichment of architectural knowledge, e.g., through the use of text mining services, we have come to believe that the opportunities semantic wikis provide for manual enrichment of architectural knowledge are a good middle ground. We have seen several forms of enrichment, most notably the explicitation of the - hitherto implicit - structure of a set of reference architecture principles and the reuse and extension of generic reference architecture knowledge in an organization-specific context.

B. Points of attention

Our experiences have also highlighted some points that require attention when one considers the use of a semantic wiki for architectural knowledge management:

Level of formality: The level of formality with which knowledge is codified may differ between semantic wikis. Some semantic wikis - such as Semantic MediaWiki - are primarily content-oriented. In these wikis, textual information is annotated with semantic properties. Other semantic wikis - such as OntoWiki - emphasize formal knowledge engineering activities. These wikis may provide better support for some tasks such as logical reasoning, at the cost of a steeper learning curve and less support for unstructured or semi-structured knowledge.

Amount of codification: It is of paramount importance to develop a clear idea of which architectural knowledge should be codified before one actually starts codification. For this, it is useful to consider the use cases that the semantic wiki should support. A common pitfall is to attempt to be (over)complete. Metamodels often tend to focus on all concepts and relations that could be recognized. To prevent a maintenance nightmare, however, one needs to limit the semantic wiki's knowledge model to only those concepts and relations that actually pertain to the questions the wiki should be able to answer.

Inferencing capabilities: Not all use cases for searching and (re)using architectural knowledge may be supported out of the box. Most of the functionality we described has been based on standard functionality provided through open source semantic wiki components. In some cases, however, we had to write additional extensions to overcome limitations in the standard inferencing capabilities. This has been the case for the PSA generator and for the particular query to remotely determine ultimate implications for GEMMA.

Graphical modeling: It should be clear that a wiki is not the right tool for graphical modeling. Although a semantic wiki can generate, even for architectural components and their relationships, simple graphical overviews such as the ones shown in Figures 4 and 5, more elaborate models should be drawn in other environments. These environments may have their own internal repositories and associated semantics. The semantics from these environments can be reused in the wiki, e.g. through export/import mechanisms, but one should make clear choices about which environment is the leading source for which architectural knowledge.

User rights management and access control: Wikis are inherently open platforms. Their main goal is to support and encourage collaboration and knowledge sharing. Consequently, most wikis have only limited support for user rights management. When sophisticated, role-based access control to individual knowledge entities is a necessity, semantic wikis may not be the best knowledge management platform available.

VI. Research Challenges and Future Work

Even though our applications of semantic wikis focus on different domains and target different aspects of architectural knowledge management, the emerging research challenges touch upon the same themes:

Knowledge model alignment: when knowledge from one semantic wiki is reused in another, as is the case with organization-specific architecture wikis that are linked to a generic reference architecture wiki, the knowledge models for these wikis must be aligned to the extent that the organization-specific wiki 'understands' the generic architectural knowledge. Alignment of knowledge models is also important from a community perspective, since it allows different parties to learn from each other and reuse knowledge even at a meta level. At the same time, different organizations have different needs regarding the wiki's knowledge model and the type of architectural knowledge that is recorded. Terminology and desired knowledge classifications differ from organization to organization, for instance because different architecture frameworks are used. The level of desired detail with which architectural knowledge is recorded may also differ. This means that some properties or knowledge entities may be important for one organization, but ignored by another.

Likewise, the ontology used to support software documentation contains generic software engineering elements like "requirement" and "decision". In an actual environment, one might want to also include domain entities, like "mortgage" for a financial domain, and "municipality" in the governmental domain. An open question is whether it is more appropriate to have separate ontologies and link them, or combine them in a single ontology.

To support different knowledge needs while preserving alignment between knowledge models, we envision the construction of knowledge model 'product lines' - analoguous to software product lines - that contain well-defined knowledge variation points.

Knowledge versioning: in a semantic wiki, architectural knowledge is always 'live' and may be continuously updated. Still, there may be a need to occasionally refer to the architecture as it was at a certain point in time. Moreover, different versions of the same architecture play a role in activities such as scenario-based analysis ("what if. . . ?") and gap analysis ('ist' vs. 'soll' comparisons). Activities such as these are an important aspect of architectural work. In the software documentation example, versioning support is essential; different development projects within one and the same organization might use different versions of requirements documents, design documents, and the like.

To support the need to refer to different versions of the same architecture, we envision a versioning approach to architectural knowledge within a semantic wiki similar to the branching and tagging versioning mechanisms well-known from software versioning systems. Ideally, such a versioning system would also be able to calculate the difference between two versions of the architectural knowledge, and feature sophisticated merging facilities that enable consolidation of two branches, e.g., incorporate the result of a what-if scenario in the actual architecture.

Knowledge updates: another type of versioning issue occurs when semantic wikis are linked and reuse each other's architectural knowledge. When a reference architecture changes - e.g., parts of it are updated or even a major new version is released - all architectures that depend on it should eventually change accordingly. These changes cannot and should not be automatically processed in the dependent architectures: the architect has the final say in the impact these changes have. Architects should, however, be notified of the changes and the areas of the architecture that are impacted. A similar situation occurs when the knowledge captured concerns software documentation: when requirements change, updates will be required for the corresponding design elements, software components, and the like.

We envision a notification and change management service that pro-actively monitors changes in the knowledge captured and determines the impact thereof on other parts. Architects and other stakeholders should be notified of these changes, so that they may take appropriate action.

VII. Conclusions

In this paper, we have reflected on our experiments with using semantic wikis for architectural knowledge management. Our applications of semantic wikis focus on different domains and target different aspects of architectural knowledge management. ArchiXL focuses on organizing and structuring architectural knowledge for reuse in e-government. The VU University focuses on searching and querying architectural knowledge for distributed software development. Nevertheless, the emerging research challenges we have identified touch upon the same themes: knowledge model alignment, knowledge versioning, and knowledge updates. We expect these themes to be fruitful areas of research that bring added value to the theory and practice of architectural knowledge management.

References

[1] A. Jansen, "Architectural design decisions," Ph.D. dissertation, University of Groningen, 2008.

[2] R. Farenhorst and R. C. de Boer, "Architectural knowledge management: Supporting architects and auditors," Ph.D. dissertation, VU University Amsterdam, 2009.

[3] M. Ali Babar, T. Dingsøyr, P. Lago, and H. van Vliet, Eds., *Software Architecture Knowledge Management: Theory and Practice*. Springer, 2009.

[4] P. Kruchten, "An Ontology of Architectural Design Decisions in Software-Intensive Systems," in *2nd Groningen Workshop on Software Variability Management*, Groningen, NL, 2004.

[5] P. Lago, R. Farenhorst, P. Avgeriou, R. C. de Boer, V. Clerc, A. Jansen, and H. van Vliet, "The GRIFFIN Collaborative Virtual Community for Architectural Knowledge Management," in *Collaborative Software Engineering*, I. Mistrík, J. Grundy, A. van der Hoek, and J. Whitehead, Eds. Springer, 2010, pp. 195–217.

[6] The Open Group, *TOGAF Version 9*. Van Haren Publishing, 2009.

[7] D. Greefhorst and E. Proper, *Architecture Principles: The Cornerstone of Enterprise Architecture*. Springer, To appear.

[8] The Open Group, *ArchiMate 1.0 Specification*. Van Haren Publishing, 2009.

[9] Architectuurteam KING, "GEMMA Thema's en Kernprincipes voor gemeentelijke proces- en informatiearchitectuur," KING, Tech. Rep., 2009.

[10] ——, "GEMMA-procesarchitectuur: Principes, modellen en standaarden voor het inrichten van gemeentelijke dienstverleningsprocessen," KING, Tech. Rep., 2009.

[11] ——, "GEMMA-informatiearchitectuur: Dienstverlening door de gemeente," KING, Tech. Rep., 2009.

[12] B. Gaakeer, "Referentiearchitectuur Onderwijs," Ministerie OCW, Tech. Rep. versie 1.1, 2008.

[13] VU University, Dept. of Computer Science, "The ClioPatria semantic search web-server," http://e-culture.multimedian.nl/software/ClioPatria.shtml.

[14] A. Tang, P. Liang, and H. van Vliet, "Software Architecture Documentation: The Road Ahead," in *WICSA 2011*, In print.