Evaluation of Architectural Knowledge Management from Process Decisions perspective in SOA

T07796

Developed by:
Shamsa Bano
139-FBAS/MSSE/F06

Supervised by:
Dr. Naveed Ikram

Co Supervisor:
Ms. Muneera Bano

Department of Software Engineering
Faculty of Basic and Applied Sciences
International Islamic University Islamabad
[2010]

Department of Software Engineering

International Islamic University Islamabad

Date: 21-04-2011

Final Approval

This is to certify that we have read the thesis submitted by SHAMSA BANO, Reg.No.139-FBAS/MSSE/F06. It is our judgment that this thesis is of sufficient standard to warrant its acceptance by International Islamic University, Islamabad for the degree of MSSE.

Committee:

External Examiner:

Prof. Dr. Arshad Ali Shahid,
Department of Computer Sciences
FAST NU,
Islamabad

[illegible]

Internal Examiner:

Mr. Mata-ur-Rehman,
Assistant Professor,
DSE, FBAS, IIUI

[illegible]

Supervisor:

Dr. Naveed Ikram,
Associate Professor,
DSE, FBAS, IIUI.

[illegible]

Co-Supervisor

Ms. Muneera Bano,
Assistant Professor
DSE, FBAS, IIUI

[illegible]

Dedicated to

My

Loving ***Ami*** **and** ***Abou***

Who encouraged me in every
step of my life and without
them I am nothing.

A dissertation Submitted To

Department of Computer Science,

Faculty of Basic and Applied Sciences,

International Islamic University, Islamabad

as a Partial Fulfillment of the Requirement for the Award of the

Degree of MSSE.

Declaration

I hereby declare that this Thesis **"Evaluation of Architectural Knowledge Management from Process Decision perspective in SOA"** neither as a whole nor as a part has been copied out from any source. It is further declared that I have done this research with the accompanied report entirely on the basis of my personal efforts, under the proficient guidance of my teachers especially my supervisor Dr. Naveed Ikram (Associate Professor) and co supervisor Ms. Muneera Bano (Assistant Professor). If any part of the system is proved to be copied out from any source or found to be reproduction of any project from any of the training institute or educational institutions, I shell stand by the consequences.

[illegible]

[Shamsa Bano]

[139-FBAS/MSSE/F06.]

Acknowledgement

First of all I am obliged to Allah Almighty the Merciful, the Beneficent and the source of all Knowledge, for granting me the courage and knowledge to complete this Thesis.

I want to pay my special regards to my kind and inspiring supervisor Dr. Naveed Ikram (Associate Professor) and co supervisor Ms. Muneera Bano (Assistant Professor), whose guidance and worthy suggestions enable me to make this humble contribution to the vast world of research.

In the end, I like to express my heartiest gratitude to my loving parents, sister, and bother in law and my dear niece for their heartily prayers and spiritual support. I am nothing without their prayers.

[illegible]

[Shamsa Bano]

[139-FBAS/MSSE/F06.]

ABSTRACT

Architecture knowledge management (AKM) makes it possible that the knowledge within the S/W architecture is documented in such a way that it should help in decisions making process for the upcoming S/W projects of an organization. Design decisions and their rationales are two important ingredients of AKM. A number of knowledge models have been proposed to manage such an important knowledge, which plays vital role in the success of any organization in IT Industry. Design decisions are of different types such as *Structural design decisions* and *Process design decisions.* All the models that have been proposed in reported literature are not capable enough to document process decisions, due to limited attention paid on them. This thesis work focuses specifically on Service Oriented Architecture (SOA) and has validated the knowledge model presented by Patricia Lago and Qing GU. In their model they have suggested some extensions in the core model for architecture knowledge management and claims that this knowledge model with these extensions is able to document the process decisions.

This thesis has validated the aforesaid claim in SOA domain. This is done with the help of two conducted case studies (two international projects) that suites the research context. These case studies are used as examples, because, they have been conducted in near past and their data is used as secondary data. The process decisions from two international projects are mapped against this core knowledge model with extensions, and the results show that the claim is genuinely feasible.

List of Figures

List of Tables

Abbreviations

SE	Software Engineering.
AK	Architecture Knowledge.
KM	Knowledge Management.
SCS	Service Centric System.
QoS	Quality of Service.
ECM	Extended Core Model.
SOA	Service Oriented Architecture.
SOC	Service Oriented Computing.
SLA	Service Level Agreement.
RSD	Run-time Service Discovery.
ASD	Architectural-time Service Discovery.
AKM	Architecture knowledge management.
SOSE	Service oriented software engineering.
SeCSE	Service Centric Software Engineering.
UDDI	Universal Description Discovery and Integration.
SENSORIA	Software Engineering for Service-Oriented Overlay Computers.

Table of contents

1. INTRODUCTION

Today we are living in a world, in which knowledge plays the most important role in the long-term success of both an individual and an organization. From past few years' software industry has recognized the importance of knowledge management especially in the area of software architecture. Software architecting is a process that generates rich amount of information [1]. Organizations in IT sector are in great need to manage this information. They should manage this knowledge, exploit it effectively and re-use it. If any organization is failed to manage such an important knowledge then it will be required to re-invest huge amount of fund, time and allocation of large resources to collect the forgotten information about the previously decided as well as discarded architectural design decisions and their rationale [1]. The management of architectural knowledge is important in every form of software architecture such as, in traditional as well as in service oriented architecture (SOA).

SOA emerges as a new and popular paradigm in the software architecture industry, it allows us to organize and utilize different services that perform different functionalities and are available over the internet. These services are loosely coupled and of distributed nature that raise the problem of knowledge management in SOA. One of the important tasks in service oriented environment is the selection of the most relevant services and its integration to the system and this can by accomplished with the help of properly managed knowledge.

1.1. Service Oriented Architecture

Service oriented software engineering (SOSE) is a new paradigm in software engineering [4]. It is becoming popular with every passing minute, which reveal the usefulness of this entirely new domain. The term Service is the key element in SOSE, which can be used for the development of applications and software solutions. It can be viewed as an individual software component, a module or a function that is performed by a software system or application. It is a set of activities to perform a specific task e.g., providing a route map in an automotive car system, viewing an online bank statement, etc. SOSE provides functions in the form of loosely coupled services that are available on a network. [4].

SOA is a paradigm that makes it possible that these loosely coupled services are available over the network. It allows the exchange of data between different applications. An organization can integrate such loosely coupled services to make the software application that can fulfill their requirements.

One can say SOA is an architectural style that allows the organization and utilization of different services available over the network. It is a platform through which one can discover and integrate related services to develop any application or support the business process of an organization.

The benefits of the SOA compelled the organizations to adapt it for the prevailing advantages and the utilities.

1.1.1 Motivation for SOA

As said in the previous section, that SOA is an architectural style that allows different services available over the network. It elevates the idea of distributed computing (i.e. request/reply design style).

In SOA, business logic and different individual functions of an application are modularized and presented as loosely coupled services i.e. service interface is independent of the implementation technology. One can build his/her own application by integrating one or more such loosely coupled service without knowing their underlying implementations.

Other aspects that plays important role in the popularity of SOA are: the re-usability through which it improves the productivity and reduces the time to market, integration and composition of services that enable SOA which could quickly response to the changing requirements. The service contracts define the interaction agreements to avoid the misunderstanding between the service provider and consumer. Autonomy and abstraction hides the service logic. The flexibility provided by SOA makes it easy to implement the complex software systems. SOA has a well-defined interface that facilitate the consumer to invoke it easily and in a standard way

1.1.2 Challenges of SOA

This new paradigm is facing some challenges for example, SOA governance due to its distributive nature [5], lack of testing tools [42], some security issues of service based applications [6], lack of development tools [42], and since it is a new paradigm there is also shortage of experienced people in this area [42]. Most importantly it is also facing the problems in managing the architectural knowledge because; the existing knowledge models for AKM are not as effective in SOA as in traditional S/W architecture [9].

1.2. Knowledge Management

We are living in an age in which organizations as well as individuals are heavily dependent on knowledge/information for their business. Knowledge is prerequisite requirement to compete with rivals and competitors as well as meeting the phase with prevailing market conditions for growth. So, the management of organizational knowledge appeared as the main factor in the long-term success of organizations.

Different people define the term knowledge in different ways such as: It is the expertise and skill that a person can have or acquired through his/her practical experience, education and through the environment in which he/she is working. [45], while some people said it is the practical capability that a person can have, others say that it is the collection of past and present experiences plus the skill of a person in his/her field [45].
So, we can define *Knowledge* as the ability that a person can have on the bases of his/her past and present experience and education. It is the learning power to know leading facts and information that a person would have to face in a particular field. It is very important for an organization that it manages its knowledge in such a way that it helps in meeting both of its existing and expecting needs or requirements.

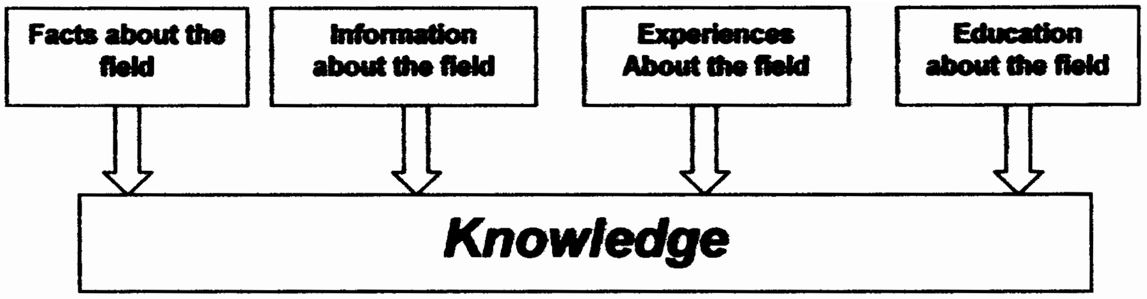
Facts about the field
Information about the field
Experiences About the field
Education about the field
Knowledge

Figure 1.2: Ingredients of knowledge

By the term *knowledge management* we mean, the storage or preservation of knowledge that includes previous experience, information and data events in an organization. This stored knowledge can be used in future events and also in decision-making process [12].

1.2.1 Architectural knowledge (AK)

Architectural knowledge includes the knowledge within the S/W architecture. It is vital for not only improving the quality of the architecture but also the quality of architecting process as well [14].

AK includes two important elements in it i.e., design decisions and the reasons (i.e., rationale) behind these decisions, that will help us to find out why a particular decision is the way it is [3].

1.2.1.1 Types of Architectural knowledge

AK is subdivided into two main categories, which are

- Knowledge that is not documented,i.e., tacit knowledge, and
- Knowledge that is formulized or documented, i.e., Explict knowledge.

Implicit or Tacit Knowledge

It is the personal knowledge that one gets from his/her experience, blief and through system and working and/or social enviornment. It is that type of knowledge that is not visible and difficult to express and communicate with others. The problem with this kind of knowledge is that it is not documented in any form and remains in the head of its creator [3].

Explicit Knowledge

It is the type of knowledge that is formally codified in documented form. In other words, it is the written explanation containing statement and details for the completion of a specific job with an adequate, sound and befitting knowledge to excersize by a person [3].

1.2.1.2 Strategies for knowledge management

Strategy for knowledge management is the way that an organization follows to manage their organizational knowledge. There are two main strategies for knowledge management, which are

- Codification, and
- Personalization.

Codification

In this kind of strategy, knowledge is managed by codification in a documented way. Codification results in reliable knowledge that can be re-used in current and future projects. It enables us to convert tacit knowledge into explicit knowledge [16] [17] [18] [19].

Personalization

The strategy of knowledge management in which the experience and knowledge (in a particular field) of people and organizations is shared through interaction among them is called personalization. Unlike codification in such kind of strategy for KM, knowledge remains with the knowledge creator [16] [17] [18] [19].

1.2.2 Architectural knowledge management and SOA

The management of knowledge is emerged as an important factor in the software industry. Software architecture is highly knowledge intensive process that requires and produces large and rich amount of information. Failure in managing such a rich amount of information/knowledge will result in the wastage of both time and money because we have to allocate some of expensive resources and human effort to recollect the forgotten information or previously decided or discarded architectural design options [1].

An organization which is lacking with a displined approach for managing architectural knowledge will face the complex process for system evolution, inedequate clarification of arguments and information sharing about the design process and also fails to identify the design errors [13].

As said previously that SOA is facing the problems in the management of architecture knowledge. The existing knowledge models for this purpose are not as effective in SOA as in traditional software architecture; they are not capable enough to document different types of design decisions (such as process decision) in SOA [9].

1.2.3 Reasons for not managing architecture knowledge

The main objective of knowledge management is to use and exploit different knowledge resources of both an individual and an organization. These knowledge resources can be skills, capabilities, different technologies and context (such as rules, conditions) in which the work is done [16].

Different researchers explained the importance of architectural knowledge management in different ways, such as, Patricia Lago [1] states S/W architecting is a process that requires and produces large amount of information and failure in managing such a rich amount of knowledge, will result in wastage of both time and money, to recollect the forgotten information about previously decided and discarded design decisions. Hans van Vliet, M. Babar and Ian Gorton highlighted the importance of architectural knowledge during software system development, when different stakeholders (Architects, Developers etc) have to communicate about the system under development. They say that it is the stage when only a well-organized architectural knowledge results in a system that fulfill user requirement [16] [1] [22].

There are different reasons for not managing the architectural knowledge, for example, lack of motivation for AKM, usually people involved in the architecting process have short term interest and they don't even bother to document this knowledge. Most of the people involved in S/W architecting process don't know how to document this knowledge and if it is documented in any form then it is not shared properly with the concerned people as a result it is not utilized as it should be [2].

1.3. Research Objctive

The research objective of this thesis work is to validate one of the architectural knowledge model presented by Qing Gu and Patricia [9] for documenting process decisions in SOA. This is because they claim that this model is capable enough to document process decision in SOA. The research question for the research work is

- Up to what extent the "Extended core model (ECM) [9]" for architectural knowledge management is documenting the process decisions made in SOA?

1.4. Process decisions

These are the decisions that are related to the development process. These are one type of architectural design decisions [20]. The reason to choose process decisions as the main theme for research work is that these are the design decisions that are ignored by almost all knowledge models for the management of architectural knowledge. They are paid very little attention as compared to other types of design decisions e.g. structural design decisions. The importance of process decisions in SOA is more than other types of software architecture because, it is a new and immature type of software architecture and researchers are still trying to develop a proper development process for it [9]. The popularity of this new paradigm compels us to work in this area to make it more useful for the software industry.

1.5. Research Process

The research process includes the following steps to answer the research question

1. Study of literatre related to the research i.e., Literature review
2. Investigation of different knowledge models for documentation of process decsoions in SOA.
3. Validation of ECM for documenting process decisions in SOA

1.6. Thesis Outline

The structure of this thesis is as follows:

.

Chapter 2:	This chapter gives brief detail about the literature survay of this research/thesis work.
Chapter 3:	This chapter performs the problem analysis and map different architectural knowledge models.
Chapter 4:	It expains the research design and provide brief introduction about the two case studies that are used to conduct this reseach work.
Chapter 5:	This chapter performs the validation of the Extended Core Model (ECM) for process decisions in SOA, which is the main objective of this research work and also provides the final conclusion.
Chapter 6:	This chapter provides a conclusion by discussing the contribution of this thesis and discusses the possible directions for future research that help out to improve AKM.

2. LITERATURE SURVEY

Knowledge management is very important in traditional S/W architecture as well as in service-oriented architecture. Since SOA is a new field and researchers are still working on the development process for it, so it requires special attention in the field of knowledge management. At the one end SOA allows us to organize and utilize different services over the Internet through loose coupling and distributed nature and at the other end this facility lead us to the problem of knowledge management. In service-oriented environment it is very important to choose the most relevant service and integrate it in the right manner so that it enhances the productivity of our software system.

Software architecture is a collection of different types of design decisions such as structural decision, and process decisions etc. the focus of attention of this thesis work is the documentation of process decisions in SOA, whereas process decisions are one type of architectural design decision. Philippe Kruchten [20] presents ontology of different types of architecture design decisions, according to which *process decisions are sub type of executive design decisions that are related to the development process, and affect the involved people and the business environment of the organization. They are not directly related to the design element or their qualities.*

2.1. SOA and AKM

The major issues and challenges that SOA is currently facing are actually rapidly increasing due to lack of the management of architectural knowledge. SOA domain is suffering with the ignorance in the management of architectural knowledge because very little work is done in this area. Number of knowledge models has been proposed to manage the architectural knowledge in software industry that can be seen in the next chapter. Architectural knowledge is the result of decision-making process that includes the design decisions and their rationale as the main factors. In SOA this decision making process indicates some concerns that require specific information, which needs to be properly documented. The main problem that the SOA is currently facing is the rare use of these models for the management of architectural knowledge. The current knowledge models are failed to properly document the information (generated in the decision making process) in SOA domain. They are not able to address all types of design decisions in SOA i.e. fail to document process decisions. As a result this information cannot be used in the future events or decision-making processes [10]. This is the gap identified by *Qing Gu and Patricia Lago* in their paper [9], which serves as the base paper for this thesis work. This paper suggests an extension in the core model for AKM and claims that core model after this extension is capable for documenting the process decisions in SOA. We call the core model with these suggested extensions as "Extended core model (ECM)" and use this term in the rest of the thesis document.

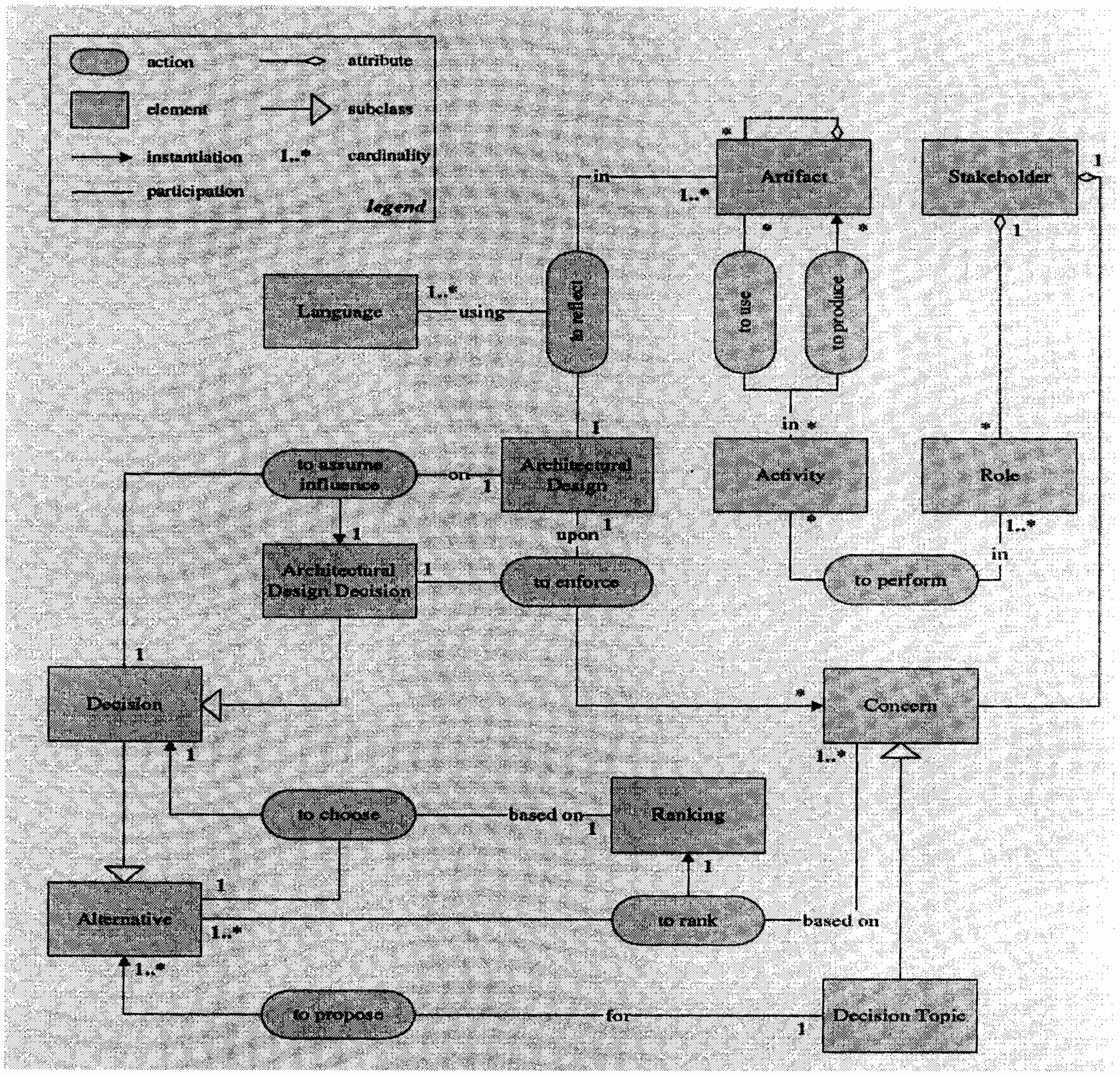

Figure 2.1: Core Model of Architectural Knowledge

They elicit two sample process decisions from the deliverables of SeCSE methodology and use table2.1 as conceptual model that contains concepts required to document process decisions. They map those concepts onto existing knowledge models and also on the core knowledge model to check that whether theses could be used to document the process decisions in SOA. As a result of that mapping the authors' claims that the existing knowledge models are not capable enough to document process decisions in SOA [9]. The core model for knowledge management [23] was used as a frame of reference [fig 2.1].

Concern	Issue on which the architectural design decision should be made.	
Ranking Criteria	Shows the priorities of the user (such as flexibility, performance etc.) and helps in decision-making.	
Architectural Decision Alternative	**Identifier**	A sequence number and a title of the decision.
	Description	Description of the decision
	Executor of the decision	Stakeholder who react on the decision
	Stakeholders Who are related to this decision	Stakeholders who are influenced by the decision
	Status	The state of the decision as accepted or rejected or under discussion, is recorded here
	Rationale	Pros and Cons of the decisions are explained here.
	Relationship (s)	Relationship of the current decision with other decisions is indicated here.

Table 2.1: Table used for mapping existing Knowledge models

When they map the above table with the core knowledge model, they were able to locate most of the concepts except the two i.e. "Executor of the decision", and "Related stakeholders". So Qing Gu and Patricia Lago suggest an extension in the core knowledge model [fig 2.2] for architectural knowledge and named it as "Extended core knowledge model (ECM)" for AKM. They claim that the core model with this extension (fig2.2) can be used to document the process decisions in SOA.

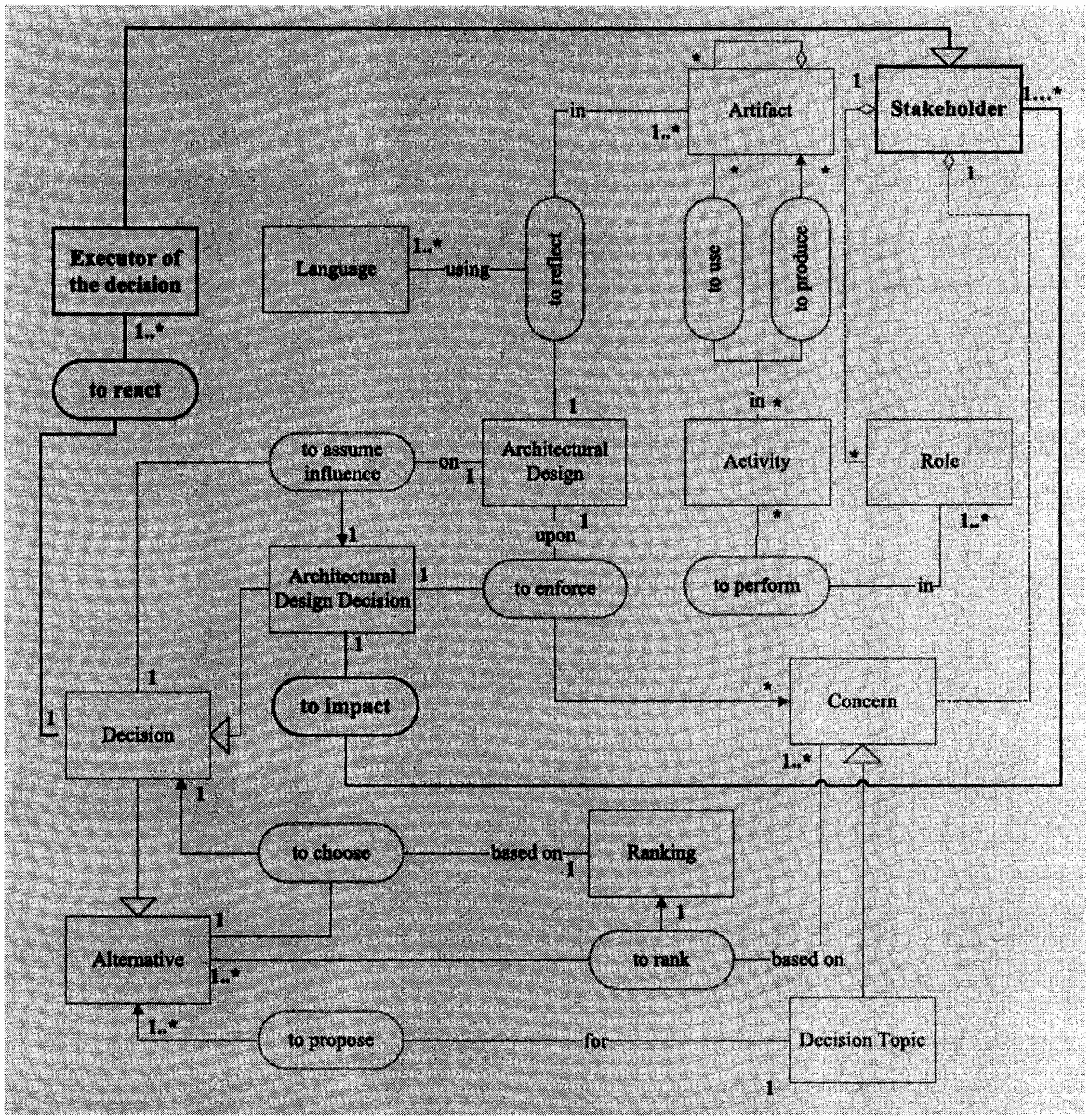

Figure 2.2: Extended Core Model of Architectural Knowledge (ECM)

(Ref taken from" SOA Process Decisions: New Challenges in Architectural knowledge Modeling [9]")

3. PROBLEM ANALYSIS

This chapter will analyze the claim of Patricia Lago and Qing Gu [9] declaring that the "current knowledge models are not capable enough to document Process Decisions in SOA".To justify their claim whether correct or not, I taken into consideration their claim and examine different concepts used to document Process Decision (as given in table 2.1) with the existing knowledge models. The hypothesis emphasis that if these concepts are mapped with in whole to the existing knowledge models then these could be considered fit to use document Process Decision.

Table 3.2 shows that all the existing models are partially mapped with the concepts used. There is not a single model that can document complete knowledge as required to document the "Process Decisions". Thus as per claim of Patricia Lago and Qing Gu that current Knowledge models are not capable for the purpose stands correct.

3.1 Mapping of existing models for AKM

This mapping includes five models that were used by Patricia Lago and Qing Gu [9] together with five more knowledge models, which validate above-mentioned claim. Table 3.1 give brief introduction of ten existing knowledge models used for mapping purpose.

1. Kruchten's Ontology [20]	**Kruchten presents this ontology for complex software systems. In this ontology "Design decisions" are divided into three categories such as *Existence*, *Property* and *Executive* decisions. These decisions have some common characteristics and dependencies with other decisions. This Ontology also shows that design decisions are not interrelated with each other's but also related with artifacts such as requirements etc.**
2. Tyree Decision Template [24]	**It is one of the early efforts in the field of architectural design decisions. Tyree tried to model ADD as a text template.**
3. Archium Model [38]	**Jansen et al presents this model, it described architecture as a set of decisions and consists on three sub models: *architectural model* (describe the S/W architecture), *design decision* (describe design decision) model and *composition model* (represent the required concepts to unit the two previous models).**
4. Case Model [3]	**Hans van Vliet et al presents a use case model together with the underlying design decision ontology.**
5. SOAD Model [25]	**Zimmermann et al. present this model and suggest architectural decision modeling as a technique for service realization. They organized the SOA knowledge in a decision tree, comprising of a *conceptual, technology and product/open source asset* level by using MDA (Model-Driven Architecture) principles**
6. S/W architecture Project Memories [26]	**Remco et al [26] presents a high-level architectural knowledge model for structuring software architecture project memories.**
7. Core Model [23]	**Patricia Lago, Han van Vliet et al presents this model in order to over come issues such as use of different terms to describe concepts with similar and identical meanings etc.**
8. Pattern-based Model [39]	**Neil et al present a pattern-based model and tried to reduce the efforts for recording design decisions with the help of patterns.**
9. ADDSS Model [40]	**Capilla et al. present this model for architecting and evolving ADD. It consists of three main parts: *project model* i.e. information related to the S/W architecture project, architecture *model* i.e. represent S/W architecture, which is mainly explain as a set of component and connectors and *decision model* i.e. represent the attributes of the decision as mandatory and optional attributes.**
10. AREL Model [41]	**AREL (Architecture Rationale and Element Linkage) is knowledge model that based on rationales i.e. it focuses on the reasons that led to take design decisions. Three key elements of this model are *Design Concern, Design Decisions and Design Outcomes*. It links the problem space with the solution space through design decision in a uniform way.**

Table 3.1 Ten Existing Knowledge Models for AKM

3.1.1 Mapping Results

This section provides mapping results with variations that were used to document "Process Decision in SOA".

Concern

The concept "Concern" represents the issue on which the architectural design decision is made. It is same as "Issue" in Tyree decision template, "Problem" in Archium model and Pattren-based model, "Concern and Decision Topic" in Core model and S/W Architecture Project Memories and "Decision Concern" in AREL model.

Ranking Criteria

This concept represents the criteria through which a particular design decision is selected from different design decisions to solve a particular problem or concern. "Ranking" in S/W Architecture Project Memories and Core Model is equal to this concept. None of the other models provides such information.

Architectural Decision Alternative

"Architectural Decision Alternative" tells about different design decisions to solve the problem. "Design Decision" in Kruchten's Ontology, Use case model and AREL model, "Alternative" in S/W Architecture Project Memories and Core Model, and "Architectural Design Decisions" in ADDSS model all provides this information and are equal to this concept.

Identifier

The concept of "Identifier" represents the title of the current design decision. "Epitome" in Kruchten's Ontology and Use case model is same to this concept. Further in "Decision" in Tyree Decision Template, S/W Architecture Project Memories, Core model, "SOA Decision" in SOAD model, "Solution" in Pattern-based model and "Architectural Design Decision" in ADDSS model are also same as this concept.

Description

This concept provides description of the current decision. "Context" in Pattern-based model and "Description" in Archium model are equal to this concept.

Executor of Decision

The concept of "Executor of Decision" is equal to "Author" in Kruchten's Ontology and Use Case model. None of the other eight models provide such information.

Related Stakeholders

This concept provides information about the stakeholders who are influenced by the design decision. None of the ten models provides such information.

Status

The status (i.e. accepted, rejected or under discussion) of the design decision is represented by this concept. It is equal to "Status" in Kruchten's Ontology, Tyree Decision Template, Use Case model, Pattern-based model and ADDSS model. There is no such concept in Archium model, SOAD model, S/W Architecture Project Memories, Core model and in AREL model.

Rationale

The concept of "Rationale" i.e. the reason why the design decision is made, is equal to "Pros and Cons" in Archium model, "Rationale" in Kruchten's Ontology, Use Case model, Pattern-based model and ADDSS model. In AREL "Qualitative and Quantative Design Rationale" capture this concept. There is no explicit concept in Core model that provides such information, but rational can be discovered by following decision loop in this model.

Relationship

This concept describes the link between the design decisions. "Related Decision" in Tyree Decision Template, "Decision Dependency" in SOAD model, "Related Pattern" in Pattern-based model, "Dependency" in AREL model and "Relationship" in Kruchten's Ontology are equal to this concept. Other models do not provide such information.

Concept Name / Model	***Kruchten's Ontology***	***Tyree Decision Template***	***Archlum Model***	***Use Case Model***	***SOAD Model***	***Project Memories***	***Core Model***	***Pattern-based model***	***ADDSS***	***AREL***
Concern	---	Issue	Problem	---	---	Concern, Decision Topics	Concern, Decision Topics	Problem		Decision Concern
Ranking Criteria	---	---	---	---	---	Ranking	Ranking	---	---	---
Architectural Decision Alternative	Design Decision	Decision	Solution	Design Decision	Pattern	Alternatives	Alternatives	Solution	Architectural Design Decisions	Design Decision
Identifier	Epitome	Decision	Decision	Epitome	SOA Decision	Decisions	Decision	Solution	Architectural Design Decisions	Design Decision
Description	---		Description	---	---	---	---	Context	---	---
Executor of Decision	Author	---	---	Author	---	---	---	---	---	---
Stakeholders who are related to this decision	---	---	---	---	---	---	---	---	---	---
Status	Status	Status	---	Status	---	---	---	Status	Status	---
Rationale	Rationale	---	Pros, Cons	Rationale	---	---	---	Rationale	Rationale, Pros, Cons	Design Rationales (Quantitative, Qualitative)
Relationship	Relationship	Related Decisions	---	---	Decision Dependency	---	---	Related Pattern	Dependencies	---

Table 3.2 Mapping of Existing Knowledge Models for AKM

* *The highlighted fields are the concepts that are added in the core model for AKM for documenting process decisions*

4. RESEARCH DESIGN

The main objective of this research work is to validate an architectural knowledge model presented in [9] for documentation of "Process Decision in SOA". In this respect, I conduct the research work with the help of two case studies named below. These case studies have been conducted in the near past and contain work suitable in context of my research work. These case studies are helpful, supportive and validate my research work and used as example. Data provided is also used as secondary data for validation of my research work.

Moreover these case studies selected for the purpose to reveals research goal from real life scenario, provide complete holistic view of the required domain and also helpful to understand previous activities taken place in SOA domain. The aforesaid two case studies used by me for my research work are supportive to give strength to validate my work more conducive and with authenticity.

The selected case studies are of exploratory type. The reasons to choose only these two projects are their accessibility, relevance, depth of knowledge, and most importantly availability of required data for the conduction of the research work.

The unit of analysis for this research work is the "Extended core model (ECM)" for architectural knowledge management presented in [9]. The data is collected through the study of different deliverables of the two case studies. The collected data includes the process decisions and their related concerns.

4.1. Case studies

Two international projects are used as the existing case studies. These two international projects are "EU-funded integrated project called SeCSE (Service-Centric Systems Engineering)" and "European Community funded project SENSORIA (Software Engineering for Service-Oriented Overlay Computers)".

4.1.1 SeCSE Project [43]

SeCSE is a European Integrated project. The aim of this project is to facilitate system integrators and service providers by creating different tools, methods and techniques for the development of service centric applications. It also supports cost-effective development of such applications. The objective of this project includes the extension in the existing approaches used for the service and system specification. The goals of SeCSE project covers different phases of software development such as analysis, design etc.

It was considerably large project that includes 17 partners from 8 countries (e.g. Italy UK, Czech Republic, Belgium, Spain, Germany and Netherlands) and its budget ranges from 9.3 – 15.3 million Euros. It was started on Sept 2004 and the end date was Aug 2008. The results achieved from this project were experimented in the European automotive and telecommunication sectors.

4.1.1.1 Data Collection

The team involved in this international project i.e. SeCSE collect data from different ways in order to achieve data triangulation. The different ways that were employed to collect data are: interviews with the stakeholders, study of related documents.

Interviews

SeCSE Team organize semi-structured interviews (with both open and close end questions) with the stakeholders involved in this project, in order to acquire information on issues related to services oriented systems, e.g., the services discovery, services integration, service monitoring etc.

Documents

A number of related documents were also studied to get information about different approaches, methods, techniques and tools for analysis, design and reasoning about the service centric system.

4.1.1.2 Approach

To achieve the desire results the SeCSE project was organized in two groups of activities i.e. Problem activities and Project activities, as shown in the fig 4.1 and 4.2.

Problem Activities

These activities focus on four areas of software engineering i.e. specification, discovery, design and management of services for which new techniques and tools will be provided.

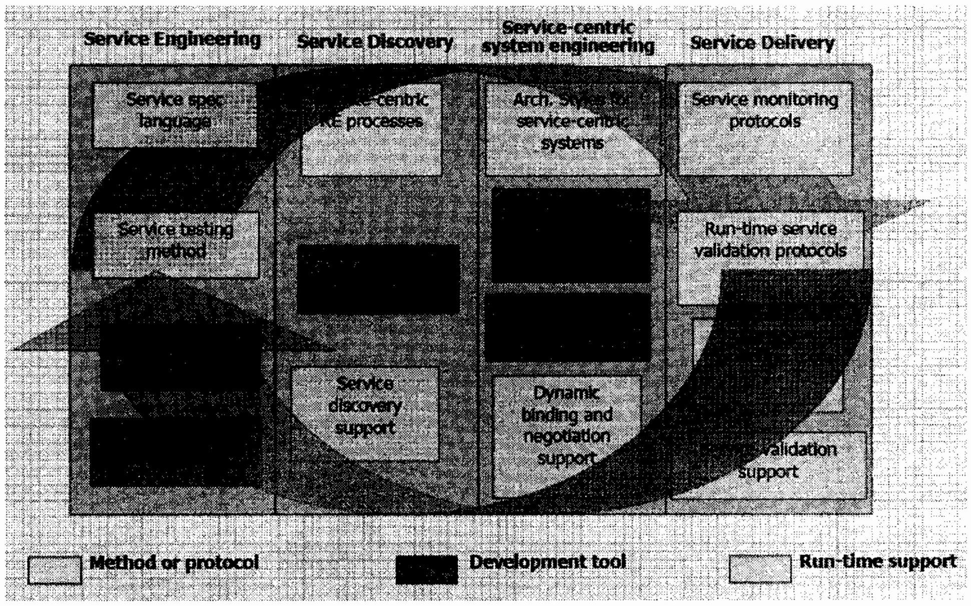
Service Engineering
Service Discovery
Service-centric system engineering
Service Delivery
Service spec language
-centric E processes
Arch. Styles for service-centric systems
Service monitoring protocols
Service testing method
Run-time service validation protocols
Service discovery support
Dynamic binding and negotiation support
alidation support
Method or protocol
Development tool
Run-time support

Figure 4.1: SeCSE Development Environment

***(Ref taken from "SeCSE** website http://secse.eng.it [43]")*

Project Activities

These activities take the results from problem activities and then integrate them in a coherent framework, and improve their applicability by using selected application scenarios.

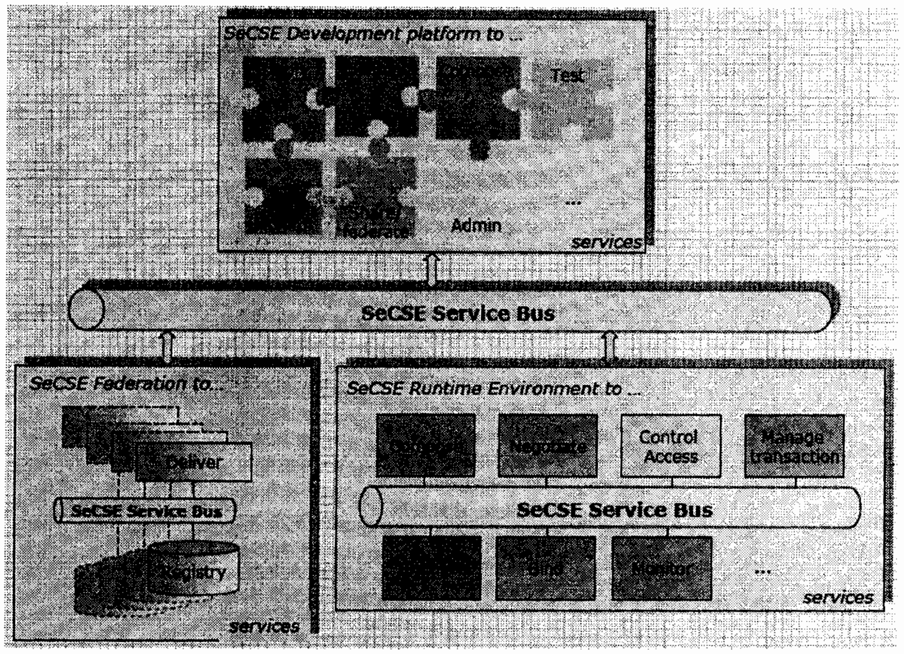

Figure 4.2: SeCES Project Activities

(Ref taken from" SeCSE website http://secse.eng.it [43]")

4.1.2 SENSORIA Project [44]

It is an Integrated Project funded by the European Community. It develops methodologies and tools to deals with the problems in service oriented computing (SOC). SENSORIA covers whole life cycle of software development. The results of this project were applied on numbers of real life case studies from areas such as telecommunications, e-business, e-learning and automotive systems.

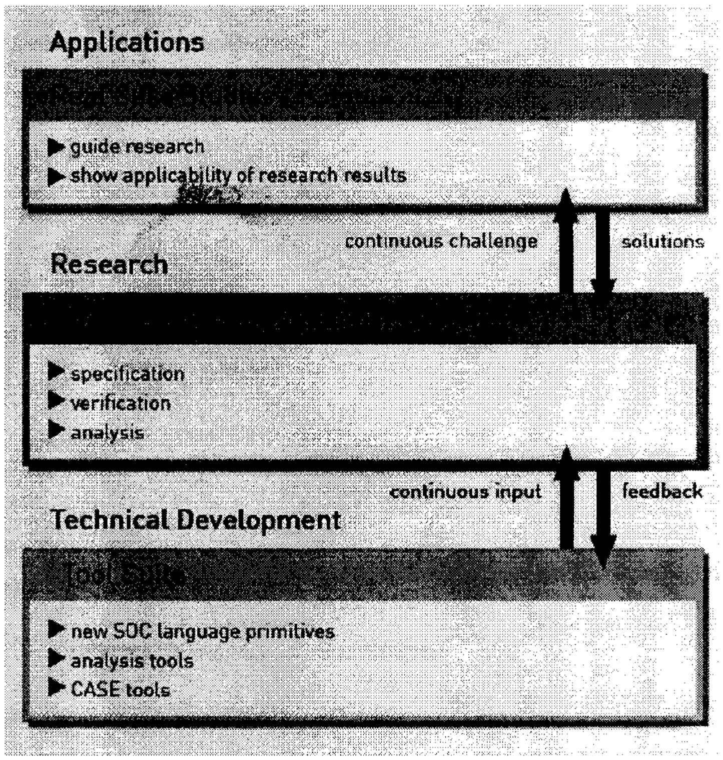

Figure 4.3: SENSORIA Project Lifecycle

(Ref taken from " SENSORIA website http://www.sensoriapist.eu [44]")

4.1.2.1 Data collection

The team involved in this international project i.e. SENSORIA collect data from different ways in order to achieve data triangulation. The different ways that were employed to collect data are: interviews with the stakeholders, study of related documents.

Interviews

SENSORIA Team organize semi-structured interviews (with both open and close end questions) with the stakeholders involved in this project, in order to acquire information on issues related to services oriented systems, e.g., the language used to model and program services oriented systems, quantitative and qualitative analysis methods for global services, development and deployment techniques for system services etc.

Documents

A number of related documents were also studied to get information about the above-mentioned things about service-oriented systems.

First of all service models were translated into formal representation by using automated model transformation. Then tools developed in SENSORIA were used to check functional correctness of services, early performance analysis and verify the applicability of service level agreement. Finally, results of the mathematical analysis were made available in the model to provide feedback to the developer.

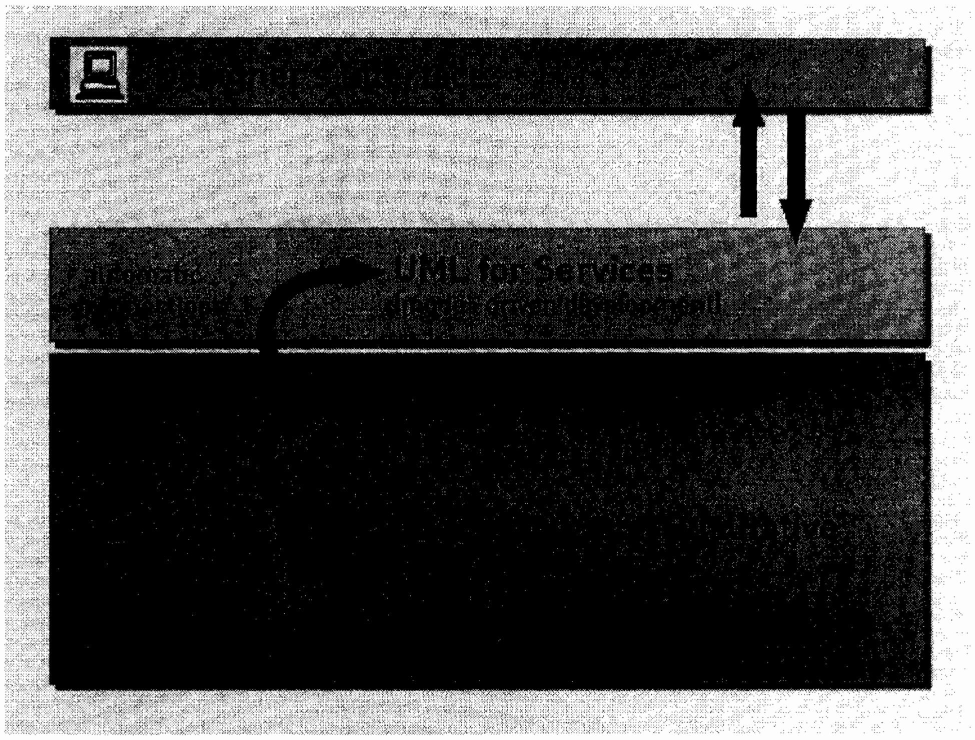

Figure 4.4: SENSORIA Project Approach

(Ref taken from" *SENSORIA website* *http://www.sensoriapist.eu [44]")*

5. VALIDATION

In order to achieve desire results of this research work, the data is collected from the two case studies in the form of some concerns; these concerns are the issues on which the process decisions are made for the required service oriented architecture. These identified concerns and their related process decisions are then mapped on the table 2.1.

5.1. The identified concerns from SeCSE project [43]

- How to extract service discovery queries (that specify the functionality and quality properties) from architecture and design models of a service centric system?
- How to integrate the discovered services into an iterative design process of service centric system?
- Who will generate the queries to retrieve the alternate service that can satisfy the violated requirements?

5.2. The identified concerns from SENSORIA project [44]

- Does the system support different types of service discoveries?
- Who is responsible to perform the task of monitoring in order to ensure that the discovered services are meeting the QoS requirements and Service Level Agreement (SLA) and also fulfilling the consumer's requirements?
- Who will generate the modes to retrieve the alternate service (that can satisfy the violated requirements) and at which time (i.e. at architecture time or at run time)?

5.3. Mapping of concerns from Case Studies

Now these identified concerns and their related process decisions are mapped on the table 2.1as presented in chapter 2.

SeCSE Project [43]

Concern	Extraction of service discovery queries (that specify the functionality and quality properties) from architecture and design models of service centric system.	
Ranking Criteria	Flexibility, reliability, recoverability, performance	
Architectural Decision Alternative	**Identifier**	**A-1.** Architecture- time service discovery (ASD)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints.
	Executor of The decision	System Designer
	Stakeholders Who are related to this decision	**1.Service Developer:** Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available. **2.Service Provider:** If provider, provides service of functionalities that requester needs, it must be locatable.
	Status	Approved
	Rationale	**Pros:** • The new and modified version of design models enable the designer to specify more queries to discover other services that can satisfy more detailed functionality of the system. • The result of this process is a complete specification of the SCS architecture models.
	Relationship (s)	

Table 5.1: concern no.1 from SeCSE Project.

Concern	Integrating the discovered services into an iterative design process of service centric system.	
Ranking Criteria	Flexibility, Reliability, Recoverability, Performance	
Architectural Decision Alternative	**Identifier**	A-2.Architecture- time service discovery (ASD)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints
	Executor of the decision	System Designer
	Stakeholders Who are related to this decision	**1.Service Developer**: Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available. **2.Service Provider**: If provider, provides service of functionalities that requester needs, it must be locatable. **3.Service Integrator**: Integrate the discovered services properly.
	Status	Approved
	Rationale	**Pros:** • The new and modified version of design models enable the designer to specify more queries to discover other services that can satisfy more detailed functionality of the system. • The result of this process is a complete specification of the SCS architecture models.
	Relationship (s)	

Table 5.2: concern no.2 from SeCSE Project.

Concern	Generation of queries to retrieve alternative service that can satisfy the violated requirements.	
Ranking Criteria	Flexibility, Reliability, Recoverability, Performance	
Architectural Decision Alternative	**Identifier**	A-3.Run - time service discovery (RSD)
	Description	Alternate services are discovered at execution time to replace the services, which are already integrated to the SCS. This is done when already integrated services are failed to provide the required functionality and quality requirements.
	Executor of the decision	System Designer
	Stakeholders Who are related to this decision	**1.Service Developer:** Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available. **2.Service Provider:** If provider, provides service of functionalities that requester needs, it must be locatable.
	Status	Approved
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software.
	Relationship (s)	#2.a in the Base paper [9]. (Service monitoring is performed by the service provider and the service consumer.)

Table 5.3: concern no. 3 from SeCSE Project.

SENSORIA Project [44]

Concern	The system should be able to support different types of service discovery	
Ranking Criteria	Flexibility, accessibility, availability, performance	
Architectural Decision Alternative	**Identifier**	**1.a.** Static service discovery (architecture-time service discovery)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints. The location for such services are hard coded in the software
	Executor of the decision	System designer (who request for the service)
	Stakeholders Who are related to this decision	**1.Service Developer:** Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available. **2.Service Provider:** If provider, provides service of functionalities that requester needs, it must be locatable.
	Status	Discarded
	Rationale	**Pros:** • Discovery of service through local repository has better run-time performance because service consumers don't have to communicate with service registry at run-time. **Cons:** • Service consumers are unaware about the services that have similar functionality and better qualities in the service registry.
	Relationship (s)	

Architectural Decision Alternative	**Identifier**	1. b. Dynamic service discovery (Runtime service discovery).
	Description	Services are searched or discovered dynamically at run time by Search engine.
	Executor of the decision	Service discovery engine (UDDI discovery engine).
	Stakeholders Who are related to this decision	**1.Service broker and Service Requester:** Monitors the services (already integrated to the system) against the predefined SLA and QoS and initiates the process. **2.Service Developer:** Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available.
	Status	Discarded
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software. **Cons:** • Run-time performance of the system is affected by the communication between service consumer and service registry.
	Relationship (s)	
Architectural Decision Alternative	**Identifier**	1.c. Combination of both static and dynamic service discovery

	Description	Adopts the combination of both static and dynamic service discovery. Where static service discovery is used at design time and dynamic service is used at run time in case of error occurrence.
	Executor of the decision	System designer and Service discovery engine (UDDI discovery engine)
	Stakeholders Who are related to this decision	**1.Service Developer:** Service developer will feel the pressure to develop better option i.e. the more the services are selected the better options are available. **3.Service broker and Service Requester:** Monitors the services (already integrated to the system) against the predefined SLA and QoS and initiates the process.
	Status	Approved
	Rationale	**Pros:** • The combination of both static and dynamic service discovery will results in a reliable system. Because, the services that are selected at the design time and fail to fulfill the required functionalities or violate any requirement can be replaced by alternatives through the run-time service discovery On the other hand the service discovery at design time will increase the run-time performance of the system because, the service consumer has no need to communicate with service registry at run-time.
	Relationship (s)	

Table 5.4: Concern no. 1 from SENSORIA Project.

Concern	The system should ensure that the service monitoring is performed by both service broker and the service requester.	
Ranking Criteria	Complexity, Effectiveness, Trustworthiness, Performance.	
Architectural Decision Alternative	**Identifier**	**2. a.** Service monitoring is performed by the service broker and the service requester.
	Description	Both of the service requester and service broker monitors the services that are integrated to the system. This monitoring is done in order to check whether these services are behaving according to SLA and delivering the said quality attributes. Service broker monitors the compliance of service behavior with SLA and QoS whereas service requester is more interested in service execution
	Executor of the decision	Service broker, Service requester
	Stakeholders Who are related to this decision	**Service provider:** the provider has to fulfill all the conditions mentioned in SLA.
	Status	Approved
	Rationale	**Pros:** • Both the service requester and service broker performs service monitoring according to their concerns. This will reduce the complexities involved in service monitoring and make it more efficient. **Cons:** • Separation of concerns involved in service monitoring will encourage the lack of trust. Conflict may occur between requester and broker to agree on a same point that a service is violating SLA.
	Relationship (s)	1.c
Architectural Decision Alternative	**Identifier**	**2. b.** Service monitoring is performed by a delegated trusted third party that both the service broker and service requester agree with.

	Description	Both requester and service broker agreed to appoint a trusted third party for service monitoring task. Which will reports monitoring data according to their concerns on regular basis.
	Executor of the decision	A delegated trusted third party.
	Stakeholders Who are related to this decision	Service broker, service requester
	Status	Discarded
	Rationale	**Pros:** • Both the service broker and service requester are freed from the task of service monitoring. The trusted third party will do the entire service-monitoring task and report generated by it will serve the purpose. **Cons:** • Both the service broker and the service requester have to agree to hand over the monitoring task to the trusted third party. • Extra efforts are required to select the trusted third party. • Extra communication is required with the third party. • The third party has to produce report on the monitoring data on regular basis and broker and requester have to response on it. • The involvement of the third party will introduces the complexities such as, failure in service monitoring and loss of communication with the third party will affect the success of the whole system
	Relationship (s)	

Table 5.5: Concern no. 2 from SENSORIA Project.

Concern	Generation of modes to retrieve alternate services that can replace the services that are already integrated to the system and satisfy the violated requirements	
Ranking Criteria	Flexibility, accessibility, availability, performance	
Architectural Decision Alternative	**Identifier**	**3.a.** Run-time service discovery
	Description	Services are searched or discovered dynamically at run time by Search engine.
	Executor of the decision	System Designer
	Stakeholders Who are related to this decision	**Service Requester:** Perform service monitoring against the predefined QoS
	Status	Approved
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software.
	Relationship (s)	2.a

Table 5.6: Concern no. 3 from SENSORIA Project.

5.4. Results without any extension in core model

By mapping the core model (without any extention) on the identified concerns and related process decisions from the two conducted case studies, I come to know that the resulted architectural knowledge is *incomplete*, because, it does not provides any information about the stakeholders involved in the architecting proces i.e. who will take the decisions and who will be affected by the taken decision, this can be seen in the following tables that shows the results without the two extended entities in the core knowledge model for architecture knowledge.

SeCSE Project [43]

Concern	Extraction of service discovery queries (that specify the functionality and quality properties) from architecture and design models of service centric system.	
Ranking Criteria	Flexibility, reliability, recoverability, performance	
Architectural Decision Alternative	**Identifier**	**A-1.** Architecture- time service discovery (ASD)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints.
	Status	Approved
	Rationale	**Pros:** • The new and modified version of design models enable the designer to specify more queries to discover other services that can satisfy more detailed functionality of the system. • The result of this process is a complete specification of the SCS architecture models.
	Relationship (s)	

Table 5.7: Concern no. 1 from SeCSE Project *(without extension in core model for AKM).*

Concern	Integrating the discovered services into an iterative design process of service centric system.	
Ranking Criteria	Flexibility, Reliability, Recoverability, Performance	
Architectural Decision Alternative	**Identifier**	A-2.Architecture- time service discovery (ASD)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints.
	Status	Approved
	Rationale	**Pros:** • The new and modified version of design models enable the designer to specify more queries to discover other services that can satisfy more detailed functionality of the system. • The result of this process is a complete specification of the SCS architecture models.
	Relationship (s)	

Table 5.8: Concern no. 2 from SeCSE Project *(without extension in core model for AKM).*

Concern	Generation of queries to retrieve alternative service that can satisfy the violated requirements.	
Ranking Criteria	Flexibility, accessibility, availability, performance	
Architectural Decision Alternative	Identifier	A-3.Run - time service discovery (RSD)
	Description	Alternate services are discovered at execution time to replace the services, which are already integrated to the SCS. This is done when already integrated services are failed to provide the required functionality and quality requirements.
	Status	Approved
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software.
	Relationship (s)	#2.a in the Base paper. (Service monitoring is performed by the service provider and the service consumer.)

Table 5.9: Concern no.3 from SeCSE Project *(without extension in core model for AKM).*

SENSORIA Project [44]

Concern	The system should be able to support different types of service discovery	
Ranking Criteria	Flexibility, accessibility, availability, performance	
Architectural Decision Alternative	**Identifier**	1.a. Static service discovery (architecture-time service discovery)
	Description	Identification of services that are specified by the design of the service centric system (SCS). These services provide required functionality and satisfy the desired quality properties and constraints. The location for such services are hard coded in the software
	Status	Discarded
	Rationale	**Pros:** • Discovery of service through local repository has better run-time performance because service consumers don't have to communicate with service registry at run-time. **Cons:** • Service consumers are unaware about the services that have similar functionality and better qualities in the service registry.
	Relationship (s)	
Architectural Decision Alternative	**Identifier**	1. b. Dynamic service discovery (Runtime service discovery).
	Description	Services are searched or discovered dynamically at run time by Search engine.
	Status	Discarded
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate

		services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software. **Cons:** • Run-time performance of the system is affected by the communication between service consumer and service registry.
	Relationship (s)	
Architectural Decision Alternative	**Identifier**	**I.c.** Combination of both static and dynamic service discovery
	Description	Adopts the combination of both static and dynamic service discovery. Where static service discovery is used at design time and dynamic service is used at run time in case of error occurrence.
	Status	Approved
	Rationale	**Pros:** • The combination of both static and dynamic service discovery will results in a reliable system. Because, the services that are selected at the design time and fail to fulfill the required functionalities or violate any requirement can be replaced by alternatives through the run-time service discovery On the other hand the service discovery at design time will increase the run-time performance of the system because, the service consumer has no need to communicate with service registry at run-time.
	Relationship (s)	

Table 5.10: Concern no. 1 from SENSORIA Project *(without extension in core model for AKM)*.

Concern	The system should ensure that the service monitoring is performed by both service broker and the service requester.	
Ranking Criteria	Complexity, Effectiveness, Trustworthiness, Performance.	
Architectural Decision Alternative	**Identifier**	2. a. Service monitoring is performed by the service broker and the service requester.
	Description	Both of the service requester and service broker monitors the services that are integrated to the system. This monitoring is done in order to check whether these services are behaving according to SLA and delivering the said quality attributes. Service broker monitors the compliance of service behavior with SLA and QoS whereas service requester is more interested in service execution
	Status	Approved
	Rationale	**Pros:** • Both the service requester and service broker performs service monitoring according to their concerns. This will reduce the complexities involved in service monitoring and make it more efficient. **Cons:** • Separation of concerns involved in service monitoring will encourage the lack of trust. Conflict may occur between requester and broker to agree on a same point that a service is violating SLA.
	Relationship (s)	1.c
Architectural Decision Alternative	**Identifier**	2. b. Service monitoring is performed by a delegated trusted third party that both the service broker and service requester agree with.
	Description	Both requester and service broker agreed to appoint

		a trusted third party for service monitoring task. Which will reports monitoring data according to their concerns on regular basis.
	Status	Discarded
	Rationale	**Pros:** • Both the service broker and service requester are freed from the task of service monitoring. The trusted third party will do the entire service-monitoring task and report generated by it will serve the purpose. **Cons:** • Both the service broker and the service requester have to agree to hand over the monitoring task to the trusted third party. • Extra efforts are required to select the trusted third party. • Extra communication is required with the third party. • The third party has to produce report on the monitoring data on regular basis and broker and requester have to response on it. • The involvement of the third party will introduces the complexities such as, failure in service monitoring and loss of communication with the third party will affect the success of the whole system
	Relationship (s)	

Table 5.11: Concern no. 2 from SENSORIA Project ***(without extension in core model for AKM)***

Concern	Generation of modes to retrieve alternate services that can replace the services that are already integrated to the system and satisfy the violated requirements	
Ranking Criteria	Flexibility, accessibility, availability, performance	
Architectural Decision Alternative	**Identifier**	**3.a. Run-time service discovery**
	Description	Services are searched or discovered dynamically at run time by Search engine.
	Status	Approved
	Rationale	**Pros:** • Alternate services with similar functionalities and better qualities can be identified at execution time. • Through run-time service discovery alternate services with similar functionality but better qualities can easily replace the existed (or already integrated) services, because their location is not hard coded in the software.
	Relationship (s)	2.a

Table 5.12: Concern no. 3 from SENSORIA Project *(without extension in core model for AKM).*

5.4.1 Achievements from extended model

The information about these two missing entities in the core knowledge model is important because of the fowlling reasons:

1. Fig5.1 shows that the architecting process is a stakeholder driven process, the roles and responsibilities of these involved stakeholders are very important and should be clearly known [26], [27].

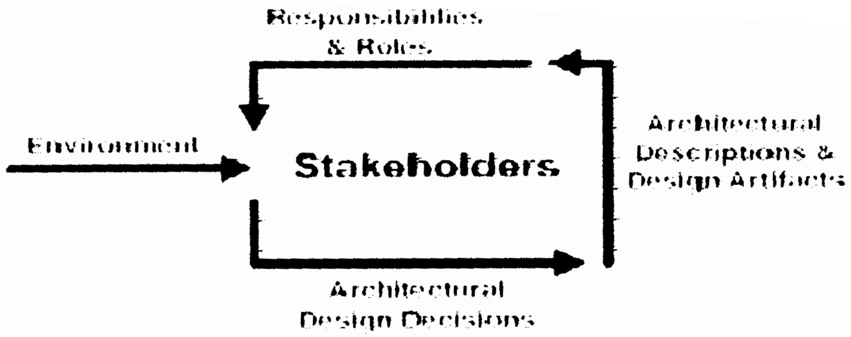
Responsibilities & Roles
Environment
Stakeholders
Architectural Descriptions & Design Artifacts
Architectural Design Decisions

Fig 5.1: Schematic outline of architecting process

(Ref taken from " Structuring software architecture project memories [26]"& " Tailoring knowledge sharing to the architecting process [27] ")

2. The new characteristics of SOA imposed process decision in service oriented systems that results in increased number of stakeholders in SOA as compared to traditional S/W architecture. These stakeholders perform different roles and responsibilities, such as, service engineer is responsible for developing the services which are then hosted and published by the service provider. Service broker and integrator are responsible to register and integrate them and finally service consumer invokes these services [9].

   Service oriented systems or environment required from these stakeholders to cooperate in such a way that the resulted application fulfills the requirements of the end user. And to obtain the above-mentioned result we should know their roles and responsibilities i.e. who will take or execute the decisions and who will be impacted by the taken decision.

3. The process decision and their results affect not only the architecting process but also the development process and the involved stakeholders. So, we should know their roles and responsibilities i.e. who will take or execute the decisions and who will be impacted by the taken decision.

5.5. Findings

On the basis of results obtained from the chapter 5 we conclude following:

- The extension in the core model for AKM is necessary. It can be seen that when the identified process decisions from the two case studies are documented through core knowledge model, the resulted knowledge appears as an incomplete knowledge. It does not provide any information regarding the involved stakeholders (i.e. who are making the decisions and who are going to be affected by the taken decisions) which are important factor in the case of process decisions in SOA, and.

- The results also validate that "Extended core model (ECM)" for AKM is the model that can document process decisions in SOA domain.

6. DISCUSSION AND CONCLUSION

The architectural knowledge management appears as an important factor for the success of any IT organization. Many tools such as PAKME [28], HyperOnto [32] and ADDSS [30], methods such as CBSP [31], approaches presented in [29], [32], [33], [37], [34] and [35] and frameworks e.g., DISCO [36] had been developed to deal with the problems in the management of architectural knowledge, but all these have their own pros and cons. Different knowledge models had also been proposed by different researchers for AKM and all these knowledge models works well in traditional software architecture but show their inability in services oriented architecture. The management of architectural knowledge is equally important in both traditional as well as in service-oriented architecture. The prevailing advantages of SOA compel the software industry to move its focus of attention towards service orientation. Architectural knowledge is composed of different design decisions and their rationale and these design decisions also include process decision as one of their type. Process decisions are the design decisions that are related with the development process and affect the involved people and organizational environment. Philippe Kruchen [20] presents these process decisions as sub type of executive design decisions. Process decisions play important role in the service-oriented architecture, since it is a new paradigm and researchers are still working on the development process for service-oriented systems. The existing knowledge models are not capable enough to document process decisions in SOA domain.

Qing Gu and Patricia Lago in their work [9] suggest an extension in the core model for architectural knowledge management, and name it as "Extended core model (ECM)" for AKM. They claim that the core model after extension is capable enough to document the process decisions in SOA domain. To justify their claim they prepare a conceptual model as shown in table2.1 that contain the concepts related to document the process decision and collect two sample process decisions form an international project "SeCSE", then they map the concepts that are used in these process decisions onto the core model. According to the results from this mapping they were able to locate most of the concepts in the core model except the two concepts i.e., the executor of the decision and related stakeholders. So they extend the core knowledge model.

Since the research work presented in this thesis is validating the above mentioned claim, so some concerns and their related process decisions are collected from two different international case studies i.e., SeCSE and SENSORIA as shown in chapter5. The concepts that are used to document these process decisions are then mapped onto the extended core model with the help of table2.1 and the results shows that the claim of Patricia Lago and Qing Gu is right and extended core model for AKM is able to document the process decisions in SOA domain.

6.1. Contribution of the research

The major contribution of this thesis include: Firstly, it validate that the Extended core model (ECM) for AKM is the model that can be used to document the process decisions in SOA, which will benefit software architects while documenting architecture knowledge. Secondly, it proves that existing knowledge models are not able to document process decisions in SOA, this is done by mapping table2.1 (conceptual model that contains the concepts used to document process decisions) onto theses existing knowledge model. This is done by extensive literature survey of different knowledge models and techniques used for the AKM. Thirdly, it helps to prove the claims of the base paper that extension in the core knowledge model is necessary that makes the Extended core model (ECM) as a model to document the process decision in SOA.

6.2. Future Work

The possible directions for the future research are, first of all use the same core knowledge model with extension in other domains of software architecture such as MDA etc, to check its applicability for documenting process decision. Secondly, since two case studies used in this thesis were already conducted and their data are used as secondary data, so one of the future directions is to conduct industrial case study to get clearer picture of the SOA domain.

REFERENCE

1. Patricia Lago, Paris Avgeriou, Rafael Capilla, Philippe Kruchten: Wishes and Boundaries for a Software Architecture Knowledge Community, In: 7th Working IEEE/IFIP Conference on Software Architecture, 2008.
2. Patricia Lago, Philippe Kruchten, et al.: Sharing and Reusing Architectural Knowledge- Architecture, Rationale, and Design Intent. In: 29th International Conference on Software Engineering (ICSE'07 Companion)
3. Philippe Kruchten, Patricia Lago, and Hans van Vliet: Building up and Reasoning about Architectural Knowledge, In: 2nd International Conference on the Quality of Software Architectures (QoSA 2006), June 2006.
4. Amelia Maurizio, James Sager, Peter Jones, Gail Corbitt and Lou Girolami: Service Oriented Architecture: Challenges for Business and Academia, In: 41st Hawaii International Conference on System Sciences – 2008.
5. T.G.J. Schepers, M.E. Iacob,P.A.T. Van Eck: A lifecycle approach to SOA governance, *SAC'08*, March 16-20, 2008, Fortaleza, Ceará, Brazil.
6. Michael Menzel1, Ivonne Thomas1, Christian Wolter2, Christoph Meinel1: SOA Security -Secure Cross-Organizational Service Composition, In Proc. Stuttgrater Softwaretechnik Fourm(SSF), Fraunhofer IRB-Verlag, Stuttgart, Germany, November 2007. pp.41-53, ISBN 978-3-8167-7493-8.
7. Qing Gu, Patricia Lago: A stakeholder-driven Service Life Cycle Model for SOA, IW-SOSWE'07, September 3, 2007, Dubrovnik, Croatia.
8. Amelia Maurizio, James Sager, Peter Jones, Gail Corbitt and Lou Girolami: Service Oriented Architecture: Challenges for Business and Academia, from SAP AG and Sager and Corbitt from California State University, Chico. Proceedings of the 41st Hawaii International Conference on System Sciences – 2008.
9. Qing Gu, Patricia Lago: SOA Process Decisions: New Challenges in Architectural Knowledge Modeling, SHARK'08.
10. Qing Gu, Hans Van Vliet: SOA decision making- what do we need to know SHARK'09.

11. Rainer Berbner, Tobias Grollius, Nicolas Repp, et al: An approach for the Management of Service-oriented Architecture (SoA) based Application Systems, EMISA 2005.
12. M. Begon˜a Lloria: A review of the main approaches to knowledge management, Knowledge Management Research & Practice (2008) 6, 77–89.
13. Muhammad Ali Babar, Ian Gorton, and Ross Jeffery: Capturing and Using Software Architecture Knowledge for Architecture-Based, Proceedings of the Fifth International Conference on Quality Software (QSIC'05), IEEE 2005.
14. Anton Jansen: Architectural design decisions, Press Grafische Industrie de Marne, Leens, The Netherlands, 2008, ISBN: 978-90-367-3494-3
15. Michal Przemyslaw Rudzki, Fredrik Jonson: Identifying and Analyzing Knowledge Management Aspects of Practices in Open Source Software Development, School of Engineering Blekinge Institute of Technology, August 2004.
16. Muhammad Ali Babar, Ian Gorton: Architecture Knowledge Management: Challenges, Approaches, and Tools, 29th International Conference on Software Engineering (ICSE'07 Companion)
17. Rik Farenhorst, Patricia Lago, and Hans van Vliet: Effective Tool Support for Architectural Knowledge Sharing, F. Oquendo (Ed.):ECSA 2007, LNCS 4758, pp. 123–138, 2007. c_Springer-Verlag Berlin Heidelberg 2007.
18. Muhammad Ali Babar, Remco C. de Boer: Torgeir Dingsøyr, Rik Farenhorst: Architectural Knowledge Management Strategies: Approaches in Research and Industry, Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent (SHARK-ADI'07).
19. Paris Avgeriou, Patricia Lago,Paul Grisham and Dewayne Perry: Architectural Knowledge and Rationale – Issues, Trends, Challenges, 2007 SHARK .
20. Philippe Kruchten: An Ontology of Architecture Design Decisions in Software – Intensive System, 2nd Groningen Workshop on Software Variability Management, Groningen, The Netherlands (2004).
21. T.G.J. Schepers , M.E. Iacob, P.A.T. Van Eck: A lifecycle approach to SOA governance , *SAC'08,*

22. Hans van Vliet: Software Architecture Knowledge Management, 19th Australian Conference on Software Engineering, 2008 IEEE.
23. Remco C. de Boer, Rik Farenhorst, Patricia Lago, Hans van Vliet: Architectural Knowledge: Getting to the Core, Springer-Verlag Berlin Heidelberg 2007
24. Jeff Tyree and Art Akerman: Architecture decisins: Demstifying architecture, I E E E 2 0 0 5.
25. O. Zimmermann, J. Koehler, and F. Leymann: The role of architectural decisions in model-driven SOA construction. In the 21st Int. Conf. on Object-Oriented Programming, Systems, Languages, and Applications,Best Practices and Methodologies in Service-Oriented Architectures, Portland, 2006. ACM.
26. R. C. de Boer, R. Farenhorst, J. S. v. d. Ven, V. Clerc, R. Deckers, P. Lago, and H. v. Vliet: Structuring software architecture project memories. In the 8th International Workshop on Learning Software
27. R. Farenhorst: Tailoring knowledge sharing to the architecting process. ACM SIGSOFT Software Engineering Notes, 31(5):3, 2006.
28. Muhammad Ali Babar, Ian Gorton: A tool for managing software architecture knowledge (SHARK-ADI'07).
29. Lenin Babu T, Seetha Ramaiah M et al: ArchVoc – Towards an Ontology for Software Architecture, (SHARK-ADI'07).
30. Rafael Capilla, Francisco Nava et al: Modeling and Documenting the Evolution of Architectural Design Decisions, (SHARK-ADI'07).
31. Charles L. Chen, Danhua Shao, Dewayne E. Perry: An Exploratory Case Study Using CBSP and Archium, (SHARK-ADI'07).
32. Ibrahim Habli Tim Kelly: Capturing and Replaying Architectural Knowledge through Derivational Analogy, (SHARK-ADI'07).
33. Davide Falessi, Rafael Capilla,et al: A Value-Based Approach for Documenting Design Decisions Rationale: A Replicated Experiment, SHARK'08.
34. Muhammad Ali Babar: The Application of Knowledge-Sharing Workspace Paradigm for Software Architecture Processes, SHARK'08.
35. Rik Farenhorst, Ronald Izaks, Patricia Lago, Hans van Vliet: A Just-In-Time Architectural Knowledge Sharing Portal, WICSA.2008

36. Chris A. Mattmann, David Woollard, Nenad Medvidovic,: Exploiting Connector Knowledge To Efficiently Disseminate Highly Voluminous Data Sets, SHARK'08.

37. Davide Falessi, Giovanni Cantone, Philippe Kruchten: Value-Based Design Decision Rationale Documentation:Principles and Empirical Feasibility Study, WICSA.2008

38. A.Jansen and J.Bosch, Software Architecture as a Set of Architectural Design Decisions, Proceedings of 5th working IEEE/FIP Conference on Software Architecture (WICSA), pp.109-120,2005.

39. Harrison,N.B., Avgeriou.P., and Zdun, U., Using Pattrerns to Capture Architectural Decisions, IEEE software, 24 (4): 38-45, 2007.

40. Capilla, R., Nava, F., and Duenas, J.C., Modeling and Documenting the Evolution of Architectural Design Decisions, Proceedings of the 2nd Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent (SHARK/ADI), 2007.

41. Tang, A., Han, J., and Vasa, R., Software Architecture Design Reasoning: A Case for Improved Methodology Support, IEEE Software , 26 (2): 43-49, 2009.

42. Service Oriented Architecture. http://en.wikipedia.org/wiki/Service-oriented_architecture.

43. SeCSE. Secse website http://secse.eng.it/, 2007.

44. SENSORIA. Sensoria website http://www.sensoriapist.eu.

45. Knowledge definition. http://en.wikipedia.org/wiki/Knowledge.