UNIVERSITY OF SOUTHAMPTON

Supporting the Engineering Design Process with Semantic Web Technology

by

DI SHI

A thesis submitted for degree of Master of Philosophy

in the
Faculty of Engineering Science and Mathematics
School of Electronics and Computer Science

March 2009

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Master of Philosophy

by DI SHI

In engineering, knowledge is generated when designing a product, but very limited useful information is normally stored during and after the design process. Although information is recorded in design document, emails, and meeting minutes, this still does not provide effective support for future engineering designers. It is apparent that even with the considerable amount of research, the design community has failed to make full use of design rationale capture in real-world projects. This is not because there is reason to doubt the potential usefulness of capturing and delivering design rationale, but instead because of excessive time, labor, cost, and project disruption that is required to make both capture and delivery of design rationale effective. In this thesis the author first introduces a flexible ontology-based schema based on the evolutional decision making model with formally defined semantics that enables the capture and reuse of design knowledge, supported by advanced knowledge and intelligent systems. As part of the work, a service-oriented, loosely coupled engineering design system framework is developed. The major issue is the representation and capture of design rationale for design management and for use in a re-design or related design. In this thesis, we describe an approach to a rationale system, *Semantic Web based Design Rationale System* (SW-DRS) which consists of three parts: (i) an ontology based argumentation representation, (ii) a knowledge based design rationale database, (iii) a design artifact model. In addition, it is demonstrated that the design rationale within a real world project can be captured for evaluating with the research objectives. Finally, we evaluate SW-DRS with a similar design rationale system, DRed, and conclude that the approach taken by SW-DRS will be beneficial to design engineering.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

During the engineering design process, a significant amount of new knowledge is generated. Designers normally talk about the artifact, reinforce ideas, support or disagree with each other, or reach conclusion by design document. Technical documents are created, modified and used during the life cycle of an artifact. They can be more or less formal, ranging from normative knowledge-based representations to natural language. The knowledge can be valuable, even critical, to people (including engineering personnel) who design, maintain, and optimize the artifact. Successful capture of design rationale is considered to be very important to the success of any engineering project. It is estimated that in a design activity up to 70% of the information used is taken from previous solutions, (Khadilkar and Stauffer, 1996). In addition engineering companies are becoming concerned with the effective use of design knowledge accumulated over previous design experiences, (Pahng et al., 1999). The design process often is in part implicit, for example, information hidden in the minutes of meetings, design notebooks, email archives, or as the designers' personal experience. This makes it difficult to retrieve past knowledge, and hence makes its reuse even harder.

1.1 Motivation

The recognition of the need to share knowledge in the engineering organizations has led to developments of systems that manage knowledge in order to learn from past mistakes, avoid re-invention when a similar problem has already been solved, and support the latter design phase where questions about early design decisions which often can not be answered (Shum, 1996).

Possibly the earliest methodology which current argumentation research[1] continues to

[1]Argumentation research is defined as the study of the arts and sciences of dialogue, and conversation, using rules of inference, logic, and procedural rules in real world settings. Argumentation is concerned primarily with reaching conclusions through logical reasoning based on certain premises.

pursue was set out by Englebart (1963). He suggested the structure that map into a human's mental structure will significantly improve the designers' capability to comprehend and find solutions within a complex problem solving situation. A large amount of work has contributed to improving computer supported notational structures - a way to represent design arguments explicitly (Kunz and Rittel, 1970; Conklin and Begeman, 1988; Fischer et al., 1989a,b; Shipman and McCall, 1997; MacLean et al., 1991; Lee, 1990). In practice none of these systems had been proved successful in real problem solving, for reasons which the author will discuss in Chapter 4.

In order to improve the productivity and efficiency of design activities by utilising and sharing design knowledge, a systematic approach for managing design knowledge is needed. This work will justify whether the computer-assisted management tools for design knowledge using Semantic Web technology provide a more effective way for construction design information over existing solutions. The design knowledge is captured by a framework based on an argumentation ontology, which is based on an improved argumentation model, which makes information more accurate hence more useful. Ontology has been proved useful in managing information about product family (design artifact) (Nanda et al., 2004). If semantic web technology can be used in building design rationale, it will make the design reason more explicit as the knowledge can be structured by ontologies. This is not only for the human, but also for the computer, and hence provides more computational services, which may be used for better design, design evolution and redesign.

1.2 Definition of design rationale

To understand why an artifact design the way it is, a clear understanding is needed about how it could be different, and why the choices made are appropriate. That results in an explicit representation which describes a design space rather than a specific artifact. This research uses the term "design rationale" to refer to this representation. Fundamentally, design rationale is a methodology for problem solving and decision making in design context. A review of the literature revealed the following definitions:

- Design rationale means statements of reasoning underlying the design process that explain, derive, and justify design decisions (Fischer et al., 1996).
- Design rationale is a representation for explicitly documenting the reasoning and argumentation that make sense of specific artifact (MacLean et al., 1991).
- Design rationale is the information that explains why an artifact is structured the way that it is and why it has the behaviour it has (Conklin and Burgess-Yakemovic, 1991).

- Design rationale is an explanation of why something is designed as it is (Grüber and Russell, 1992).
- Design rationale includes the design problems, alternative resolutions (including those which are later rejected), trade-off analysed among these alternatives and a record of tentative and firm commitments that were made as the problem was discussed and resolved (Blessing, 1994).

Some researchers also argue that design rationale comprises information not only about decision making, issues, alternative solution (including those which are rejected), trade-off analysis among these alternatives, criteria etc, but also about design artifact data including requirement, function, structure, etc. (Lee, 1997; Werner and Ahmed, 1999; Brissaud et al., 2003; Nomaguchi et al., 2004).

1.3 Importance of the problem

Research into design rationale capture was first reported by Kunz and Rittel (1970), and Issue-Based Information System (IBIS) was first introduced. Most of that work emphasizes how design rationale should be represented, especially the notation used for representing the design domain vocabulary and their relationships. The challenge is not to intrude on the designers as they accomplish the design task, especially when time is a critical factor during a design task. One contribution of this work is the ability to systematically collect and share knowledge by making use of the connectivity provided by Internet technologies. For example, the Semantic Web (Berners-Lee et al., 2001), initiated by W3C, carries the promises to make the web machine-understandable by enriching available information with logic based semantics and provide us with new framework for knowledge interchange and sharing. The Semantic Web is a web of data, implying a common format for integration and combination of data drawn from diverse sources. Moreover, Semantic Web is about language for recording how the data relates to real world objects.

This work will not only answer the question how to organize and manage that knowledge, but also suggest the way how this knowledge enhances the design activities, such as how to discover the correct knowledge over enormous amount of knowledge available, how the right design knowledge is retrieved at the required part of the design phase. The use of web services in design rationale system will provide a modular approach that can be reconfigured as the design activity processes, allowing optimal information retrieval at all stages of the process.

How the design rationale is used depends on its representation format and content. Shipman and McCall (1997) claimed that the perspectives of design rationales are: argumentation, documentation and communication. Computer support during the design process can considerably reduce time and improve the design. The aim of this research is to provide a framework in which a design problem can be argued thoroughly and hence a better solution generated effectively. Furthermore, the design rationale, if well structured, can be converted into documentation, which benefits for users extending to the design process and can be used for project management communication.

The use of well designed design rationale system can thus improve design management, collaboration, design reuse, maintenance, learning, and document. Such a system can benefits a number of activities and user group: better design (designers), maintenance (system maintainers), education (new trainees, students, learning program), and documentation (future designer and maintainers) (Lee, 1997).

1.4 Research question and contribution

In this work the author has developed an approach to design rationale capture and dissemination that is based on web services and semantic web technologies. The existing systems either require a large amount of effort by designers to construct the rationale which causes the designers' resistance, or the capture of design rationale that is no accurate enough to benefit other users. It is expected that the proposed approach will minimize the non-productive effort required to create design rationale and maximize its utility for those designers creating design rationale, therefore the method will delivery measurable benefits in the quality and quantity of design rationale captured over existing systems.

The formal research question for this work can be expressed as: **what are the improvements that the argument structure developed in this work can give to design rationale capture.**

The primary contribution of this work derives from the way it approaches the problem of capturing engineering design rationale. Issue Based Information System was first published in 1970s and a number of derivatives followed in the following decades. Nomaguchi et al. (2004) introduces a new element which is called emphasis extending IBIS. Emphasis is defined as a node that represents importance of the argument. Ahmed and Wallace (2001) investigate design strategies need to be employed when possible solution infer to decision during evaluation. While design strategy is too implicit, the author introduces the concept of "criteria" to replace design strategy, relating solution, evaluation and decision which construct the decision making model that best represents the decision making process.

The second contribution of this research is the argumentation ontology developed by the author, which can be the basis of rationale capturing in the design rationale capture system framework. The modular framework which is based on a service-oriented architecture allows the integration of complex information domain. This highly flexible structure in turns enhances the reusability and interoperability of the system implemented with this framework.

1.5 Structure of thesis

Following this instruction, Chapter 2 presents an overview of knowledge management research. The definition of knowledge is reviewed, followed by a summary of recent development in knowledge-oriented technology which has led to significant benefits for organizations.

Chapter 3 introduces the typical engineering design process. Some important capture methodologies are reviewed and summarises the approaches to capture design rationale.

Chapter 4 details a service-oriented architecture for capturing design rationale. The services are loosely coupled, which means the service interface is independent of the implementation. The designers's work will be supported by the development of a system built on top of loosely connected ontology classes via a semantic web portal.

Chapter 5 introduces how semantic web help for capturing design rationale, including the ontologies the author build in the system. And also a vision about semantic web can help possibly reach a higher automation level is discussed, if fair amount of comprehension and complex decision-making, the right model knowledge is well prepared and highly accessible.

Chapter 6 presents the design of the proposed evaluation approach strategy, as well as defining the evaluation questions and goals. To illustrate the approach, the author carried out a robot design case study, which used the SW-DRS decision making model to capture the entire robot chassis design. Finally a brief comparison between SW-DRS and a similar rationale capture system DRed is made.

Chapter 7 summarises the importance and contribution of this research achieved to date and presents the future research directions.

1.6 Declaration

This work is based upon the work undertaken by the author within a collaborative research environment. This thesis is the original work of the author and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Chapter 2

Knowledge Management

Over the past ten years, organizations have taken considerable interest in how to enhance their performance, efficiency and reduce the resource waste. That rises the live issue in business world termed "knowledge management". As more companies begin knowledge management programs and move through the plan, review and implementation stages, demand for knowledge management products and services will increase, as seen in many new publications, conferences, IT products and company. The worldwide spending for knowledge management services including consulting, implementation, support, outsourcing and training grows from $776 million in 1998 to $2.3 billion in 2000, and reach $12.7 billion in 2005, according to a research firm International Data Corporation (IDC, 2006). However, despite the excitement surrounding knowledge management there is significant debate and confusion on what it is or what it should be. This chapter presents an overview of knowledge management.

2.1 What is knowledge, and knowledge management

Before going any further it is important to define what is meant by knowledge. However, the termed "knowledge" is hard to define precisely and simply, this is not surprising, for example how would a manager define "management" and how a scientist define "research". The Oxford English Dictionary's definition of knowledge is "awareness of familiarity gained by experience".

Within the academic literature there are many thoughtful and thought-provoking definition of knowledge, including the important distinctions made between data, information, knowledge and wisdom (Ackoff, 1989; Bellinger et al., 2002). As shown in Figure 2.1, data is the raw material of information. It is the product of research and discovery. A single piece of data has no meaning unless its context is understood. Data needs to be transformed to information. Information is a meaningful flow of data. The patterns and relationship in the data is pointed out and discussed leading to data being transforming into information. It is the data that has been given meaning by way of relational connection. This "meaning" can be useful, but does not have to be (Bellinger et al., 2002). Whereas, knowledge is the appropriate collection of information, as its intent has to be useful. There are numerous definitions of knowledge - to be found at conferences, e-prints and on the web. One of the widely cited definition is introduced by Davenport and Prusak (1998). Knowledge is a fluid mix of framed experience, values, contextual and actionable information, and expert insight the provides a framework for evaluating and incorporating new experiences and information. Knowledge indeed consist of two main aspects, one is the body of information, which contain the linked data. While the other as knowledge is created in a situation, and it is accumulated and integrated and held over time to handle specific situations, so can also be defined as actionable information. Wisdom calls upon all previous levels of consciousness, it give us understanding about which there has previously been no understanding. It is just a scenario that computers will take the role of human being thinking, the author will not go further in our discussion.

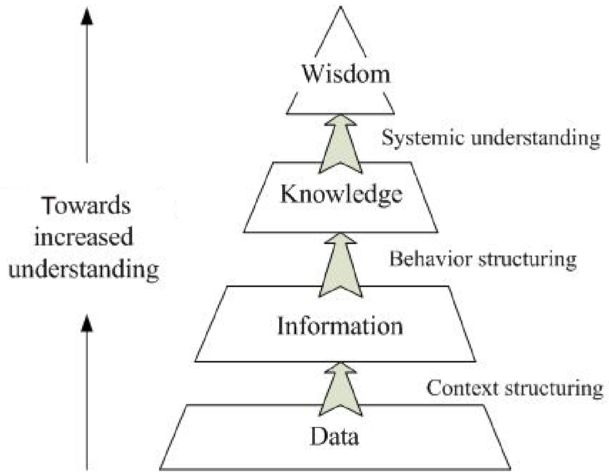


FIGURE 2.1: The route from data through to wisdom based on Ackoff (1989)

Various people have thought carefully about varying definitions and produced their own analysis their working problem, for example in the following quotation from Steyn (2004):

> *"Information consists of data, but data is not necessarily information. Also, wisdom is knowledge, which in turn is information, which in turn is data, but, for example, knowledge is not necessarily wisdom. So wisdom is subset of knowledge, which is a subset of information, which is a subset of data."*

The knowledge management domain is complex, with many subareas of specialization, so it is difficult to give a brief definition. Is it a organization strategy, latest information technology or business information processing? Base on the definition of knowledge,

knowledge management (KM) can be considered the discipline that provides strategy, process and technology to share and leverage information and expertise that will increase our level of understanding to more effectively solve problems and make decisions (Harigopal and Satyadas, 2001). Knowledge management is a concept which an enterprise gathers, organizes, shares and analyzes the knowledge of individuals and group across the organization in ways the directly affect performance. Knowledge management predicts retrieving the right information, in the right context, for the right person, at the right time, for the right business purpose.

Knowledge management draws from a wide range of disciplines and technologies.

- **Document management**. It moved on to making content accessible and reusable, making the document be meaningful by defining metadata and associate it with document objects.
- **Data warehouse and rationale database**. They shift the concept of managing "structured" data to the methodology for storing "unstructured" content, with the models for representing and managing knowledge resources.
- **Semantic network**. Semantic networks are formed of ideas commonly termed "hypertext without the content". The traditional World Wide Web lacks the capability to locate relevant information rapidly. In 2001, Tim Berners-Lee, James Hendler and Ora Lassila wrote a seminal article in which they envisioned the Semantic Web (Berners-Lee et al., 2001). Their scenario demonstrating how the Semantic Web might be used, through identifying and using services on the Web.
- **Expert System, Artificial Intelligence**. Knowledge-based systems use specialized sets of coded knowledge to reason and perform intelligent tasks. This is in contrast with the conventional type systems which rely on data and general algorithms to solve less intelligent tasks. Knowledge-based systems have proved to be much more successful than previous problem solving systems from practices.
- **Decision support system**. The research combines the field from cognitive science, management science, business processing to system engineering in order to produce both effective and qualitative artifacts, and integrate such artifacts within the decision making processes of designers. While similar to a Knowledge Management System, the emphasis is on specific product design rather than organization knowledge sharing and knowledge flow.
- **Cognitive science and organizational science**. Acquisition from how to learn and know will certainly improve tools and techniques for gather and transferring knowledge. The social perspective such as organizational behaviour and culture are also paramount.

In the following section, the author seeks to better understand how knowledge is being managed and processed.

2.2 Knowledge life cycle

The process of transforming data and information into knowledge is a continuous cycle. Knowledge management is a cyclic process involving three related activities: creation, integration, and dissemination (Fischer and Oswald, 2001). Whereas, Sim (2004) propose knowledge management can be more than that, Figure 2.2 depicts the knowledge management cycle as consisting of four fundamental steps that involves creation, storage, integration, and dissemination. However, there are various versions of knowledge life cy-

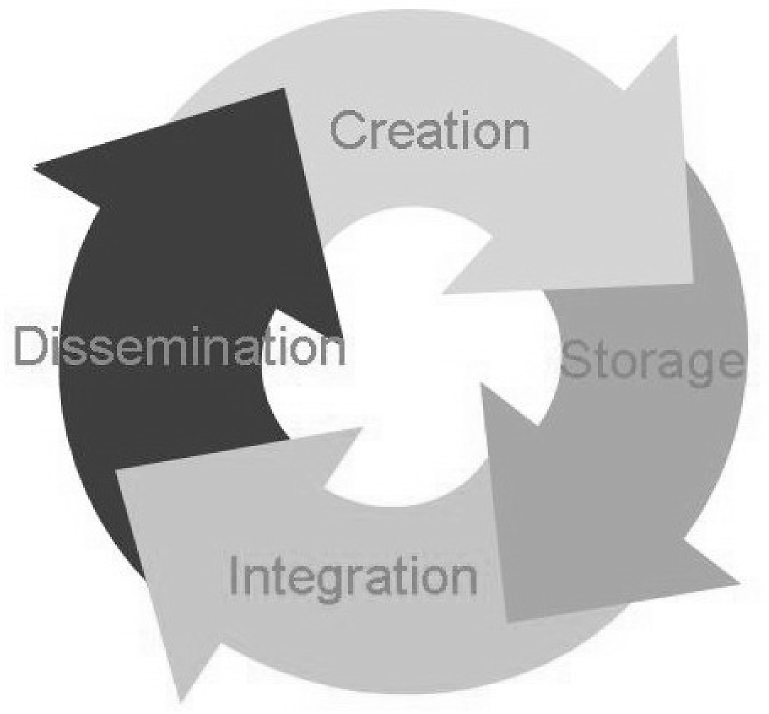


FIGURE 2.2: A knowledge cycle shows the activities defined in section 2.2 (Sim, 2004)

cle, Millard et al. (2006) presented four stages cycle: knowledge acquisition, knowledge modelling, knowledge annotation and knowledge reuse. Wong et al. (2006) identified four key activities in knowledge management - knowledge creation, knowledge mapping, knowledge retrieval and knowledge use. Recently, a comparative complete and specific knowledge life cycle was defined by AKT (2003). This knowledge management cycle includes six activities: acquiring knowledge, modelling knowledge, reusing knowledge, retrieving knowledge, publishing knowledge, and maintaining knowledge, which the author will discuss in Section 2.4.2.

2.2.1 Creation

Based on the resource that new knowledge is created, the creation process contains two perspectives. One is the conversion from data into information, then into knowledge, this has been discussed in last section. For example, system-assisted knowledge discovery processes find new patterns in existing information during data or text mining (Kosala and Blockeel, 2000). While the other perspective attracts more interest in recent years. According to Fischer and Oswald (2001), traditional knowledge management approaches assume that the critical issue for workers is to find the "answers" in organizational memory that apply to the current problem. A design-based approach assumes that the organizational memory will not contain all the knowledge required to understand and solve such problems. So, workers must create new knowledge. Furthermore, knowledge is not static as information, it works only when collaborate with its specific situation. Knowledge to someone is only information to others until it is validated.

2.2.2 Storage

Within this version of the knowledge cycle the storage activity also incorporate the capture and elicitation of the knowledge. While in this stage, the work is the formalizing of the knowledge, usually this requires the use of an ontology. Explicit knowledge can be captured as it is created, but the implicit knowledge capturing is considered to be a research challenge. Actually, the key issue is the approach for representing knowledge (Fischer and Oswald, 2001). Implicit knowledge usually can be elicited from the sources using questionnaires, interviews, or leveraging from a collaborative environment. Knowledge thus gathered can be represented using schema such as semantic web, scripts, expert systems, design rationale system, etc. On the other hand, tacit knowledge can not be captured precisely at this stage of knowledge life cycle but only can be identified after integrate with other organization sources, including personal and group social factors. This process require the ability to map from domain concepts into formalizations the system requires.

2.2.3 Integration

Former organisational memory is not only the source of information to help users understand the problem they face, but also a origination for new information and products created during work. Although the problems users solved are unique in some aspects, they are also similar to those previously solved. The process is a continuous activity by which an organisation introduces new knowledge claims to its operating environment and work with others or reties old ones. They are actively integrated into the work process and social practices of the community that construct them. Knowledge integration includes all knowledge transmission, teaching, and other social activity that communicates either an understanding of previously produces organizational knowledge to knowledge workers, or the organizational knowledge base.

Fischer and Oswald (2001) summarises knowledge integration comprises two tasks: conceptual generalization during which relating information from on context to information from another; representational formalization by which putting information in a form such that computational mechanisms can access and interpret it. While in our perspective, the second task is the main objective of storage in the knowledge life cycle.

2.2.4 Dissemination

Knowledge created, stored, and integrated is ready for distribution via multiple objectives. Dissemination includes "pushing" the knowledge to users and users retrieving, sharing and reusing the knowledge when needed. The range of push mechanisms includes information and knowledge portals, intelligent agents, and recommendation systems (Suresh and Egbu, 2004). An agent application probably suggest how to use knowledge after capturing and storage (Maes, 1991). Applications of intelligent agents are required in all stages of managing knowledge, especially during automatic knowledge organization and retrieval. Such agents will interact with knowledge worker to understand user interest or expertise and present relevant information as well as interact with other knowledge agents to assemble knowledge base. Given the social capital implications of leveraging knowledge management, and the related aspects of culture and change, robust agent models can help us significantly in understanding and providing immediate value.

As the author indicates, knowledge management is a cyclic and continuous process during ongoing business activities. And the distinctness between these four steps is difficult to identified. Furthermore, it seems lack an effective and general approach for serving the knowledge management cycle. The next section will review approaches to knowledge management approach and analysis why some applications failed.

2.3 Approaches to knowledge management

There are many ways to slice up the multi-faceted world of knowledge management. However, it is often useful to categorize them. Sveiby (1996) identified two "tracks" within knowledge management:

- **Management of Information**. Researchers and practitioners in this field tend to have their education in computer and information science, telecommunications, database and data analysis. They are involved in construction of information management systems, AI, reengineering, group ware etc. To them knowledge equals objects that can be encoded, stored, transmitted and processed by IT systems.

- **Management of People**. This is organizational theory with backgrounds in philosophy, psychology, sociology or business management. Proponents of this theory believe that knowledge equals process, a complex set of dynamic skills, know-how etc, that is constantly changing.

Sveiby's characterization is considered acceptable, but it may also been classified in the flavor of diverse objectives of knowledge management: mechanistic approaches and behavioristic approaches.

Mechanistic approaches to knowledge management are characterized by the application of technology and better accessible and structured resources. The main assumptions of the mechanistic approach include:

- Better accessibility to information is a key issue, including enhanced methods of access and reuse of documents (hypertext linking, databases, full-text search, etc.)
- Networking technology in general (especially intranets), and groupware in particular, will be key solutions.
- In general, technology and sheer volume of information will enhance the knowledge management process. This approach places the emphasis on the technology to provide a solution.

Behavioristic approaches, with substantial roots in process re-engineering and change management, tend to view the knowledge problem as a management issue. Technology though ultimately essential for managing explicit knowledge resources is not the solution. These approaches tend to focus more on innovation and creativity (the learning organization and recommend solution) than on leveraging existing explicit resources or making working knowledge explicit.

Assumptions of the behavioristic approaches include the following:

- Organizational behaviors and culture need to change dramatically. In the current information-intensive environments, organizations typically become dysfunctional relative to their business objectives.
- Organizational behaviors and culture can be changed, but traditional technology and methods of attempting to solve the knowledge problem have reached their limits of effectiveness, therefore a more holistic view is required. Theories of behavior of large-scale systems are often invoked to resolve these challenges.

2.4 Knowledge technologies

Knowledge is hidden in all existing sources including personal rationale, systems, databases, meeting notes, email archive and organization culture. It enhances the business process reengineering, in which the knowledge can be modelling into objects that can be encoded, stored, transmitted and processed, this approach is termed knowledge technology.

Knowledge technologies have emerged as a concept distinct from Knowledge management. Knowledge technology is one that adds a layer of "intelligence" to information technology, to filter appropriate information and deliver it when it is needed.

It is considered to be an emphasis what are the key features that the knowledge technologies should identify to achieve knowledge management's objectives and usability. Some typical features are given below:

- **Content:** the ability to create, store, deliver information with a specific value. The content also requires the ability to manage-aggregate, filter, mine and use taxonomies.
- **Collaboration:** can be synchronous using instant chat and, net-meeting, telephone or asynchronous using virtual workspace and emails.
- **Learning:** including self-paced, collaborative, face to face approaches.
- **Portal:** provides the user an interface to the knowledge allowing personalization, searching and navigation. This can lead to a pervasive experience.
- **Business intelligence and integration:** through application integration, and data aggregation.

2.4.1 Current technologies

Knowledge management research is fairly recent, the first phase occurred in mid-80s to 1990, and was characterised by a degree of experimentation. During this phase researchers were exploring the value created by leveraging the competence and skills of people, innovation and knowledge creation. Knowledge management was not invented as a theoretical concepts but a practical approach to manage knowledge and benefits from it. Then from 1991 to 1997, knowledge management was driven by the IT revolution and explosion of the Internet. The IT solutions and management processes during this time were about re-using knowledge and to avoid re-inventing the wheel. Table 2.1 lists the technologies developed in the past few decades according to the categories of knowledge management life cycle. Recently, knowledge management issues are becoming human concerned. Researchers are beginning to realize that human beings, and not IT systems, are at the core of value creation. More and more people have come to realize the efficiency through IT is not enough. The real value for corporations and society will be generated only by creating environments that enable all people to create and share knowledge. The table also reflects this trend.

Life Cycle Event	Technologies
Creation	Data Mining, Text Mining, Modelling
Storage	Date Warehouse, Workflow, Expert Profiles
Integration	Data Aggregation, Taxonomies, Quantitative Mining, Structured and Unstructured Indexing
Dissemination	Expertise Location, Portals, Personalize Content, Collaborative Filtering

TABLE 2.1: Life cycle events, and typical technologies used for their delivery

2.4.2 Future technologies

It is a commonly held belief that people live in a world where there has been an explosion of data, information and knowledge. But knowledge is only of value when it can be used effectively and efficiently. Knowledge needs to be acquired, modeled and represented, stored and retrieved, used and reused, published and maintained (Shadbolt and O'Hara, 2004).

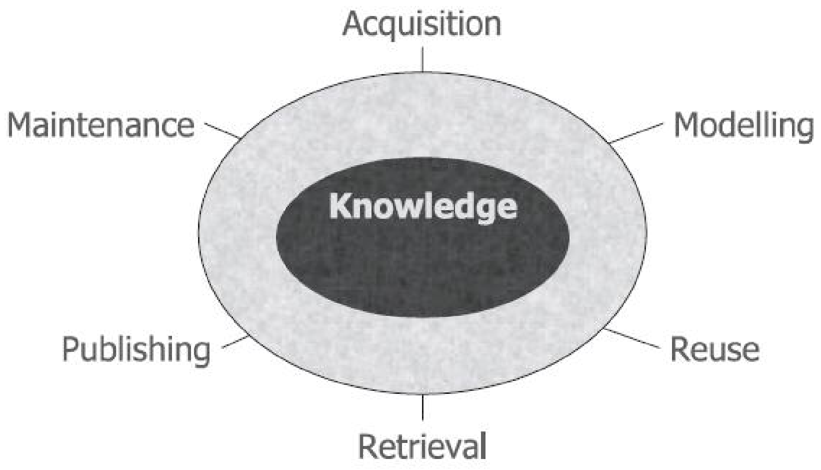


FIGURE 2.3: The Advanced Knowledge Technologies model (Shadbolt and O'Hara, 2004)

Figure 2.3 represented six challenges addressed by Advanced Knowledge Technologies project (AKT, 2003), which is an interdisciplinary research collaboration, aiming to develop technology and understand the dynamics of knowledge within an organisation to facilitate the management of knowledge and extraction of value from it.

- **Knowledge acquisition.** This research is focusing on the development of KA techniques appropriate for the new information-rich environment. This involve making tacit knowledge explicit, acquiring and integrating knowledge from multiple sources, acquiring knowledge from unstructured media, as natural languages.

- **Knowledge modelling.** Modelling bridges the gap between the acquisition of knowledge and its use. The structure must enable usability for problem solving. A central idea will be the use of ontologies, powerful formalisms for expressing and structuring conceptual schemes for domains.
- **Knowledge reuse.** Understand the use and application of knowledge, enable more leverage to be gained from the knowledge already at hand, thereby increasing the returns on investment in those knowledge assets.
- **Knowledge retrieval.** Finding a particular piece of knowledge in a very large knowledge repository, understanding the structure of the stored knowledge in order to navigate through it efficiently and supporting dynamic extraction of knowledge, which alter regularly and quickly during problem-solving.
- **Knowledge publishing.** This work includes the creation of representational forms - web pages, narratives, sets of hyperlinks - tailored to context, using ontologies, which aims to get the right knowledge, in the right form, in the right place, to the right person, at the right time.
- **Knowledge maintenance.** - keeping the knowledge repository functional. This involves assessing the fitness for purpose of the knowledge bases, improving the representational efficiency, regular updating of content as content changes.

2.5 Knowledge Maintenance

The value of a corporate information system tends to degrade with time. This is particularly true if exemplary knowledge (experience, case-specific knowledge) is stored in the information system, as is typically done in experience bases, lessons learned systems, best practice databases, or case-based reasoning systems, because such knowledge is gained almost continuously in daily work (Nick and Althoff, 2001).

Knowledge maintenance can be considered to categorize into following aspects:

- **Enhance** is an essential element in the transformation of information into knowledge. Migrating content to a new system, creating a hierarchy or taxonomy, linking, adding descriptive meta data, and categorizing knowledge into appropriate classifications are some of the steps that help turn information into knowledge.
- **Collaborate** brings people together around the knowledge they create and consume, facilitating the growth and refinement of this body of knowledge, as well as spawning new types of knowledge through this interaction.

- **Manage** allows for the storage, security, and retrieval of organizational knowledge throughout the lifecycle, providing the necessary infrastructure to keep the knowledge fresh and relevant.

While within the knowledge management of engineering design, because of the problem with engineering design, which should be flagged as the size of corporation databases and the stored poor quality of the knowledge. That results in knowledge maintenance being a significant challenge.

2.6 Rich HyperText systems

Hypertext is a framework which enables powerful management of large amounts of information. Using hypertext concepts, it is possible to structure data in different ways, at different abstract level and display and manipulate differently with typed nodes and links in rich hypertext system. For example, if a user interface formally recognized different link types, it could display and manage them differently. The most obvious type distinction is between a website's internal links and links that point to other sites, and an application based on this, is the design icons that notify users about links to internal or external sites. And also, rich hypertext system enables an explicit structure, which frees users from inherent constraints to individual site designs. Users need no longer suffer under bad sites, different navigation.

In hypertext model the two fundamental concepts are node and link. Nodes are the primary information containers. A node has three major parts: contents, attributes and anchors. The node content is where the properties of the hypertext is stored. The attributes are name-value pairs, in which additional node information may be represented. A link represents a binary relationship between two nodes. An anchor is an abstraction which associates some value in a node with a link; the value is typically a position or a region in the node contents. Thus, an anchor makes it possible to come from a node to a link, given some part of the node contents as the starting point of the traversal (Nørmark and Østerbye, 1995).

The model of rich hypertext is an extension of the basic hypertext model. The extension are types entities, type organization and internal node structure. All nodes and types are typed entities, type organization means the types of nodes and links are organized in a hierarchy, in which properties of super types can be inherited to subtypes, while the contents of the node are divided into sub-units, which are organized using the concepts of attributes and anchors.

In rich hypertexts all the nodes have types. The type of a node may reflect its role in the hypertext. The nodes in rich hypertexts are connected by typed links. In many application domains it may in addition be important to ensure that a rich hypertext obeys a set of topological constraints. A structure may be imposed on the contents of nodes as well. Thus, the “richness” stems from the typing of both nodes and links and from the degree of structuring at the microscopic level (inside nodes and links) as well as at the macroscopic level (among nodes and links) (Nørmark and Østerbye, 1995). Rich hypertexts may appear in a variety of different application domains, one such domain is structured argumentation, as represented by gIBIS.

2.7 Conclusion

As knowledge management shifts emphasis from single-handed information to the explicit and implicit processes that using the information, new knowledge management practise are more focused toward changing an organization’s climate, as companies seek to find ways to identify the types of knowledge they have and what they need and how to produce. One outcome of this emphasis is the evolution of enterprise portals bringing knowledge straight to the desktop, which has revolutionized effective business or engineering decision-making. These will benefit the proposed design rationale capture system. In the following chapters of the thesis, the author will discuss knowledge capture in engineering design and proposed the SW-DRS framework for capturing engineering design rationale.

Chapter 3

Engineering Design

An early role of knowledge management was as a support tool for the design engineer generally within the context of product design. The goal of knowledge management as a process is to improve the organization's ability to execute its core business functions more efficiently and effectively. The key to knowledge management is capturing intellectual assets for the tangible benefit of the organization. On the other hand, the aim of any design engineering consulting firm is to produce projects with high quality and in less time.

3.1 The Design Process

Engineering design is frequently described as an ill-defined problem: usually many possible solutions exist, the engineers' objective is to seek the best solution from among these many alternatives. To identify and develop each of these alternatives in an effective manner, a procedure know as the engineering design process is followed.

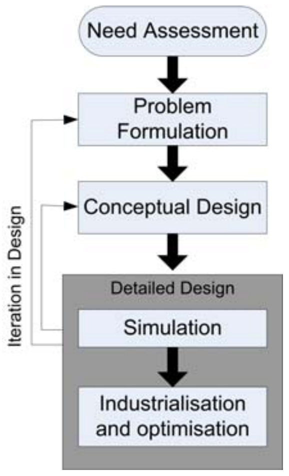


FIGURE 3.1: An engineering design process, based on Fischer et al. (2004).

Figure 3.1 shows a typical engineering design activities. The first stage is needs assessment, which aims to identify the objectives to be achieved by the solution. Then the designer is required to formulate the problem, and structure the search for a solution, identify the design specification and finally identify the required resources. The needs assessment and problem formulation also compose the requirements in other description. Requirements are handled and analyzed to evolve product concepts. Following, starting from the design concepts, designers will define components, arrangement, and dimensions. Once a concept has been selected, the job of producing a detailed design begins, which include a set of drawings and specifications in which the final design of the elements are fixed. After successfully simulating, testing and evaluating a design prototype, the design can proceed with full production. Based on the experiences of the users, the feedback can be used to start this design of the subsequent product. In Figure 3.1 the design process is shown as a linear work, but designing is a non-linear process, there are always iterations driving the activity backward to any stage before the solution produced.

At different stages of an artifact's design process, the activities can concentrate on the features or on the process, (Regli et al., 2000). During the initial stage of the design process, from the need assessment to the concept design, there is concern about the requirement and possible options, for example how it will work, what devices are required. As the design process proceeds, the emphasis moves to the process design - normally concentrating on manufacture requirements and material specification. In the detailed design stage, the objective is to meet the functional and non-functional requirement which leads to more feature based work. It is clear that the design process is both process and feature based, which places requirement on the design rationale tools (Regli et al., 2000).

3.2 Knowledge in Engineering Design

Documents that are produced during the engineering and detail works include design procedures; technical reports; plans that contain engineering layout of the design, the schematic diagrams, and the specific details; and technical specifications classified. Engineers usually look for the precise information related to a certain technical system by project. All these documents are considered to be explicit knowledge that is produced before and they are then built into the company's memory to become internal sources available for future projects. Emails are used to transfer knowledge among individuals within the organization. For example, many procedures, computer programs, lessons learned from experience, guidelines, and design hints are exchanged.

Figure 3.2 shows the type of knowledge that is needed in a mechanical engineering design process. The knowledge needed at the concept level includes criteria, design hints, and

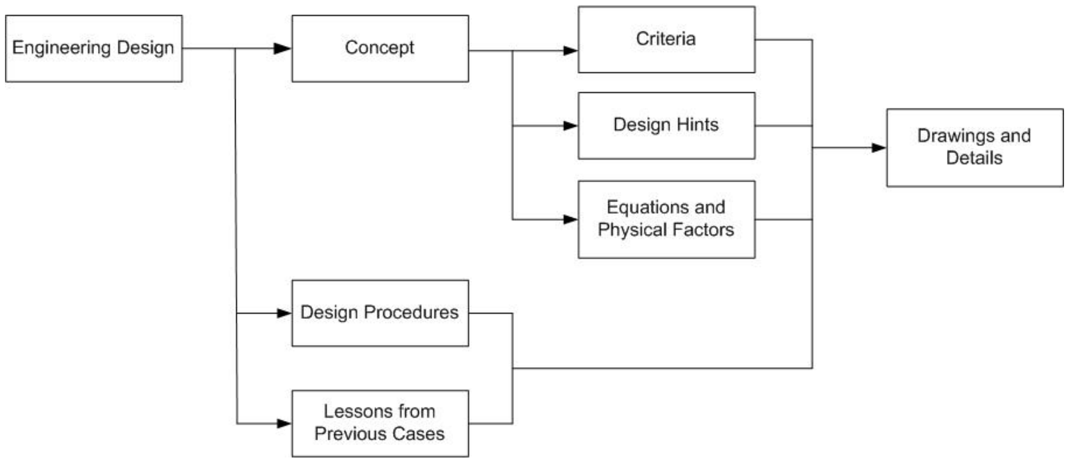


FIGURE 3.2: Type of knowledge and its flow within a mechanical engineering design process (Mezher et al., 2005).

equations and physical factors. Criteria are defined as the constraints and engineering standard that designers should follow to perform their design. Design hints are the keys to a good design free of problems. Engineering work is based on engineering equations that are a base for design.

Knowledge that can be used in the design procedures include calculations, methods, design shortcuts prepared by the engineers, programs validated by the department, and selection programs for equipment. All these procedures have to be validated by lead engineers with greater experience, mainly chief designers and group leaders. This part is the most important since it facilitates the work for a faster design and better production.

Knowledge can be gained from lessons from previous case whereas many problems were faced and solved. These experiences are translated into knowledge where engineers can take them into consideration in future designs. Gathering and managing this type of knowledge is a good step toward eliminating design errors.

Design work is basically knowledge gained from other engineers that faced similar problems. This kind of knowledge can be acquired through formal or informal communication channels. This knowledge gained must be captured and shared throughout the organization.

3.3 Design argumentation

As defined in Chapter 1, design rationale can be considered to be the argumentation behind the design artifact, identifying the reasoning which has been invested in the design of an artifact. The objective of design rationale research is to find the most acceptable and accessible representations for expressing design reasoning, whilst minimizing the effort in its generation while maximizing its usability. Since Rittel's Issue Based Information Systems was presented in 1970s, the design rationale research has invested on argumentation design rationale for decades and produced a variety of extensions. Shum (1996) identified three strong influences which serve as historical and conceptual roots to the current interest in argumentative design rationale. The author also finds that a significant number of design rationale researches in US are based on IBIS, and conversely others in Europe are based on Toulmin. The author will introduce and compare the two methodologies in this chapter.

3.4 Representation of rationale

The majority of work on design rationale has concentrated on capture and representation. The representation of design rationale has been studied extensively. (Lee, 1997) claims that design rationale representations range from formal to informal. Informal representation captures rationale in a natural language, audio/video recordings and raw drawings. The information is ill-structured and only provides limited computational services. But as the author defined, design rationale should go further more. In a formal representation, information is defined as formal objects that the system can interpret and manipulate, providing computational services. But they do not output information in a form that a human can understand. Combinations of both of these approaches can overcome their individual limitations, so the existing systems were mostly built using semi-formal to represent argumentation.

3.4.1 Issue Based Information System (IBIS)

IBIS can be tracked back to the early work of Kunz and Rittel (1970). IBIS was developed to provide a simple yet formal structure for discussion and exploration of wicked problems[1]. An issue is an identified problem to be resolved by deliberation, followed by one or more positions that respond to the issue. Each position can have any number of arguments that support or object the position. One issue can either generalize or specialise from another issue. IBIS organize the deliberation process into a network of these elements and their relations. The key element of IBIS is *issue-position-argument*, of whom the relationship is illustrated in Figure 3.3. Other version of IBIS systems are gIBIS (graphical IBIS) and itIBIS (indented text IBIS).

The Design rationale editor (DRed) (Bracewell et al., 2004) is one of the many proposed derivatives of the IBIS concept. DRed is a software tool, which is currently being used in aero engine design. It provides a facility for structuring the electronic design

[1]One clear sign of a wicked problem is that there is no clear agreement about what the real problem is. Wicked problems can not be solved in the traditional sense, because one runs out of resources -time, money, energy, etc.before a perfect solution can be implemented.

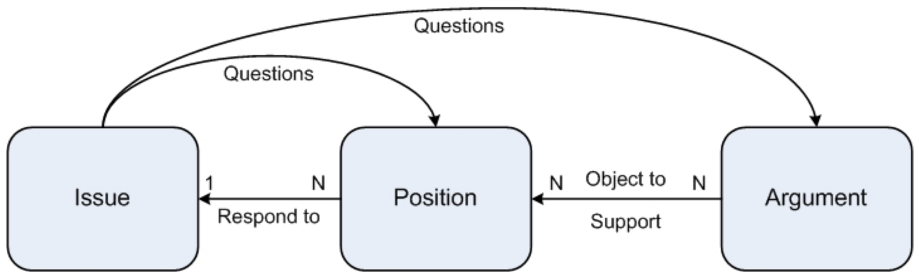


FIGURE 3.3: The relationship of issue, position and argument
The notation 1-N indicates one to many, and N-N indicates many to many.

information generated during design and is intended to work together with CAD, office and web applications.

3.4.2 Procedural Hierarchy of Issues (PHI)

In 1970s, a variety of projects were undertaken that attempted to use IBIS to solve real world problems, but it was found that they were unable to adequately support design tasks. As a result researchers found a fundamental problems with the IBIS method, (Fischer et al., 1996; McCall, 1991). Fisher et al identified that two types of information are omitted from IBIS. One is the dependence relationships between issue resolutions, that is, relationships representing the fact that answering an issue often depends on how other issues are answered. The other one was questions that were not deliberated, that is, questions for which the pros and cons of alternatives answers are not considered. PHI (an extended IBIS) is introduced to overcome the drawback. It broadens the concept issue and alter the elements to issue, answer and argument. First, it simplifies relations among issues by providing the ”serve” relationship only. Secondly, it introduces two methods to deal with the issues: deliberation (argumentative process) and decomposition (break down the issue into a series of sub-issues). The prime issue is by definition the whole project and PHI always ends with the resolution of prime issue. A more detailed comparison between PHI and IBIS can be found in McCall (1991).

3.4.3 Design Space Analysis (DSA)

DSA makes the analysis of the design space the central issue. The key elements as indicated in Figure 3.4 are: *questions* highlight key design issues, *options* refer the possible answers to the questions, *criterion* for assessing and comparing the options. In addition, *Assessments* are the relationships between Options and Criteria (supports or objects-to), and *Arguments* are used to conduct debate about the status of the above entities and relationships. While the issue based representation aims to capture the history of design deliberation. In contrast to IBIS-based approaches which aim to capture spontaneous design deliberation, the Question, Option and Criterion (QOC) representa-

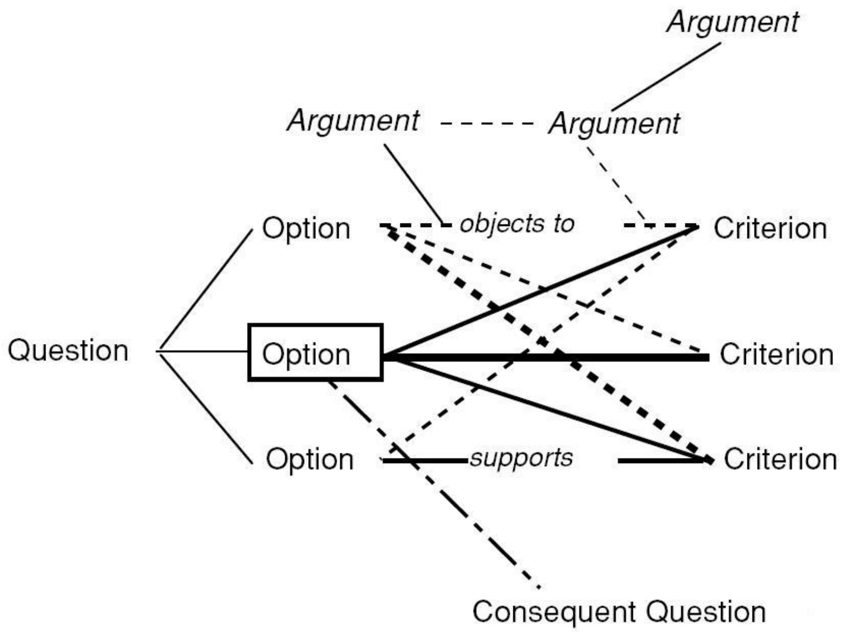


FIGURE 3.4: The generic Question-Option-Criterion vocabulary (MacLean et al., 1991). The link-thickness is used to indicate relative weights of the assessment.

tion, (MacLean et al., 1991), encourages the systematic development of design Options structured by Questions, choice amongst them and also the considerations that lead to choice.

3.4.4 Decision Representation Language (DRL)

DRL was introduced by Lee (1990), which aim at being expressive enough to represent design rationale and providing enough services to reward the user. Five design spaces are defined in DRL: Argument, alternative, evaluation, criteria and issue. Each space holds part of the information about the complete design.

As show in Figure 3.5, Alternatives represent the options from which to choose. Goals represent the properties that an ideal option should have. A Decision Problem represents the problem of choosing the Alternative that best satisfies the Goals. Each Alternative is related to a Goal via an Achieves relation, denoted as Achieves(Alternative, Goal). A relation in DRL is a subclass of Claim; in particular, the relation Achieves(Alternative, Goal) represents the claim that the Alternative achieves the Goal. The overall evaluation of an alternative is represented by the plausibility of the relation, for example, the claim, Is-the-Best-Alternative- For(Alternative. Decision Problem). The plausibility of this relation, in turn, is a function of the plausibility of the Achieves relations between the alternative and all the goals as well as of the importance of these goals. An Alternative is evaluated by arguing about the plausibility of the Achieves claims linking the alternative to each of the Goals, and about the importance of the Goals. More generally, one argues in DRL by producing a Claim, which can Support, Deny, or Presuppose other Claims.

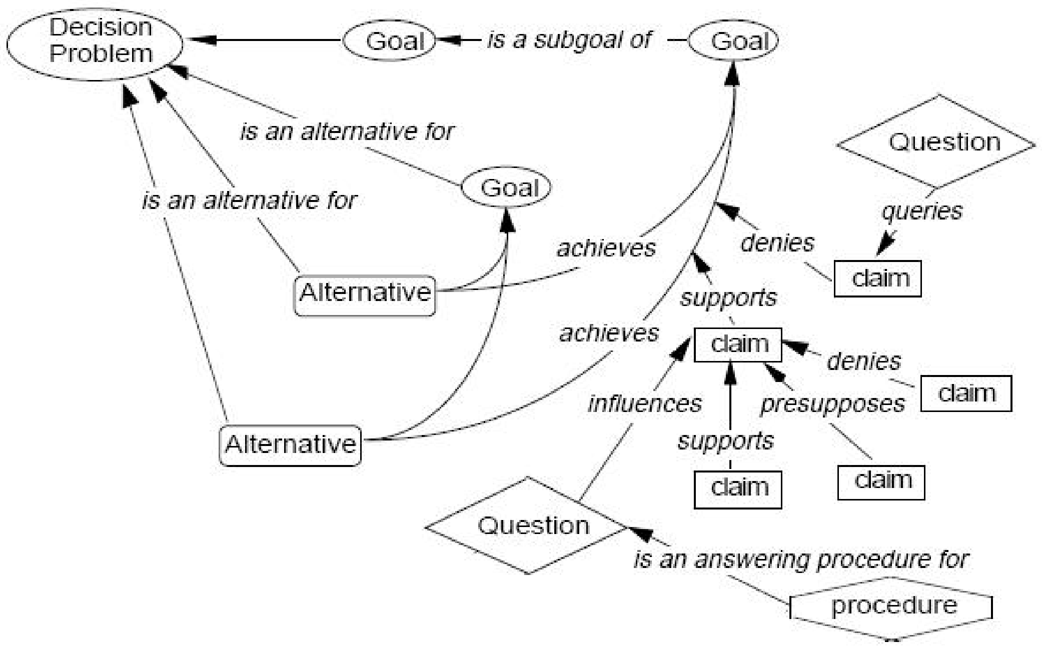


Figure 3.5: The object types that form the Decision Representation Language vocabulary (Lee, 1990)

These relations – Supports, Denies, Presupposes – are, as mentioned above, claims; as such, they, too, can be argued about. A Question Influences a Claim if the plausibility of the Claim depends on how the Question is answered.

3.4.5 The Toulmin Structure

Toulmin's theory is used in a variety of communication classes including the Fundamentals of Communication course, Public Speaking, Business and Professional Speaking, Communication Theory, Interpersonal Communication, and others. This theory is also used in some philosophy and English composition courses.

Toulmin (1958) identifies the three essential parts of any argument as the claim, the data and the warrant. *Data* (Grounds) is the evidence, facts, data, and information that are the reason for the claim in the first place- a reasoned beginning. A *Claim* is the position on the issue, the purpose behind the argument. The *Warrant* is the component of the argument that establishes the logical connection between the data and the claim. "the arguments in which the warrant entitles us to argue unequivocally to the conclusion" (Toulmin, 1958).

Toulmin also identified other elements: backing, qualifier and reservation. *Backing* is support material that supports the warrant in the argument. Backing can help understand the reasoning used in the warrant. Because Toulmin advocated practical reasoning, all arguments have relative strength. *Qualifiers* represent the verbalization of the rela-

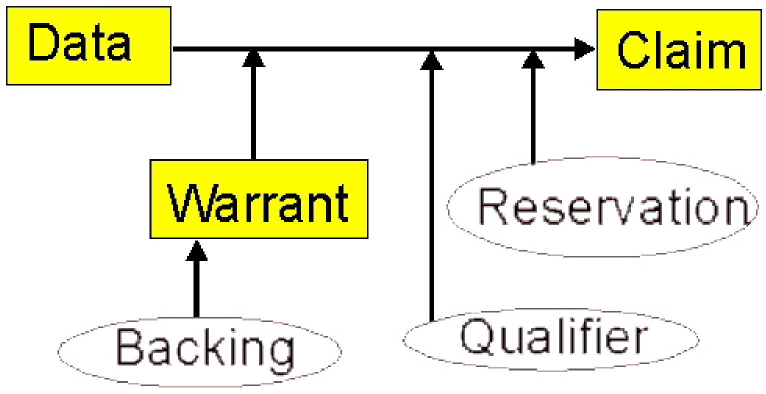
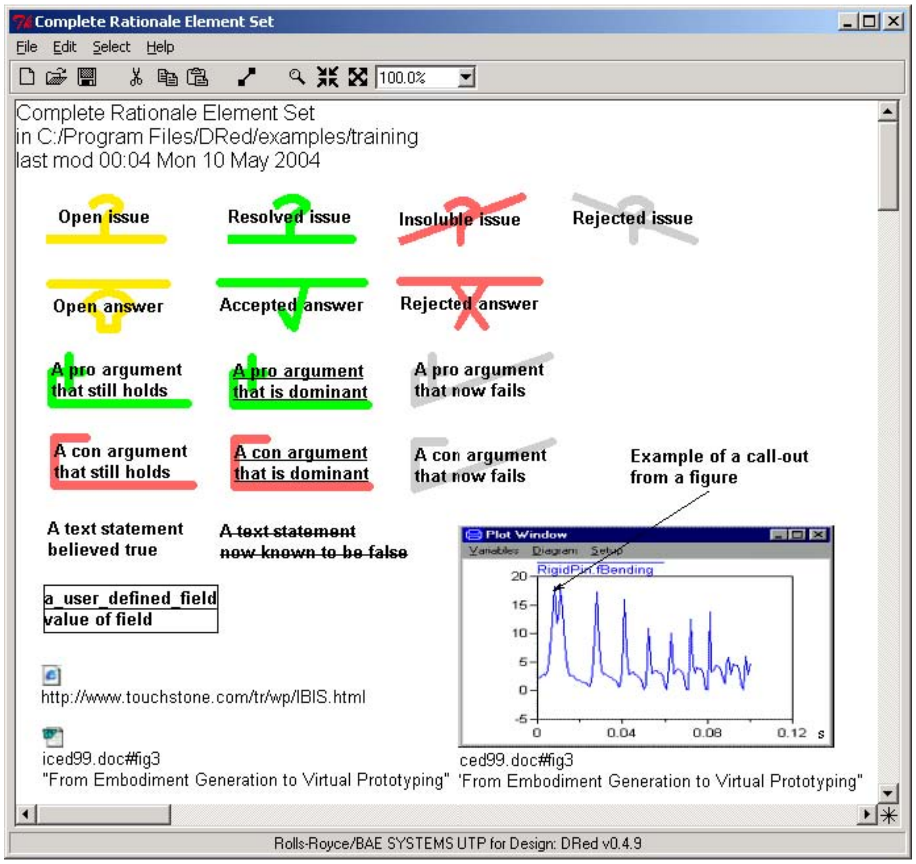


FIGURE 3.6: Toulmin Structure (Soukup and Titsworth, 1998)

tive strength of an argument. A *Reservation* or *Rebuttal* is an exception to the claim presented by the arguer. In Toulmin's model, arguments are not considered universally true. The Reservation demonstrates how arguments can be strengthened via the limitations of the argument.

3.4.6 DRed (Design rationale editor)

A new IBIS-based software tool called DRed has been developed by researchers in Cambridge for Design in the Engineering Design Centre (EDC) in 2004 (Bracewell et al., 2004). DRed allows designers to record their design rationale at the time of its generation and deliberation. The design rationale is displayed in a document as a graph of nodes linked with directed arcs. The user creates the nodes by choosing from a predefined set of element types. The key element types are: issue, answer, and argument. And each type has a number of states. For instance, issue can either be open, resolved, insoluble or rejected. Answer can be open, accepted or rejected. A completed definition of rationale element set in DRed in shown in Figure 3.7. Recent research has been aimed at determining if DRed improves the richness of the recorded information. DRed documents were analysed to investigate the nature of the key element types and were compared to Design Definition Reports, which are a form structure for engineering company to structure design requirements (Aurisicchio et al., 2006). Results indicated that DRed issues were predominantly phrased as questions. The questions were mainly formed to address design problems. Using DRed, designers mainly captured questions to address the generation and analysis of new solutions. The engineering processes captured and structured in DRed, compared to those presented in the Design Definition Reports, are richer in the number of recorded design solutions and in the number of pro and con arguments underpinning those solutions.


Complete Rationale Element Set
File Edit Select Help
100.0%
Open issue
Resolved issue
Insoluble issue
Rejected issue
Open answer
Accepted answer
Rejected answer
A pro argument that still holds
A pro argument that is dominant
A pro argument that now fails
A con argument that still holds
A con argument that is dominant
A con argument that now fails
Example of a call-out from a figure
A text statement believed true
~~A text statement now known to be false~~
a_user_defined_field
value of field
Plot Window
Variables Diagram Setup
RigidPin.fBending
20
15
10
5
0
-5
0
0.04
0.08
0.12
s

Complete Rationale Element Set
in C:/Program Files/DRed/examples/training
last mod 00:04 Mon 10 May 2004

http://www.touchstone.com/tr/wp/IBIS.html

iced99.doc#fig3
"From Embodiment Generation to Virtual Prototyping"

ced99.doc#fig3
'From Embodiment Generation to Virtual Prototyping"

Rolls-Royce/BAE SYSTEMS UTP for Design: DRed v0.4.9

FIGURE 3.7: The available DRed elements

3.4.7 Other approach

- **Functional Representations (FR).** Chandrasekaran and Iwasaki (1993) showed a representational scheme, describing how the device works (or is intended to work). In the functional representation scheme, design rationale is used as an account of how the designed artifact serves or satisfies expected functionality. One can use FR to capture the causal components of design rationale. FR takes a top-down approach to represent a device: the overall function is described first, and the behaviour of each component is described in the context of this function. FR encodes the designers account of the causal processes in the device that culminate in achieving its functions. Tasks that design rationale should be able to support are: control of distributed design activity; reassessment of device functions; generation of diagnostic knowledge; simulation and design verification; redesign; and case-based design. FR provides a partial rationale for choices made about components and their configuration; its limitation is that FR only captures the causal knowledge about device operation.

- **Active Design Documents (ADD) (Garcia et al., 1993)**, is a design rationale system for routine, parametric design. By having a domain and task specific knowledge based approach, the designer can assign parameters. If the designer's recommendation matches the system's, the system records rationale already built into the knowledge base. If there is a conflict between the designer's action and the system's, the designer is informed and allowed to either modify the criteria, change their action, or override the system's recommendation. ADD also allows the designer to simulate parameter changes.

3.5 Approaches to capturing design rationale

Producing and capturing design rationale is a major difficulty in creating a design rationale system. The ideal design rationale system would be non-intrusive. This is desirable because recording rationale is not only time consuming for the designer, it also can distract them from the design task they are performing. On the basis of when to produce the rationale and designers' interactivity to capture rationale, the author divides design rationale capture methods into three categories:

- **Reconstruction (Lee, 1997).** This is usually after design performed, hence outside the design process. The advantage of this approach is that it is non-intrusive, the disadvantage is that it may not accurately or completely capture the rationale
- **User intervention (Bracewell et al., 2004).** In this approach, the rationale is produced during the design process. This is often done by having the designer use a methodology that aids in capturing the rationale. Comparing the reconstruction, this method can capture more accurate rationale, the drawback is that it may require designers' extra effort to build the rationale and disturb normal activities.
- **Automatic (Garcia et al., 1993; Molavi et al., 2003).** This is the ideal approach to produce design rationale, it assumes there is a method to capture the communication among the designers and team members. This means the designer need not do anything but pursue their normal design custom. It will not disturb the design process because it does not perform intrusively.

Table 3.1 shows the comparison of these approaches. The author compares these approach by three aspects: produce time, interaction, accuracy, which specify design rationale is built after or during the generic design process, seldom or a large amount of effort required by designers and the accuracy means the efficiency of the captured knowledge, how much it will help better design.

Method	After/During Design	Low/High Interaction	Low/High Accuracy
Reconstruction	After	Low	Low
User intervention	During	High	High
Automatic	During	Low	Various, Depends on the logic, approach of system

Table 3.1: A summary of the approaches to the methods used to capture design rationale

3.6 Summary

This chapter has defined what is understood by the design rationale and reviewed a number of competing approaches, which have had significant influences in design rationale research to present and capture design knowledge. To provide a general view of the development of the research into design rationale capture, the design rationale systems reviewed in this chapter are summarized in Table 3.2. Each system is described in two dimensions: type and capture method. The following sections will concentrate on a detailed discussion of the systems in the table.

As identified in the table, most systems are built on argument approach, IBIS both serves the design process order and reason logic, so the author will investigate the research on building a new argumentation structure which is based on IBIS methodology. The ideal capture will be automatic and not many system implement automatic generate rationale, therefore a significant effort is needed to produce the rationale both non-intrusively and accurately.

IBIS serves as a source of inspiration for argumentative design rationale research. However, IBIS enforces a structure on how issues are discussed but not on how the problem is explored, how alternatives are elicited and evaluated and, how consensus is reached. In the next chapter, a decision making model based on ontology will be introduced and as well a framework to capture and retrieve design rationale.

Notation	**System Name**	**Type**	**Capture Method**
IBIS	IBIS (Kunz and Rittel, 1970)	AB	
	gIBIS (Conklin and Begeman, 1988)/itIBIS (Conklin and Burgess-Yakemovic, 1991)	AB	User intervention
	Viewpoint (Fischer et al., 1989a)	AB	N/A
	DRed (Bracewell et al., 2004)	AB	Reconstruction/User intervention
PHI	JANUS (Fischer et al., 1989b)	AB	Not specified - assumes KB of rules and example designs exists. rationale use, not capture
	HOS (Shipman and McCall, 1997)	AB	Reconstruction
	PHIDAS (Shipman and McCall, 1997)	AB	Reconstruction
DSA	QOC (MacLean et al., 1991)	AB	
	DRARS	AB	N/A
DRL	SIBYL (Lee, 1990)	AB	Not specified - assumes KB of rules and example designs exists. rationale use, not capture
ADD	(Garcia et al., 1993)	ADB	User intervention
Toulmin	(Toulmin, 1958)	AB	N/A

Table 3.2: Table of existed design rationale systems. Type: Argumentation-base (AB) or Active Document-based (ADB)

Chapter 4

A Framework for Capturing Design rationale

Most work on design rationale has concentrated on capture and representation. Capturing, or recording, design rationale is time consuming and expensive. The more intrusive the capture process, the more designer resistance will be encountered. Because it is time consuming and viewed as documentation, design rationale capture is viewed as expendable if deadlines are an issue (Conklin and Burgess-Yakemovic, 1991). That is the reason why existed systems fail to serve the objectives, so better structured approach to capture is needed.

The system proposed by the author, the *Semantic Web based Design rationale System*, SW-DRS will primarily help model the design activity and hopefully will result in an easy, effective and better design, rather than in domain-knowledge about the artifact designed in a CAD system. As the development in software engineering and artificial intelligent community, more and more systems are designed as objects, components or services oriented for easy integrating with other systems and reusing. The author describes a loosely coupled architecture later and defines the elements and their relationship during design activities.

4.1 Scenario

The engineering design environment is highly distributed in nature and is characterised by a large number of information sources, which together with the designers forms a complex sociotechnical system (Crowder et al., 2003). In this paper the researchers define a future engineering design environment as a scenario, with particular emphasis on the social and technical systems that will support designers in their day-to-day activities. SW-DRS can be considered to be an implementation of the research on design supporting environment based on this scenario. The use of SW-DRS can be illustrated as the following scenario.

> *Zed is a designer within a mechanical engineering company, he is requested to undertake a specific design task. Zed initially discuss the requirement with the customer or needs assessment staff, which aim to identify the objectives achieved by a solution. Using an application termed the Designer Knowledge Desktop, Zed create an online design folder*[1] *and define a prime issue.*
>
> *To progress the design, Zed obtains previous work and background information. The Designer Knowledge Desktop assumes that most of the knowledge in this domain is well structured and easily accessed. The domain knowledge is described using predefined ontology and stored in knowledge database either centrally or distributed. There are also web services available to access the knowledge database over the Internet and intranet. Knowledge tools can downloading the relevant information or make it available for browsing depend on an application specific ontology employed. Previous work and similar design are constructed in a design space, for example the online design folder. All the elements produced during the design process are modelling into objects and stored associately. The design result in a final solution, which is presented in a graphic network of issues, proposal answers, argument and quantitative selection criteria, with attached emails, faxes, models and trial results etc (Crowder et al., 2003).*

In the scenario it is assumed that the technical tools of the future design environment have been embodied in an application termed KTfD (Knowledge Tools for Designers). The proposed architecture is shown in Figure 4.1, showing that all the information that can be accessed through the KTfD desktop. All the documents are stored locally, ensuring that they are available even though the original resource might be irretrievably lost. KTfD is able to access information from anywhere in the design office or company, through the local wireless network. The designers use electronic logbook for a degree of pervasive computing, as it permits active reconfiguration as a function of location. It is recognised that the KTfD is not only for knowledge management, but also has access to the full range of office and data analysis tools.

To implement this system, application will be required in standards, means web services and ontologies. It is recognised that populating the knowledge base and the associated links is a key issue. All objects within the knowledge database are version controlled and

[1]All the documents in the online design folder are shared amongst the design team. Individuals in the design team have different security permission, for example, process manager can view the issue-solution process through his own Designer Knowledge Desktop and other designers can check out any object defined in DKD, correct it or define a new object. As the object are all version controlled, only the latest version work, and the process can reverse and forward to any stage of the process.

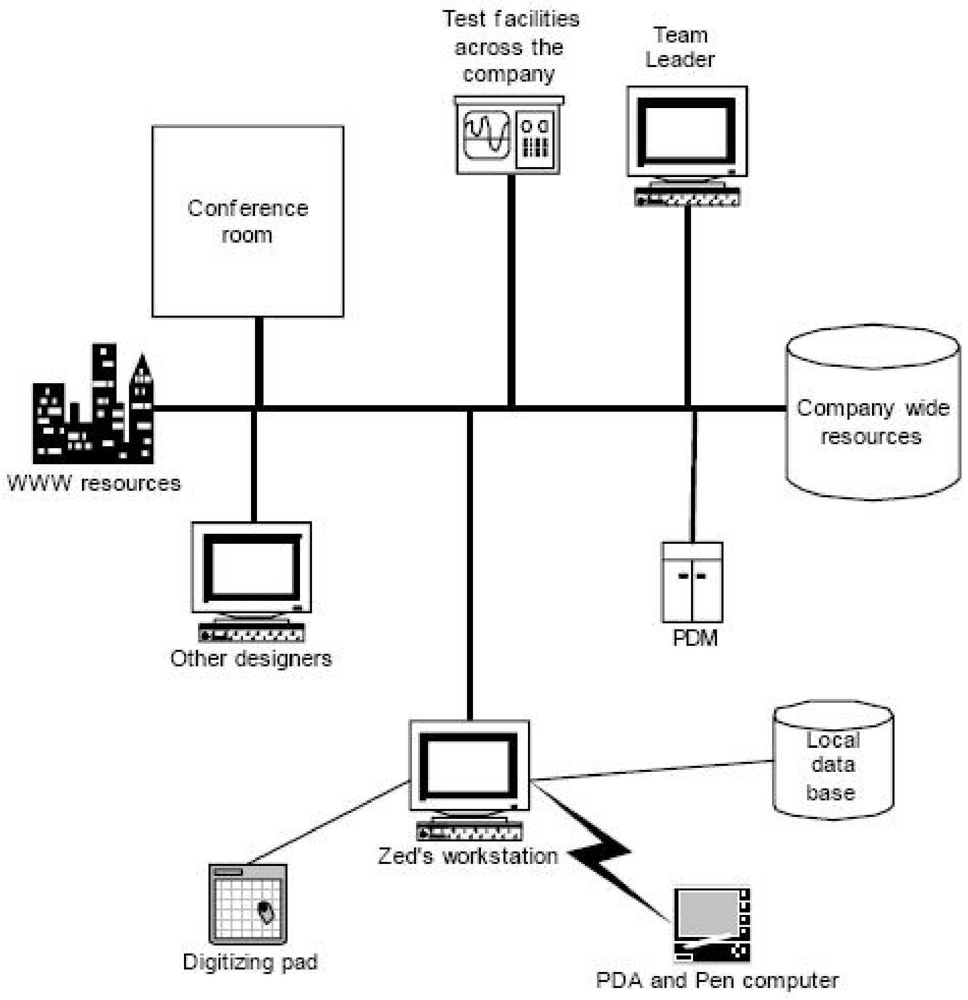


FIGURE 4.1: Proposed KTfD architecture (Crowder et al., 2003).

Designer Knowledge Desktop will act as a knowledge management tool for the designers to gather, access and analysis all the information required to performance their daily tasks, and also access to the full range of commercial tools in the organization.

4.2 The Semantic Web Solution

Semantic Web is a new vision of web, which can be viewed as a knowledge management environment introduces new requirements, including the ability to extract metadata and learning ontologies. In engineering design, it is an effective way to describe its information, but there is little metadata to describe the design process and also the rationale which produced during the design. In particular, designer search the requiring information based on a method mainly by matching text strings, lacking precision. Moreover, design is a decision instrument to express product features and production information, thus decisions made during design process have severe influences on the success of the product. To improve the design process performance, establish shared value particularly in the context of project related knowledge can improve collaboration amongst designers, and therefore allow them to make accurate decisions in order to reduce the potential cost of time and effort.

The Internet associated with its most popular application, the WWW, provides interconnected infrastructures that are commonly used to facilitate the accessibility of digital resources. The vision of a Semantic Web was created by Tim Berners-Lee in order to enable automated information access and use based on machine-processable semantics of data. Tim Berners-Lee defined the Semantic Web as "an extension of the current web in which information is given well defined meaning, better enabling computers and people to work in co-operation" (Berners-Lee et al., 2001).

Ontologies are specific vocabularies of concepts and their relations amongst these concepts. As defined by Grüber (1993), an ontology is a formal explicit specification of a shared conceptualization. Ontologies permit computers to better categorise, retrieve, query and deduce information than the current technology. The concept of ontology applied in Artificial Intelligence is to facilitate knowledge sharing and reuse. Ontology is claimed able to provide a shared and common understanding of a domain so that people and various application systems can communicate across the widely spread heterogeneous sources. In this respect, ontologies are useful to organize and share information while offering intelligent means for content management as well as enhancing semantic search in distributed and heterogeneous information sources. It should be appropriate to apply such a strategy to the domain of engineering design to support better information as well as knowledge sharing amongst the designers.

The W3C has made significant progress toward standardizing the specification necessary for the Semantic Web. As show in Figure 4.2, W3C finalized RDF which builds on XML, providing a data-modeling framework for knowledge with structure based on triples of subject, predicate and object. Building on RDF, RDFS is a simple ontology-modeling language that uses concepts such as classes, subclasses, properties and domains. The Darpa Agent Makeup Language (DAML) in the US and the Ontology Inference Layer (OIL) in Europe are extensions of RDFS, offering a richer ontology language. W3C also finalize the Web Ontology Language (OWL) based on these two languages. Other researchers also have been working on tools for creating and editing ontologies. One of the first also still the best known is from Stanford University. Protégé lets users construct a domain ontology and enter domain knowledge.

By considering the typical engineering design process, it is clear that engineering design is also committed to virtual-organizational business process and relationships, which are mainly project oriented. The benefits of using semantic web technology in engineering design rationale capture can be summarized as follow:

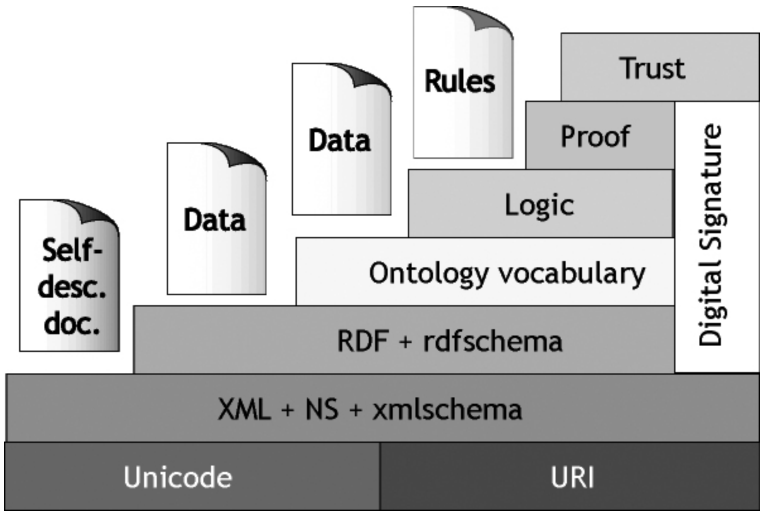


FIGURE 4.2: The Semantic Web "layer cake" (Berners-Lee et al., 2001) shows ontologies and semantic data being built on previous layer.

- Structuring

  The knowledge organized by using semantic web technology has the following traits: knowledge description, the scope and content of the description is widely extended based on knowledge, thereby forming more accurate understanding of information; common representation of rules and structure which are the prerequisites for knowledge exchange in different systems.

- Sharing

  Reuse and sharing of knowledge is an integral aspect of the Semantic Web. Associations are described as typed link. Therefore, link-types are knowledge relations, a set of link types may be represented as an ontology. Also the ontology represents relationships of domain concepts, including domain models, enrich links and product content. In Semantic Web environment each user in this community or organisation can browse and search the knowledge base, which enable other users or organizations to reuse and share the content.

- Traceability

  Many ideas related to design rationale include traceability, since it is in the interest of designers to be able to trace the chain of reasoning why particular choices have been made. This type of traceability can be implemented as creating traceability through the use of semantic links between paragraphs of documents, or between concepts in a knowledge database.

4.3 Why Semantics

A formally defined design rationale representation may allow the integration of the formal semantics of the artifacts being designed, and allows automated computations over such representations. When such representations are available in a distributed environment, it is possible to envisage the collaboration between designers with semi-automated support, where design rationale representations can be searched for, recovered and integrated during the process of designing a new artifact. Such availability can therefore be the basis for collaborative design, among designers working with a given artifact.

The integration of different design rationals is possible only if the following conditions are satisfied: the artifacts are built from the same type of formal model; the design rationals are used to represent the same domain of application (e.g. a robotic design) and the design rationale of the artifacts is represented using the same (or compatible) representation scheme(s) (e.g. a common ontology vocabulary) (Medeiros et al., 2005).

A semi-automated computational environment is being built to support designers through processing of formal design rationale representations. This environment uses the formal model for the artifact being designed to suggest design options at each step in the design, and records the corresponding choices made by the designer, using a special purpose description language that will be described later. Depending on the richness of the formal model of the artifact being designed, the system may suggest new alternatives, and also check the consistency of decisions made by the designer. Theoretically, when formal semantics for the artifacts are available, fully automated systems could be constructed to automatically synthesize artifacts, but this is neither the approach nor the focus taken in this paper. The author explicitly require human intervention in defining design steps or operations in producing the final design.

4.4 Requirement and services

A lot of work is done on an artifact after it is finially designed. It is a major requirement to have a recorded process that supports effective design. Successful engineering design approach requires that design rationale be presented in an easily accessible way. In design activities most of the information is taken from previous solutions, as a result the rationale is essential to provide context for an engineer designer who need to understand the design decisions and all the constraints associated with the previous design.

Record design history. This service as a design event log, records the raw data of the design process on a timespan. This information will provide a rich resource when appropriate rationale construct method introduced. Furthermore, different perspectives

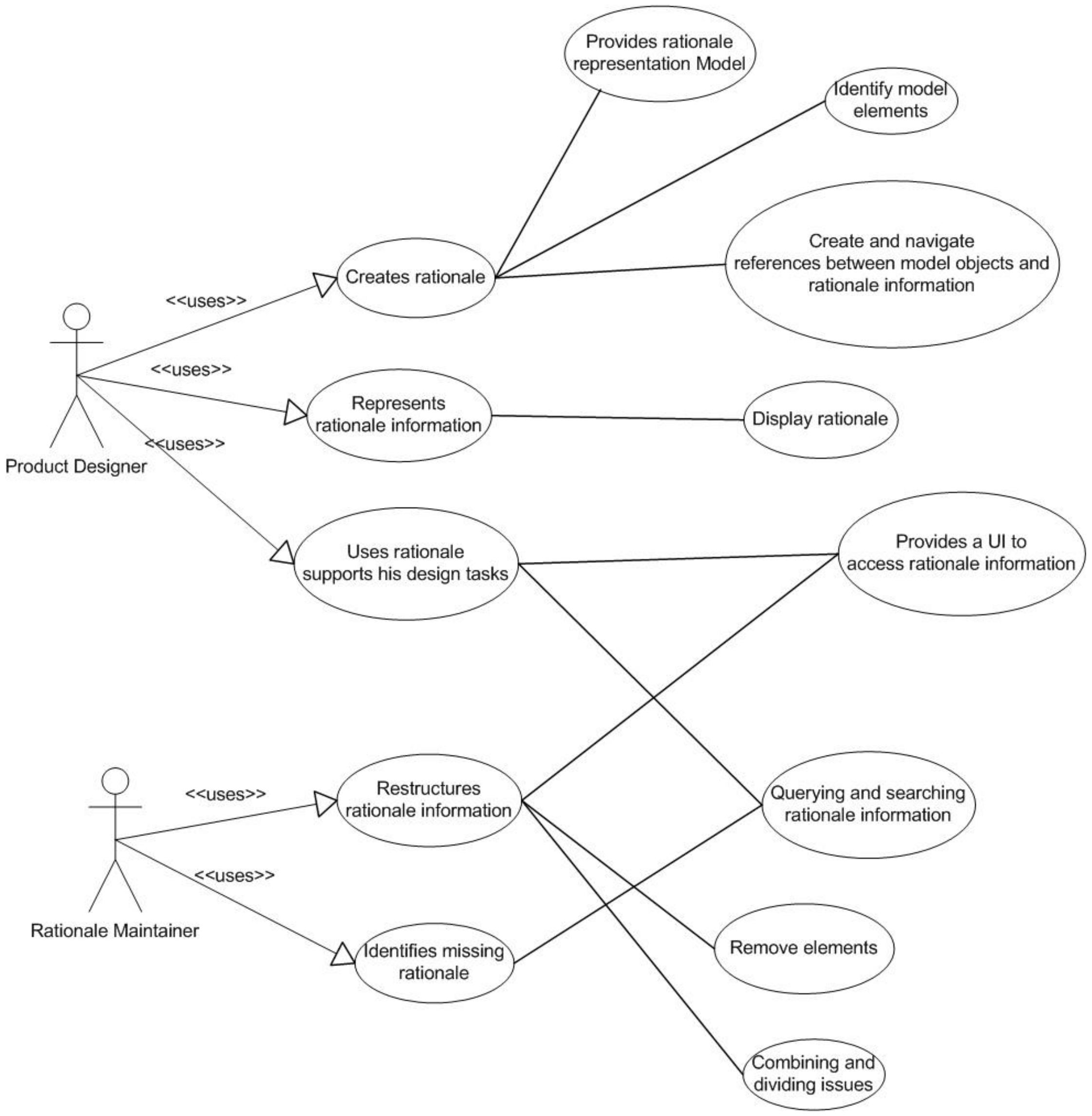

Provides rationale representation Model
Identify model elements
Create and navigate references between model objects and rationale information
Creates rationale
<<uses>>
<<uses>>
<<uses>>
Represents rationale information
Display rationale
Product Designer
Uses rationale supports his design tasks
Provides a UI to access rationale information
Restructures rationale information
<<uses>>
Querying and searching rationale information
<<uses>>
Rationale Maintainer
Identifies missing rationale
Remove elements
Combining and dividing issues

FIGURE 4.3: The use case for design rationale system identifying two users, the product designer and the maintainer of the rationale.

can benefits various members in engineering community by extracting rationale from this raw data.

Argumentation. Design teams exchange knowledge about artifacts they developed. For instant, design team members produce possible solutions to meet the requirement, reinforce the ideas, agree or disagree the solution and provide the reasons. The design process should be represented in a network of reasoning logic, which will provide rich semantic knowledge for argumentation.

Generic(domain independent) tools. By making general tools, one can avoid developing multiple tools doing slightly different jobs. Costs are thus cut by loosely couple architecture and reusing certain component. That make a complicated system decompose into diverse components, which thus minimize maintenance and redundancy cost.

Multiple designer support. Many of the current design rationale system are not set up for multiple designers. The multiple support will allow information sharing, fully argumentation and collaboration.

Communication. Design rationale capture often occur during designers' communication, for example, email, oral records and regular meeting including the use of the whiteboards. The service which should transform this information to electronic record will be essential but not sufficient condition for its retrieval. For effective retrieval, communication must be well indexed. This indexing is often difficult and in practice it is seldom done adequately. It tries to record the thoughts or design event occur rather structure them. It lacks discipline and logic of the design rationale thus merely provide the participants of the team to track down the information.

Minimize burden for constructing rationale. Although the tremendous value of explicit design rationale is well accepted, the engineers will resist if substantial time required for constructing or the tools change the manner in which engineers work. Especially, a system will be proved useless if the cost of building semantic rationale overcome the benefits that the structured information can provided.

Learning. Unlike the most other requirements which should be meet during design process, this will help after design. The knowledge that design decisions will be made known to people outside the project group has the potential to influence those decisions, help designers and developers form other community Documentation better understand the design. System will provide designers' reasoning in a form that will be clear to people outside the project group.

Constraint management. Design process can be viewed as a process of managing the dependencies between requirement and solution (components that implement the requirement). The ideal design rationale system should explicitly express these dependencies as design constraint, which reflect the relationship among design parts, decisions, argumentation, options and avoid inconsistence.

The detail use case is shown in Figure 4.3, the objectives of this research is to help designer make a more effective and better design, which are not like a commercial design rationale system that also supports rationale maintainer and rationale manager. To demonstrate the design rationale model and the system framework, it is a key issue to reduce the effort in building the system, avoid the technical problem when building complex distributed system. Implementation of a design rationale system as a commercial one for diverse user roles is unrealistic and time consumed. As a result, the goals of SW-DRS is to support designers in creating and representing design rationale and to capture, annotate, and cross-reference the generated rationale and information.

4.5 IBIS vs Toulmin

IBIS is recognised as one of the most known approaches, which grew out of Rittel's 1970s work (Kunz and Rittel, 1970) on an Issue-Based Information System (IBIS) to attempt the additional capture of the rationale behind the design. This includes the decisions required; all the alternatives evaluated; the reasoning behind the choice of one alternative over another; and the dependencies between earlier and subsequent decisions.

The second root concerns the representational form for decision logic. *The Uses of Argument* by Toulmin (1958) was originally introduced as a challenge to the dominance in philosophy of formal, Aristotelian logic (Shum, 1996). Toulmin's aim was to develop a view of logic which was grounded in the study of reasoning practice. The view of the logical structure of arguments led to a graphical format for laying out the structure of arguments, which continues to be pursued in much subsequent argumentation work.

The author begins by using these two methodology to capture a simple design case, and then compare the rationale captured and represented as graphical forms, finally summaries how and the differences between IBIS and Toulmin argumentation rationale representation. To explain the key differences, the author structures the redesign for export of the Eastman Kodak's Fun Saver Panoramic 35-mm single use camera (Figure 4.4). In 1993 Kodak researchers began to work with people in Europe, Japan, and United States in order to identify those feature that should be incorporated into the next-generation design of this camera (Voland, 1999). A simple design for a new camera with a chargeable flash is captured by both IBIS and Toulmin methodology.

> *"One example of a product that was redesigned for export is Eastman Kodak's Fun Saver Panoramic 35-mm single-use camera. (The consumer returns the camera, together with the film inside, for developing; a new camera then must be purchased for further picture taking.) In 1993 Kodak researchers began to work with people in Europe, Japan, and the United States in order to identify those features that should be incorporated into the next-generation design of this camera.*
>
> *It was found that Japanese consumers wanted a design that would be thinner, lighter, and require less time to recharge its flash. German consumers sought greater recyclebility of the parts, whereas other respondents desired rounded corners the would make the camera more comfortable to carry and use.*
>
> *Because of this focus upon the world market, the camera was redesigned to fit around the film pack, thereby minimizing its size. Rounded corners, a plastic ellipse the prevents a users from inadvertently blocking the picture, and automatic recharger so that the flash will always be available to light a scene, and components that either can be reused or recycled also were introduced. Furthermore, the needs of photofinishers were reflected in the use of*

Figure 4.4: Eastman Kodak's One-time-use KODAK FUN SAVER Panoramic 35 Camera.

a standard (35-mm, 400-ASA) film and a cover that could be opened easily. (Voland, 1999)"

The author developed the rationale for this camera design by IBIS method, as shown in Figure: 4.5, the design case begins with the premier issue "camera design", while the premier issue is often too complicated to be answered by one single solution, so that it is divided into a series of sub-issues: shape design, flash design, material design.

In engineering design, it often begins with the design questions that must be answered, or open issues that must be addressed. In other word, there are unsolved questions with an amount of solutions to be argued or considered. While Toulmin structure is inappropriate in indicating the situation that a number of solutions address one single design question and a number of arguments relate to one single solution. In other hand, IBIS prevents the disadvantage and shows better performance. The construction process by IBIS methodology is consistent with the engineering design process. Toulmin structure begins with a claim, can also be called a statement that already answered the design question, which is the conclusion of the design process. In this camera case study the design space need to be captured by more individual diagrams. However, both the IBIS and Toulmin successfully captured the essential elements of the design space, the author will demonstrate the differences in the sub-design case "Flash design" in the following.

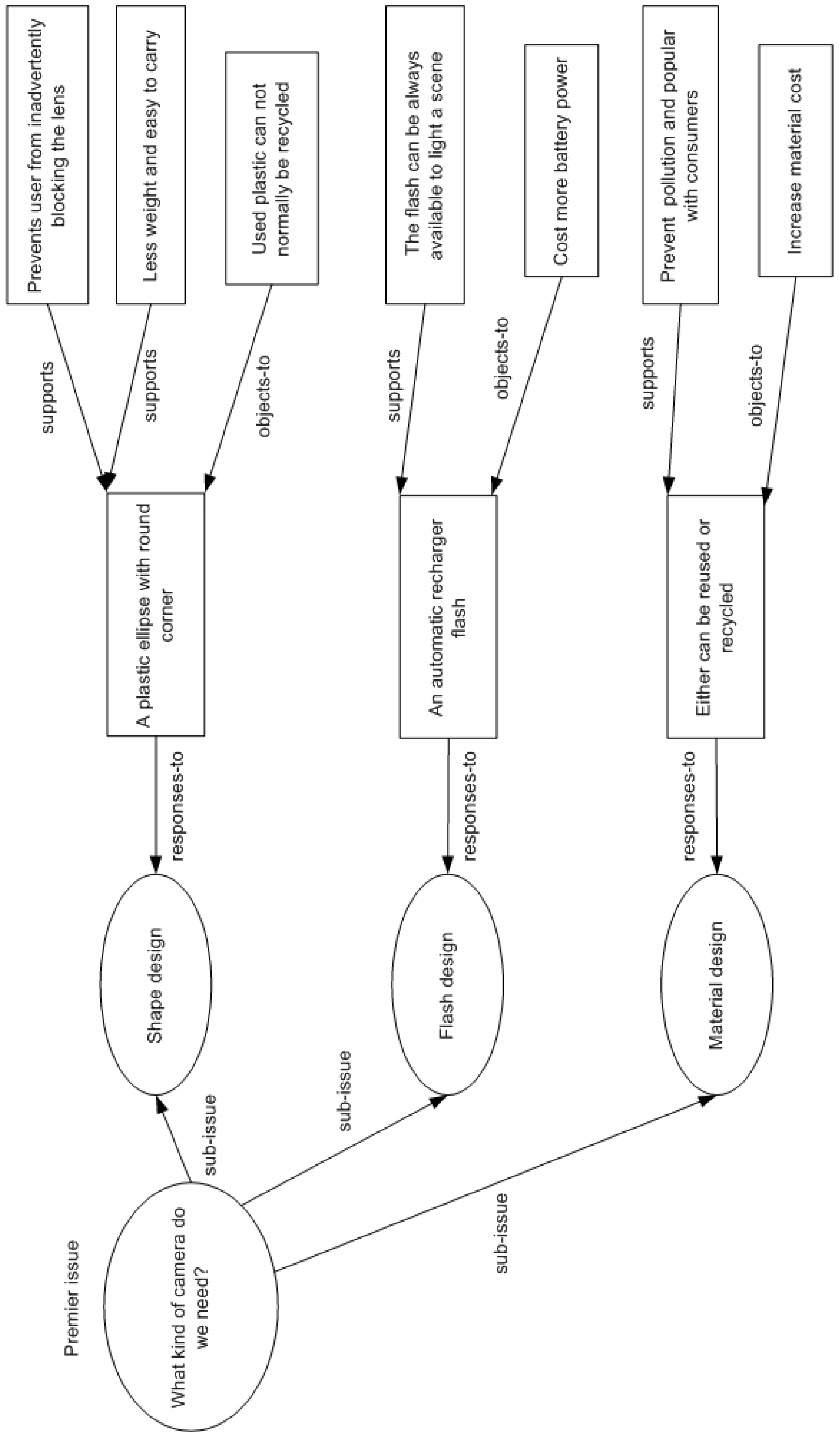

Premier issue
What kind of camera do we need?
sub-issue
sub-issue
sub-issue
Shape design
Flash design
Material design
responses-to
responses-to
responses-to
A plastic ellipse with round corner
An automatic recharger flash
Either can be reused or recycled
supports
supports
objects-to
supports
objects-to
supports
objects-to
Prevents user from inadvertently blocking the lens
Less weight and easy to carry
Used plastic can not normally be recycled
The flash can be always available to light a scene
Cost more battery power
Prevent pollution and popular with consumers
Increase material cost

Figure 4.5: The camera design representation by IBIS

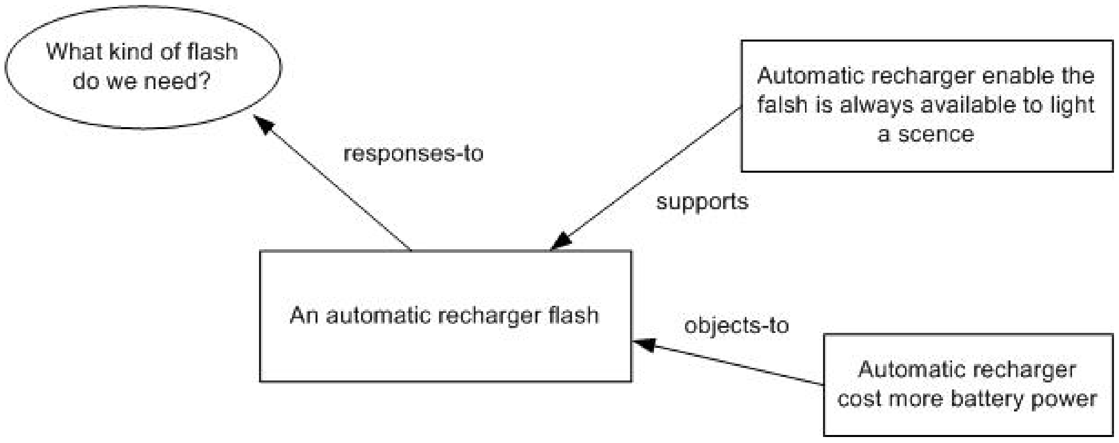

What kind of flash do we need?
responses-to
Automatic recharger enable the falsh is always available to light a scence
supports
An automatic recharger flash
objects-to
Automatic recharger cost more battery power

Figure 4.6: The flash design representation by the IBIS structure

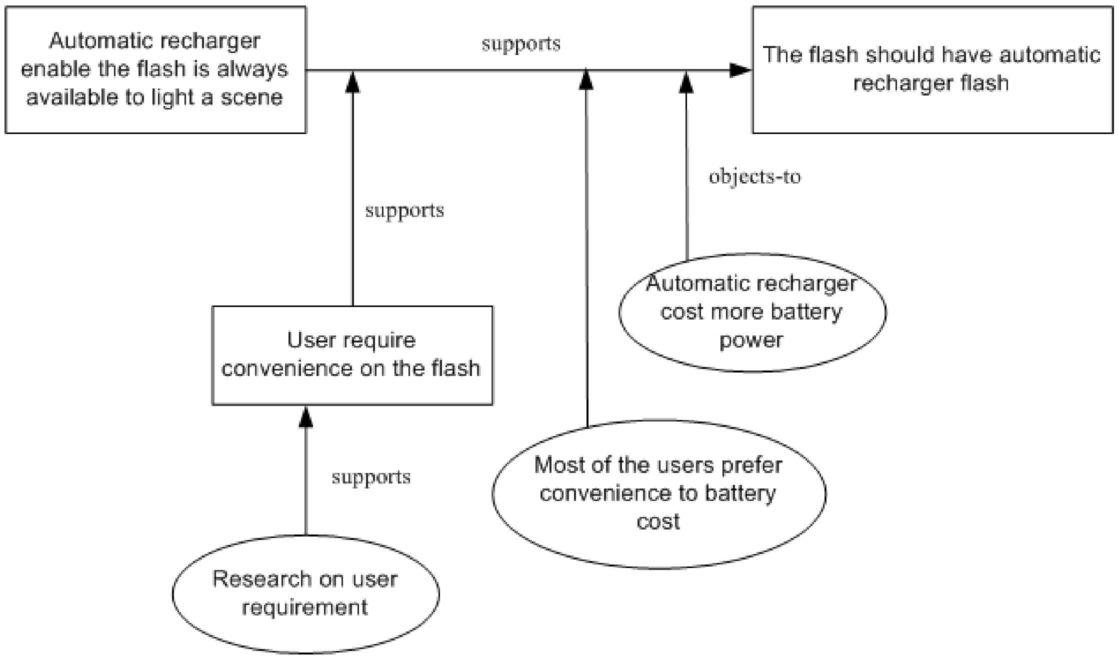

Automatic recharger enable the flash is always available to light a scene
supports
The flash should have automatic recharger flash
supports
objects-to
Automatic recharger cost more battery power
User require convenience on the flash
supports
Most of the users prefer convenience to battery cost
Research on user requirement

Figure 4.7: The flash design representation by the Toulmin structure

As metioned above, both the IBIS and Toulmin capture the essential element of the design space. There are differences between IBIS and Toulmin structure elements. The table 4.1 below show a clear understanding.

During capturing design rationale by using Toulmin structure, on occasion data and backing were difficult to distinguish. Some elements of the rationale can either be captured as data or warrant. Toulmin's approach can be considered too general, its template can easily confuse users, leading them to argue about the template roles and not the relevant design issues, and it causes design rationale objects types to context dependent (Reich, 2000). As perhaps would be expected, the Toulmin model did not provide for

Toulmin element	IBIS element
Claim	Agreed position
Data	Support argument
Warrant	Evidence for supporting argument
Backing	No equivalent
Reservation	Object to argument
Qualifier	Weight for support argument

Table 4.1: Equivalent between Toulmin and IBIS elements

delineation of components of the problem-solving process. Moreover Toulmin's approach is inappropriate for expressing many arguments related to design (Lee and Lai, 1991). One other side, the ability of IBIS or gIBIS to represent design rationale is limited (Lee and Lai, 1991), using IBIS, additional elements should be introduced, for example, evidence to support the argument, and evaluation, alternatives etc. for most of the design cases.

4.6 A decision-making model

Capturing or recording design rationale is a particularly difficult problem. Recording all decisions made, as well as those rejected, argument behind them, trade-off and alternatives can be time consuming and expensive. That can be only built effectively when a clear understanding of human reasoning has being developed.

Models for recording the decision-making have been published since 1970s (Kunz and Rittel, 1970). The author has reviewed these models in Chapter 3, and selected IBIS as the basis for this research. Notably lacking in IBIS is the notion of goal or objective against which the alternatives are being evaluated (Lee, 1990). While representing issues, positions, and arguments, gIBIS (graphical IBIS) fails to support representation of goals (requirements) and outcomes (Sigman and Liu, 2003). In gIBIS, objectives are only implicitly represented. The main contribution of DRL comparing IBIS is the requirement of design is explicitly addressed. Lee also identified many good reasons for explicit representation of goals. It makes people articulate and become aware of the objectives against which alternatives are being evaluated. The explicit representation also allows people to argue about these objectives and change them if desirable.

Even the agreed positions possibly relate to arguments which are objected to. Nomaguchi et al. (2004) introduces a new element which is called emphasis extending IBIS. Emphasis is a node that represents importance of the argument. Before the designer makes the decison that response to the issue, the solution need to be evaluated, compare with other alternatives and also constrained by criteria. Ahmed and Wallace (2001) investigates design strategies need to be employed when possible solution infer to decision during evaluation. While design strategy is too implicit, the author introduces the concept of criteria to replace design strategy, relating solution, evaluation and decision.

Figure 4.8 introduces a description model that best represents the decision making process.

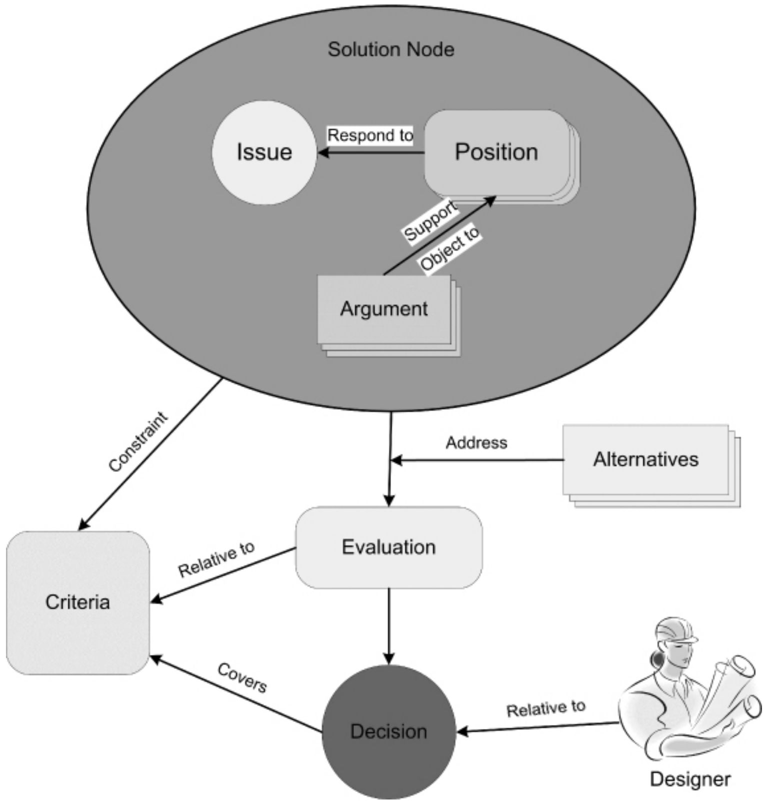


Figure 4.8: The decision making model, developed from the IBIS approach

The solution node, which is issue-based, comprise the generic elements of reasoning logic. An issue is an identified problem to be solved by deliberation. The key element of a design task is to solve the prime issue, which is first identified. The issue is normally decomposed into sub-issue if the issue is too generic or complicate to be solved by a simple solution. Position is the proposed solution to solve the problem. Each issue can have any number of positions. Each issue, position, and argument node should have the following attributes:

- Text description: a description that explains the contents of the issue, position or argument
- artifact information: information in a product model that is related to this node

The evidence for either supporting or opposing the position is represented as argument. Alternatives indicate other options for the problem, typically taken from previous design, or other case study. While evaluation makes explicit the evaluation measure and experimental data, analytical results and rank the alternatives for future design change. As usual, after fully evaluation, the selected decision should be recorded explicitly and there is a designer who responds to the decision.

An ontology based system on this model will be closer to knowledge base end than gIBIS. gIBIS is mainly a hypertext system whose services focus on the presentation of structure with the purpose of making it easy for people to see the structure of what is represented. Although gIBIS has the semantic types, these types are mainly for: enabling people to see clearly the structure of the represented knowledge, navigate through semantic links, and enter additional pieces of knowledge at appropriate places (Lee, 1990). Comparing gIBIS, the author views SW-DRS mainly as a knowledge-based system which uses ontology based representation so as not to force people to formalize the knowledge arised during design process. SW-DRS is more expressive and usable by providing computational services, whereas the other IBIS system is motivated mainly by ease of use.

4.7 System layer and architecture

To support design and other creative work processes, highly flexible structures are needed, which allow the integration of complex of information and specific domain. This flexibility has been achieved by using a service-oriented architecture. This is a new idea for building software system, in this architecture, an application's business logic or individual functions are modularized as services for client applications. These services are loosely coupled, which means the service interface is independent of the implementation. Therefore, there is no need for developing all the components to get the system work and new functions can be integrated easily. When the similar situations and features present, the system will easily reuse the functionality of existing systems rather than building them again. The designers's work has been enhanced by building the system on top of loosely connected ontology modules which is identified by Semantic Web Portal. Figure 4.9 shows a simplified view of the system architecture.

One of the key component is representation schema, which is argument based as discussed in previous section. The design knowledge is generated during design, mainly recorded by design document and drawings. These information are unstructured and provide non-semantic. That raw data can be proceed into the Design Knowledge Processing Engine and generate the structured rationale. The design logic is explicitly represented as objects and their relations as indicated in the following section. Design rationale is stored in MySQL implementing knowledge database, which make the information semantic, so that the retrieval and reuse of this knowledge can be possible. Another important element in the system is design ontology. People often produce different terms and concepts to describe the same facts. Thus it sometimes makes puzzle problems and conflicts and difficult to process the knowledge and extract them from knowledge base. The concrete ontology definition provides a solution. This system also extends its usability by integrate with other product modelling software such as CAD system.

4.8 Proposal for the SW-DRS system

In a Service-Oriented Architecture (SOA) environment, functional components expose services to support accessible to other applications via loosely coupled standard-based interfaces. These components, described Web Service, enable a uniform interface to heterogeneous systems to search, bind and invoke among them. Applications based on different programming languages, object models and platforms have a more efficient way to interact with each other applications. Besides, applications are independent with each other and there is no centralized control on them. As the concept of service oriented architecture, the main functions are divided into internal Web Services. These sub-services are Web Services with no public interface, yet have all the advantages of Web Services.

4.8.1 Web portal

The web portal integrate all the available web services and present an unified view to the potential web services consumers. It contains a simplified description of the function, location or providers of the service, moreover invoke and render methods are also specified in the portal. For example, Web Service Description Language (WSDL), Universal Description, Definition, and Integration (UDDI), and Simple Object Access Protocol (SOAP) are the fundamental pieces of the SOA infrastructure. WSDL is used to describe the service; UDDI, to register and look up the services; and SOAP, as a transport layer to send messages between service consumer and service provider. While SOAP is the default mechanism for Web services, alternative technologies accomplish other types of bindings for a service. A consumer can search for a service in the UDDI registry, get the WSDL for the service that has the description, and invoke the service using SOAP.

4.8.2 Representation schema

The main concepts during the design process discussed above, are issues, positions and arguments, which are presented as classes in ontology and instances in the graphical windows in Figure 4.10. All product information including individual components are organized as folders in lefthand column. In the righthand column, the system explicitly displays either product information or design rationale as a tree structure. That definition structures can be manipulated by a template definition editor. As show in Figure

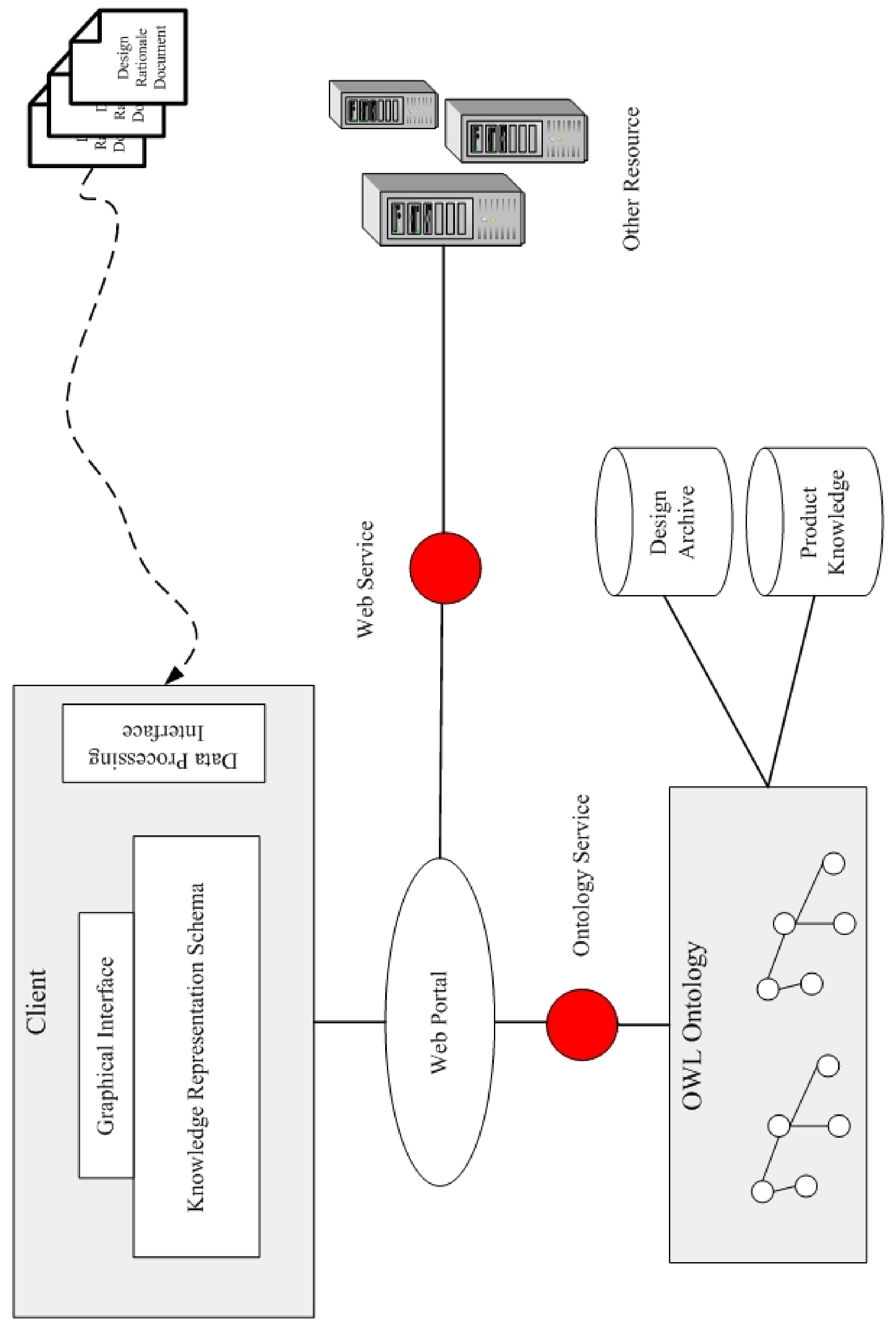


FIGURE 4.9: The proposed System Architecture

4.10, this system is project based, all the requirement, variables, drawing and process knowledge are manipulated in a design folder and virtually hierarchize according to their logic, such as decision tree or decision map. The design rationale for a specific engineering product can be enhanced by referential held in other system across an organization. These other system can include data warehouse, product data management and CAD systems. In the case of the CAD system attributes related to the product can be stored as metadata attached to a design or drawing file. This is currently not shown in the proposed representation, but could be addressed by the addition of more branches in the project manager panel (to the left column of Figure 4.10).

4.8.3 First step from informal documents to formal documents

In engineering design, large amounts of data are accessible, however they vary in formats and are stored at various organizations leading to difficulties of data discovery, data interoperability and usability. An mechanism needed to incorporate information about data identification, data presentation form, data content and data distribution regarding the dataset. The successful element of a design rationale system is to minimize the effort designers required to build the rationale, not forcing them to formalize the unstructured data, argue with the rationale template. An easier mechanism for transforming informal data to formal data is required.

For decades, people use metadata to exchange and share information, however, different organizations normally structure their information based on diverse types of metadata. To overcome this limitation, researchers have formalize the XML syntax and standard. XML allows to invent tags, and for the tags to contain both text data and other tags. Also, XML has a built-in distinction between element types. Figure 4.11 shows the initial stage of the conversion from unstructured data to metadata. In the CORE (2006) project, the objective was to provide the learning materials display online (HTML format) so it facilitates surgeons. Since the system is XML based and most existing document are either paper or electronic based, a service to transform Word document to XML document was a major issue in the project. Within the CORE project the author selected a open code program called WVConverter and encapsulated it into a windows service to do this work. See Figure 4.11 for transforming template based word document to XML format.

Although XML is unequaled as an exchange format on the Web, the unconventional data structures that mix trees, graphs, and character strings represented by XML documents are hard to handle in even moderate amounts, let alone by the billion. Moreover, the order in which elements appear in an XML document is significant and often very meaningful. This seems highly unnatural in the metadata world. The semantic web was introduced to take the advantage of the previous metadata. If assuming all the design documents are structured based on the semantic web technical standard, an automatic

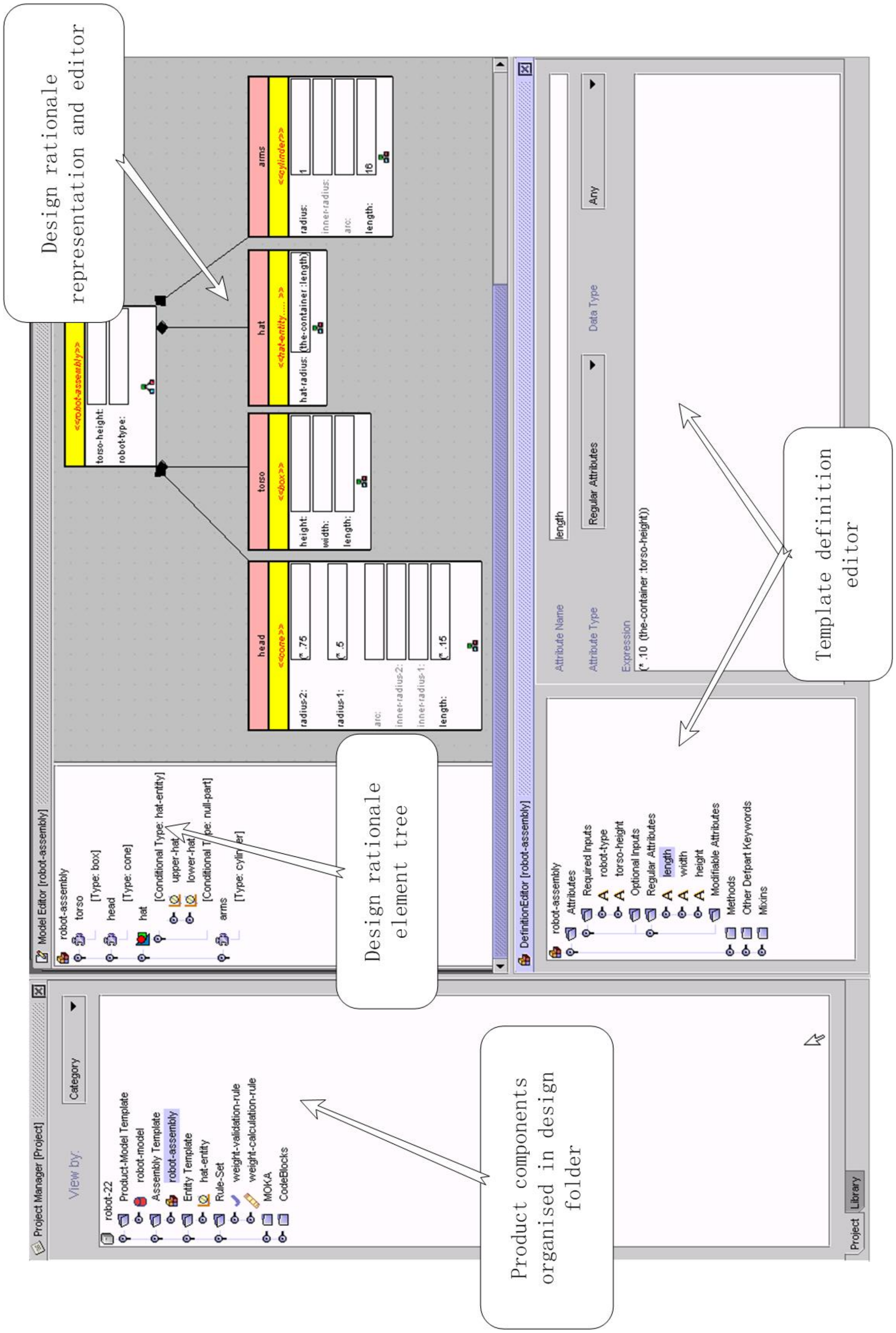
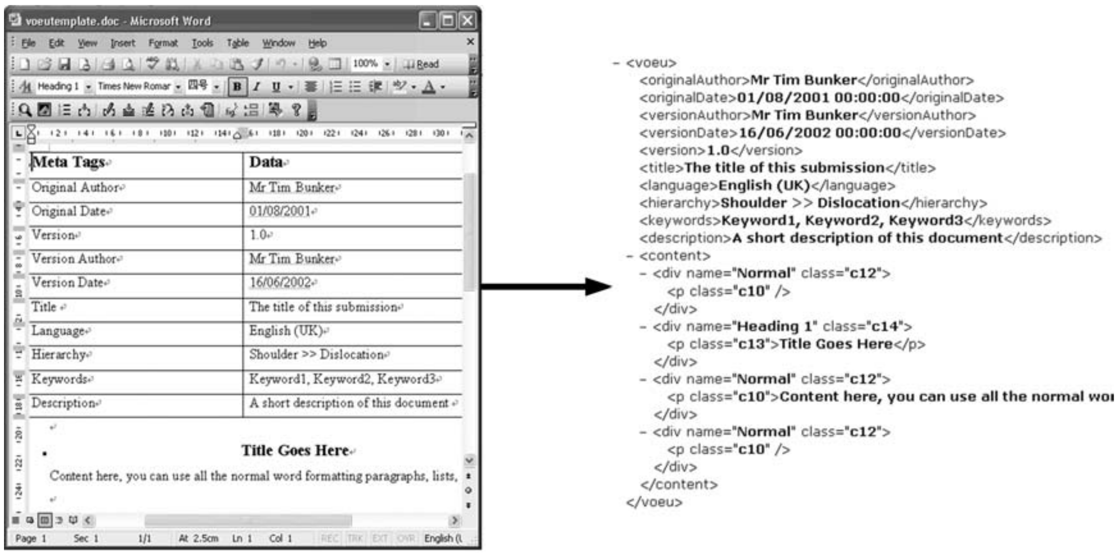


FIGURE 4.10: The proposed explicit representation of design rationale in SW-DRS


voeutemplate.doc - Microsoft Word

Meta Tags	Data
Original Author	Mr Tim Bunker
Original Date	01/08/2001
Version	1.0
Version Author	Mr Tim Bunker
Version Date	16/06/2002
Title	The title of this submission
Language	English (UK)
Hierarchy	Shoulder >> Dislocation
Keywords	Keyword1, Keyword2, Keyword3
Description	A short description of this document

Title Goes Here

Content here, you can use all the normal word formatting paragraphs, lists,

```
- <voeu>
    <originalAuthor>Mr Tim Bunker</originalAuthor>
    <originalDate>01/08/2001 00:00:00</originalDate>
    <versionAuthor>Mr Tim Bunker</versionAuthor>
    <versionDate>16/06/2002 00:00:00</versionDate>
    <version>1.0</version>
    <title>The title of this submission</title>
    <language>English (UK)</language>
    <hierarchy>Shoulder >> Dislocation</hierarchy>
    <keywords>Keyword1, Keyword2, Keyword3</keywords>
    <description>A short description of this document</description>
  - <content>
    - <div name="Normal" class="c12">
        <p class="c10" />
      </div>
    - <div name="Heading 1" class="c14">
        <p class="c13">Title Goes Here</p>
      </div>
    - <div name="Normal" class="c12">
        <p class="c10">Content here, you can use all the normal wor
      </div>
    - <div name="Normal" class="c12">
        <p class="c10" />
      </div>
    </content>
  </voeu>
```

Figure 4.11: Word to XML converter (CORE, 2006)

or semi-automatic approach to capture rationale will be available by using XML(s), RDF(s) and ontology.

4.8.4 Basic ontology

Ontology enable the seamless exchange of information between different parties. Since the participants in engineering design have slightly different views on the design rationale, significant effort is required to produce the resulting ontologies, which is discussed in next following chapter. The author details two types of ontologies as the system prerequisite below.

- **Argument ontology** For the reasons of interoperability, the Ontology Web Language (OWL) standard from the Semantic Web approach would be the first choice for building ontologies. This ontology is formalized in OWL based on IBIS argumentation model. In this ontology, issues, positions and arguments are represented as classes, subclasses and their relations as well. The details of argument ontology are introduced in following chapter.

- **Domain ontology** The author intends to investigate our methodology on an industrial case for evaluation, which is a robot chassis design. It is necessary to develop an robot ontology to describe all the attributes, variables of subcomponents and their dependency. As more semantical structure provided, the product knowledge can be easily access during design process.

4.8.5 Knowledge Base

This component store the ontology in RDF triples, where a triple consists of a subject, a predicate, and an object. The subject identifies what object the triple is describing. The predicate defines the piece of data in the object which will be given a value to. The object is the actual value. Unlike conventional tools that store their data in fixed schemes, knowledge base use ontologies with explicitly defined semantics to structure the data domain. Triple stores are the database of the Semantic Web world, designed to hold massive numbers of RDF triples in such a manner that the information they encode can be simply retrieved. While this is sufficient for most applications, common relational database solutions can effectively scale to hundreds of gigabytes of data.

4.8.6 System interface

Figure 4.12 illustrate the interfaces for SW-DRS. The functionality of the system is made available to computational service through the interfaces of the individual components,as discussed below.

Web browser interface will enable users to use the application over Internet through any kinds of Internet browser.

Graphic representation viewer/editor Modeling the elements into graphic nodes and links, which provides an intuitionistic understanding.

Interface for processing electronic data (email, document et al.) It is important to transform the electronic data into a more structured format, which facilitates valuable semantics.

Domain concept interface The designer will formalize their thinking, reasoning into design logic tree and the formalized information will be proceeded into ontology though domain concept interface.

Ontology data interface enable upper applications to semantic process stored ontology information and provide computational service.

Link interface This interface is implemented by Sesame 2.0 API, the functionality is to store RDF triples into data base.

Web service interface will use the available engineering service online, which makes SW-DRS more flexible and powerful.

Product architecture model interface This system provides a interface for designers to browse the product data associate with the argumentation.

Integration with third party software The system can enhance its performance by integrating with CAD, office or web applications.

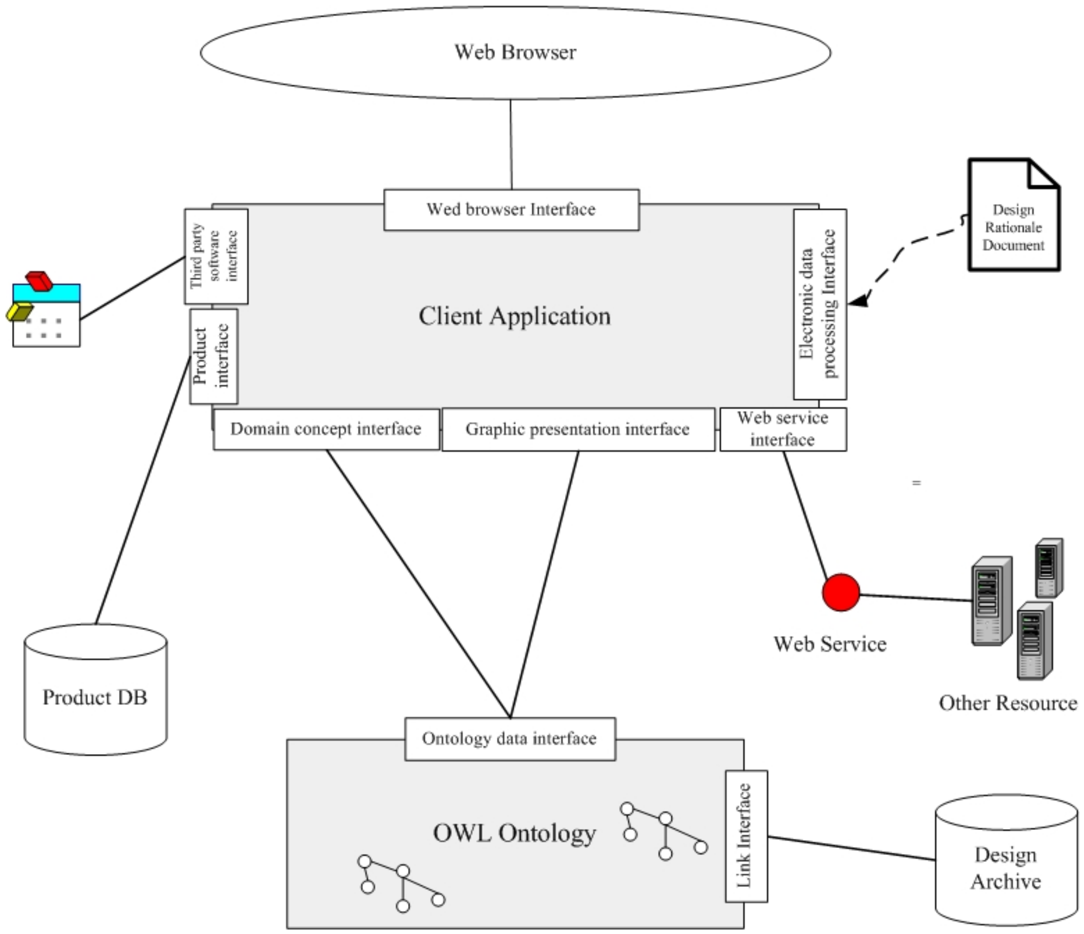


FIGURE 4.12: Interfaces for the proposed SW-DRS

4.9 Summary

As discussed in this and previous chapters, a specific design rationale consists of a large number of documents and related material. It is further complied with the rationale that contains both technical and socio-technical elements. Ontologies provides the mechanism for sharing and integration of this kind of knowledge. It is now widely agreed that ontologies are a core enabler for Semantic Web vision. In this chapter, the author proposes an integrated formal argumentation model to be used in SW-DRS framework, allowing the capture of the engineering design process. The decision making model supports the process in several ways. In discussions, it focus the designers and help to structure their arguments. In the sharing and reuse phase, the rationale captured can be consulted to better understanding the current or previous design task process. It is expected to be implemented as a service oriented architecture, thus introduce several benefits, which are flexible data structures and work processes for the support of creative and design processes, improved retrieval mechanisms based on use of Semantic Web technologies, and integrated representation of resource, data analysis and knowledge retrieval. Several other the required supporting research topics, such as knowledge database requires further effort, however the author intent to integrate this system with other research findings.

Chapter 5

Semantic Web and the SW-DRS System

During the engineering design process, significant amount of decision-making used to be archived manually, because of the number of people, companies and systems involved. Ontology based technologies provide the ability for modelling specific domain knowledge. Thus computer based systems can gain further understanding accordingly, to some extent gathering and understanding knowledge and making decisions. These efforts requiring fair amount of comprehension and complex decision-making, can possibly reach a higher automation level, once the right model knowledge is well prepared and highly accessible.

5.1 Designing ontologies

As discussed in Chapter 4, recording the decision making process can be time consuming and expensive. That information can only be built effectively when a clear understanding of human reasoning being developed, ontologies provide the opportunity. The construction of the ontology itself involves two process: formalization of the definitions and "consensualization" on these definitions The formalization of definition usually means transforming a definition from a natural language into a formal one using the specification defined by World Wide Web Consortium (W3C, 2007). This is not an easy task, normally it goes through a few phases, first summarize a textual definition from the information collected in documents or shared understanding in the community, then write a first formal definition, using only basis constructs of the language, then add more subtle elements.

5.2 Ontologies for engineering design

Over the past few years, many ontology and knowledge based development tools have been developed. The ontology building and editing tool in our research should meet some common requirements. The ontology must be tightly integrated with information sources specified as engineering design knowledge. In fact, when build ontology, it constantly refer to data sources or documents to validate the conceptualization. Hence, the engineering environment should provide quick and easy access to reference material. One of the first and also still the best known is from Stanford University. Protégé lets users construct a domain ontology and enter domain knowledge. The ontology should also provide interface to the conceptualized knowledge of the domain. A single interface can not be the best in all situations. It must be possible to customize the tool's interface according the specificities of the domain and to the reference data and documents. This work should not go too further into the domain, which is very time consuming but simple enough for usage. Protégé is in fact extensible in this case.

5.2.1 An Argumentation Ontology

The ontology should reflect a consensual view of the domain. However, ontology building is usually a collaborative task that involves domain experts and ontologists's effort. In addition, in an organization, there are many different kind of individuals, each has his personal view of the domain according to his background and function. This argumentation ontology should try to conciliate those different viewpoints in order to make it usable and effective. To support collaborate and argumentation work on ontologies, the author have defined our model in Chapter 4 added IBIS-like components. The argumentation ontology is visualized in Figure 5.1. The various concepts and their relation are represented by different symbols: ▭ denotes *classes*, → denotes *has, response* or *operate* relation and ⇾ denotes subclass *inherit* from parent class.

The main concepts in our ontology are **issues**, **positions** and **arguments**, which are represented as classes. They are ontology implementations proposed by IBIS methodology. Issues introduce new topics in the discussion from a conceptual point of view. They are usually goals, requirements or questions from a project which need to be issued or answered. These are actually formalized and implemented in ontology rather than conceptual models. Positions are related to issues in the sense that they respond to them. They are possible solutions to the questions or goals. An issue instance can relate to one or several position instances. In addition, addressed positions are also restricted by rules, which include the subclasses, requirement and constraint. Positions are only taken into account when they meet the rules. Arguments are *ON* either on particular position or one particular issue.

Typically, our lead engineering designers with more experience will start by proposing

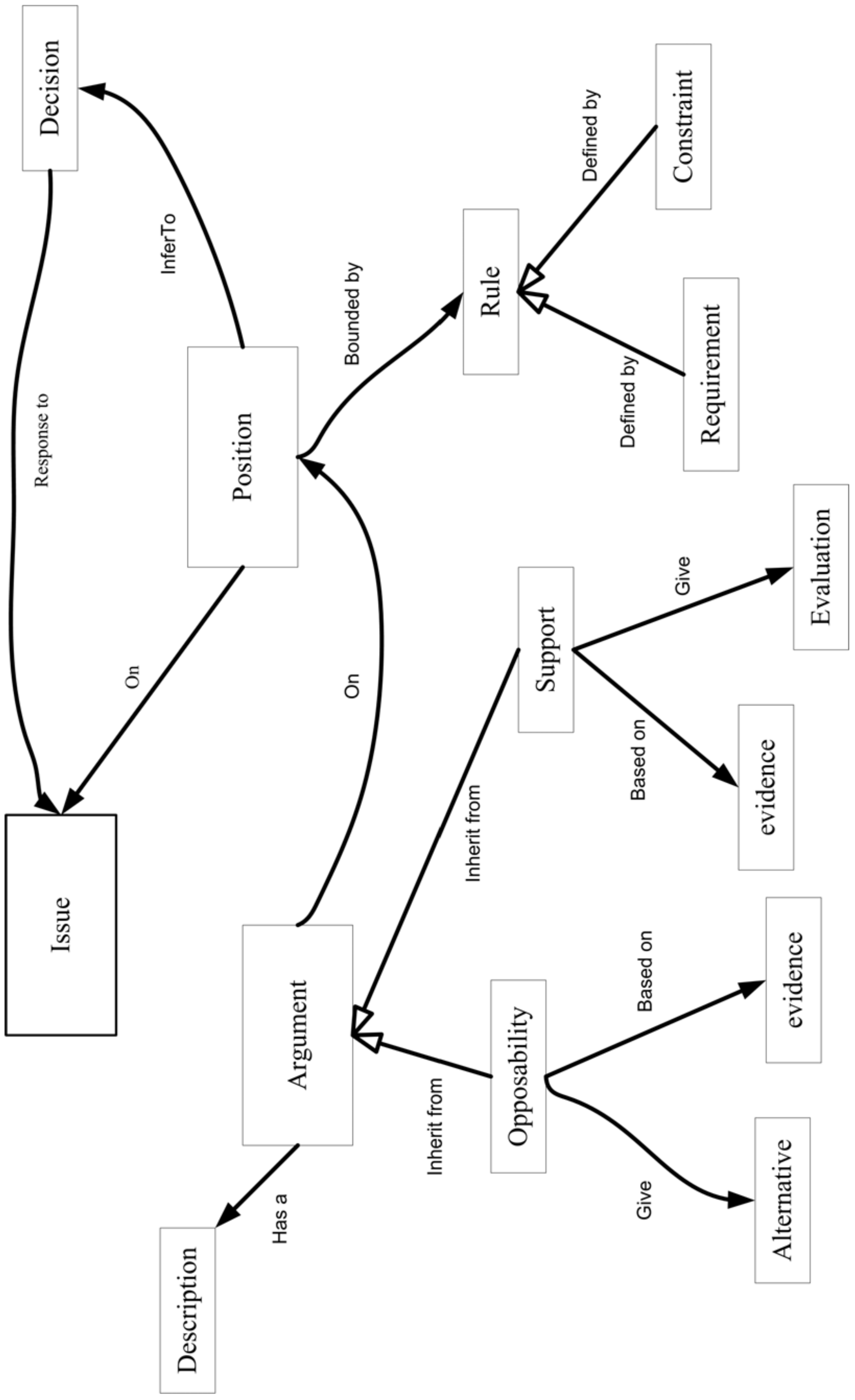


FIGURE 5.1: Major classes of the argumentation ontology for SW-DRS

new issues to be introduced in ontology, during which they discuss how issues should be formalized through conceptual ideas. Then detail design experts with domain knowledge will provide positions *ON* issues. Arguments will be exchanged over designers. Arguments for (pro) an issue are called **support**. Arguments against (con) an issue are called **opposability**. In what regards arguments in favour, the author identified **evidence** related to either support or opposability. In case of support, an support instance possible provide a evaluation. While in case of opposability, an alternative should be indicated if there is one.

Positions are stated in discussion. They clarify the *ON* an issue or an argument under discussion. Once enough arguments have be provided and the system will automatic infer to a **decision** from a position. In other word, positions lead to decisions. Decisions are response to issues. A decision instance has a status that can be vary from **postponed** (not enough arguments), **discarded** and **agreed**. Either agreed or discarder is decided by the weight methodology on the basis of arguments either agrees or disagrees.

All issues, positions and arguments are related to on particular or some designer by *givenBy* function. Designer are normally human engineer and they have the potential to be extended to Agents. In that case, fully or partial automatic design will take place.

The ontology is implemented in Protégé 3.2.1, a screen shoot is shown in Figure 5.2.

5.3 Ontology and services semantic modelling

Nowadays, if potential applications intend to benefit from web services, they can only perform lookup among millions of them via a UDDI server. To achieve a higher-level automation, the author introduced an ontology-based description framework to describe web services in Chapter 4. This service oriented architecture aims at partial automation by utilizing the semantic web portal and semantic registry of web services instead of web-distributed application. In this framework, developers can attach concept and properties of the shared ontology onto web services methods and parameters for registering their semantics by which can easily interoperate them with other ontological information.

5.3.1 Ontological web services

With the standard specified by W3C, the basic way to describe any web service, each operation has input and output message parts, which can be simply as a parameter, or complex as an ontology instance. Service is presented as the following relationship:

Operation (Input message) $\rightarrow$ Output message

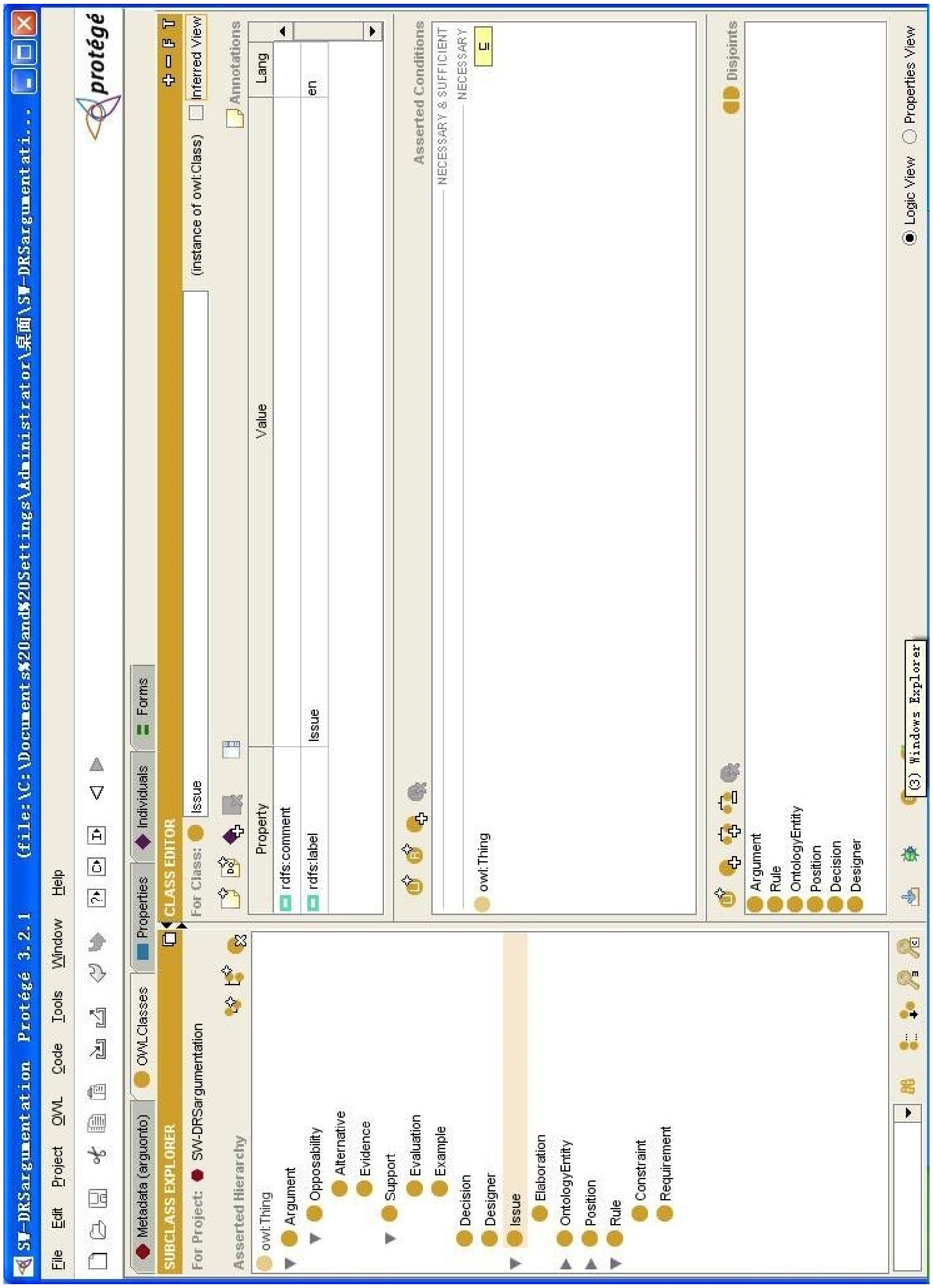


Figure 5.2: Implementation of the argumentation ontology shown in Figure 5.1 by Protege 3.2.1

The operation and together with concept and properties manipulated by the web services methods will be described by ontology (Operation Ontology). They can related to input and output parts with semantic association to achieve a high-level automation.

5.3.2 Semantic WSDL interface

To benefit the semantical ontology, it is necessary to develop an OWL-based web service ontology, as well as supporting tools and agent technology to enable automation of services on the Semantic Web. OWL-S, OWL-based Web Services Ontology, is this kind of tools, which supplies web service providers with a core set of markup language constructs for describing the properties and capabilities of their web services in unambiguous, computer-interpretable form. The objective of OWL-S is to facilitate the automation of web service tasks including automated web service discovery, execution, interoperation, composition and execution monitoring.

5.4 Knowledge Base

A knowledge base is a special kind of database for knowledge management. It provides the means for the computerized collection, organization, and retrieval of knowledge.

Knowledge bases can be categorized into two major types (Ullman, 1988):

- Machine-readable knowledge bases store knowledge in a computer-readable form, usually for the purpose of having automated deductive reasoning applied to them. They contain a set of data, often in the form of rules that describe the knowledge in a logically consistent manner. Logical operators such as And (conjunction), Or (disjunction), material implication and negation may be used to build it up from the atomic knowledge. Consequently classical deduction can be used to reason about the knowledge in the knowledge base.
- Human-readable knowledge bases are designed to allow people to retrieve and use the knowledge they contain, primarily for training purposes. They are commonly used to capture explicit knowledge of an organization, including troubleshooting, articles, white papers, user manuals and others. The primary benefit of such a knowledge base is to provide a means to discover solutions to problems that have known solutions which can be re-applied by others, less experienced in the problem area.

A knowledge base may use an ontology to specify its structure (entity types and relationships) and its classification scheme. An ontology, together with a set of instances of its classes constitutes a knowledge base.

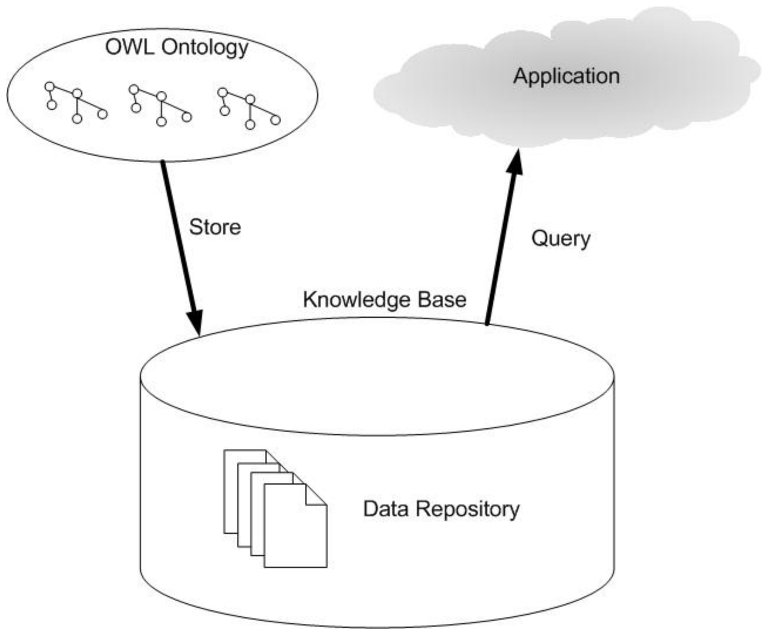


FIGURE 5.3: Store and query process using Knowledge Base

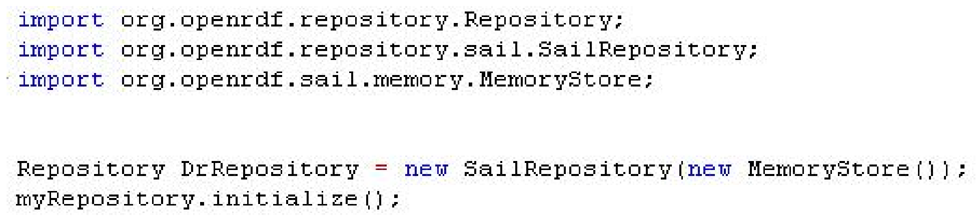
A triple store is designed to store and retrieve identities that are constructed from triplex collections of strings (sequences of characters). These triplex collections represent a *subject-predicate-object* relationship that more or less corresponds to the definition put forth by the RDF standard. The problem space of storing this sort of data has been explored by the graph database, object database, PROLOG language and, more recently, semantic web communities. A scalability report on existing RDF data stores has been published (Lee, 2004). In the report, Sesame is rated high in terms of its performance, ease of use, and deployment. Based on this report, the author has made the decision to use Sesame to implement the data triple stores. Sesame described as *"A Generic Architecture for Storing and Querying RDF and RDF"* was designed to use some typical storage systems as the underlying persistent store. The storage systems are various relational database management systems (RDBMS). Moreover Sesame have already implemented the interfaces for PostgreSQL, MySQL, and Oracle. The author use Sesame 2.0 and MySQL 5.0.3 for developing the triple store for the prototype.

Figure 5.3 shows the most important process implemented in SW-DRS, which are store and query functions. These are also the essential functions in any type of triple store.

5.4.1 Storage

The storage of RDF triples is by the concentration of the DBMS-specific code into a Storage And Inference Layer (SAIL) which interfaces between the RDF-specific methods and the database API. SAIL is a high-level and lightweight Java API that includes support for RDF schema semantics and data-streaming operation. SAIL can work on top of any RDBMS, ODBMS, existing RDF stores, RDF files or network services. The first step in any action that invokes Sesame repositories is to create a Repository for it. Repository objects operate on Sail object for storage and retrieval of RDF data.

- **Creating a RDF Repository.** As shown in Figure 5.4, the constructor of the SailRepository class accepts any object of type Sail, so simply pass it a new main-memory store object. Following this, the repository needs to be initialized to prepare the Sail that it operates on, which includes operations such as restoring previously stored data, setting up connections to a relational database, etc.



```
import org.openrdf.repository.Repository;
import org.openrdf.repository.sail.SailRepository;
import org.openrdf.sail.memory.MemoryStore;


Repository DrRepository = new SailRepository(new MemoryStore());
myRepository.initialize();
```

FIGURE 5.4: Creating the RDF Repository by Eclipse 3.2.

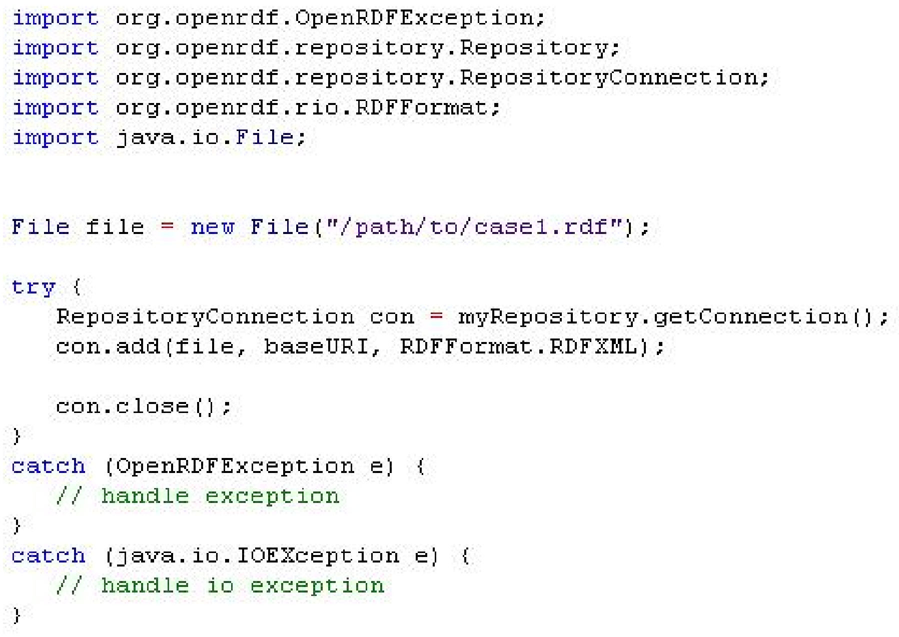
- **Adding RDF to the repository.** The Repository API offers various methods for adding data to a repository. Data can be added by specifying the location of a file that contains RDF data. Figure 5.5 illustrates adding RDF data form a RDF file to the repository. Application performs operations on a repository by requesting a *RepositoryConnection* from the repository. On this *RepositoryConnection* object application can the various operations, such as query, getting, adding, or removing statements, etc.

5.4.2 Query

The Repository API has a number of methods for creating and evaluating queries. Two types of queries are distinguished: tuple queries and graph queries. The query types differ in the type of results that they produce.

The result of a tuple query is a set of tuples (or variable bindings), where each tuple represents a solution of a query. This type of query is commonly used to get specific values (URIs, blank nodes, literals) from the stored RDF data.

The result of Graph queries is an RDF graph (or set of statements). This type of query is very useful for extracting sub-graphs from the stored RDF data, which can then be queried further, serialized to an RDF document, etc.



```
import org.openrdf.OpenRDFException;
import org.openrdf.repository.Repository;
import org.openrdf.repository.RepositoryConnection;
import org.openrdf.rio.RDFFormat;
import java.io.File;


File file = new File("/path/to/case1.rdf");

try {
   RepositoryConnection con = myRepository.getConnection();
   con.add(file, baseURI, RDFFormat.RDFXML);

   con.close();
}
catch (OpenRDFException e) {
   // handle exception
}
catch (java.io.IOException e) {
   // handle io exception
}
```

Figure 5.5: Adding RDF to the repository by Eclipse 3.2.

5.5 Summary

The integration of the basic ontology into a semantic web environment will enhance the performance of the system infrastructure. Web Services are standards that enable remote invocation among heterogeneous systems and hence a perfect solution to leverage diverse legacy resources as well. Ontology releases the possibility of machine readability and of precise understanding among computing entities. The author sees both the utility and possibility of an enhanced automation level through the perfect combination of Web Services and ontology by which systems can know the semantic of the resource and integrate necessary services. Through modelling the semantic relations among Web Services interfaces, the system connects the missing semantic link to automate task execution accomplished, in a way.

In this chapter, the author have presented an argumentation ontology to be used in design discussion, in particular in engineering design process. This ontology supports the process in several ways. In discussions, it focuses the designers and helps to structure their arguments. In usage and analysis, the ontology will help better understanding the current version of the design issues. Since the ontology covers all aspects of the discussion activities, namely issue raising, formalization of the issues and decision making, the designers are always informed about the current status of the discussion and the model they are building. The main contribution of this research is a formal argumentation model, which is an adaptation of the IBIS argumentation model specifically for engineering design process. Although at present it is difficult to express the usefulness of the designed ontology, in next chapter the author will investigate on a engineering case study using the designed ontology in the infrastructure to evaluate the success of the ontology as well as the discussion about system framework with related work.

Chapter 6

Evaluation

The evaluation of Semantic Web Design rationale System (SW-DRS), discussed in this chapter, was conducted experimentally rather then analytically. This chapter will provide the foundation for further system evaluation. The reason for this is that, in order to evaluate a system analytically, require providing that the system is complete, at present, the SW-DRS is partly implemented. And a formal specification of the design rationale problem that the system is attempting to solve is required, although the goal is to help designer a better and effective design, it is rather implicit.

An small case study and expert reviews are discussed in this chapter permitting an initial evaluation the SW-DRS.

6.1 Evaluation approach

After the system infrastructure is established, the next stage of work includes the evaluation of the constructed system infrastructure. Generally, there are three methods of estimating the performance of a system, a comparison evaluation, a simulation evaluation and a user evaluation (Wills, 2000). In this work, the author compared SW-DRS with another design rationale capture system, DRed.

Design is an iterative process of prototyping, testing, analysing, refining and evaluating a work. Most critical decision-making in a design process happens during or after the evaluation phase of each iteration, before moving on to the next iteration. In this respect, it is crucial to consider the evaluative aspect of a design process when one tries to capture design rationale. Using simulation method to evaluate the performance of the system is based on the model of the real system built with software programming tools or system simulation tools. It often provides a rough performance estimation for the system infrastructure to be constructed. For modeling the evaluative aspect of a design process, it is necessary to draw upon the concept of system design evaluation in systems engineering as goal, question, usability, expert review and produce evaluation matrix.

The author will adopt the results based on argumentation model to develop the program logic and methodology for the evaluation. The implication of this approach is to focus on achieving and measuring results for the priorities. It will establish a logic for selecting results and indicators from data sources (design reports) for the evaluation analysis.

In a user evaluation, the system or service is tested by real users carrying out real tasks in realistic conditions. A user test will identify aspects of a design that cause users difficulty, confusion, or misunderstandings. These may lead to errors, delays, or in extreme cases inability to complete the tasks for which the product or service is designed. A typical user evaluation will answer the following questions: Do they understand it? Can they understand how to operate it? Can they use it successfully to complete the tasks it is designed for? User evaluation will benefits in several ways. It verifies that real users can use it successfully, or identifying what prevents them from doing so. It also helps designers understand users and see things from their perspective, so that they are more likely to design something that works for users first time.

However user evaluation is very time consuming for searching relevant users, structure the testing report. A difficulty for user evaluation is to make the evidence objective and realistic, not based on personal opinion or speculation. The author will identify detail elements for evaluation in our future work.

The SW-DRS evaluation includes two core components:

- **results evaluation**, which will examine whether the program is meeting the needs and improving the outcomes for engineering designers.
- **process evaluation**, which will review the efficiency and effectiveness of the implementation of the ontology based SW-DRS.

In combination, the two different approaches will address the designers' requirement during engineering design process.

6.2 What to evaluate

The first step of evaluation is to identify what element should be evaluated or what set of questions be answered and how answered. Shum and Hammond (1994) made two evaluation claims : "Argumentation-based design rationale is useful" (utility) and "Argumentation-based design rationale is usable" (usability). In this chapter, the author will evaluate the aspects listed below:

- **Design rationale usability,** which means, Does it succeed in capturing the rationales looked for by designers? In order to evaluate the usability of SW-DRS, it is necessary to compare the designers' requirements with the content of these rationale captured. Indeed, it will be possible to claim that design rationale is usable only if designers' requirements can be met by such a system.

  To address this issue, the author account for the design questions expressed by the designers that are answered by the system. The author considers that a designer's question is answered by the design rationale either if the an answer in outputs, or if the system provides designers with a relevant answer which existed in the rationale.

- **Effective design.** Evaluation can also been implemented by comparison, which the author compare SW-DRS with paper based design report and other design rationale system. First it is necessary to define a number of evaluation element, which contains the goal, usability, time consuming and etc. Next phase the author runs the realistic tasks on the system identified for comparison, produce a comparison table and give the conclusion.

6.3 Metrics for engineering design

The acceptability of systems depends on various aspects, one of the most important of which is usefulness. The usefulness of a system includes both its utility and usability (Grudin, 1992; Nielsen, 1993) defines the usability of a software system based on five sub-characteristics where the system: (a) should be easy to learn so that the user can rapidly start work using the system; (b) should be efficient to use, so that once the user has learned the system, a high level of productivity is possible; (c) should be easy to remember, so that the casual user is able to return to the system after a period of not having used it, without having to learn everything all over again; (d) have a low error rate, so that users make few errors during the use of the system, and if they do make errors they can easily recover from them; also, catastrophic errors must not occur; (e) should be pleasant to use, so that users are subjectively satisfied when using it; they like it.

The author set out by discussing seven questions in relation to SW-DRS Table 6.1. Questions 1-3 are derived from Shum (1996), assessing the utility of the system. Questions 4-7 are taken directly from the usability sub-characteristic definitions above, evaluating the usability of the system.

In order to justify the usability and utility of a software system, it is necessary to numerical measure of the system, which are called software metrics in software development. Algorithm performance is a complex aspect that could be measured by various elements,

Evaluation questions
1. Does the system help or hinder the reasoning process?
2. Can we review the designers' reasoning process?
3. Is the representation rationale easy to understand?
4. Is the model easy to use by the engineering designers?
5. Is the system easy to learn?
6. Is the system easy to remember?
7. Is the system subjectively pleasing?

Table 6.1: Questions for evaluating the utility and usability of SW-DRS

A Goal-Question-Metrics (GQM) approach has proven to be particularly effective in selecting and implementing metrics (Fenton and Pfleeger, 1997). The GQM approach was first suggested by Basili et al. (1994).

In SW-DRS system evaluation, it is necessary to provide a three-stage framework using GQM approach, the three steps are

1. Develop a list of major goals for the evaluation, these were described in Section 6.2.
2. From each goal, derive a set of questions that must be answered if the goals are too implicit.
3. Identify what is to be measured in order to answer the questions.

The goals for evaluation in SW-DRS is defined as - *Improve the utility and usability of the design rationale system for engineering designers.* The questions have been defined in Table 6.1, and metrics for meeting the goals are inferred from the questions. Table 6.2 shows the full elements need to evaluated.

6.4 A case study

The SW-DRS evaluation has carried out using a student summer internship project (Cooper and Crowder, 2006). The goal of the project was to design and build a mobile robot on which robotics experiments could be carried out in the real world. The particular emphasis on use of the robot was for experiments involve in navigation, of many forms using a variety of different sensors. The author captured the rationale in

Purpose	Improve
Issue	Utility and usability
Object	Better and effective design
View point	Engineering designers

Questions	**Metrics**
Do we capture the required information for the design task?	Design rationale capture for a design task
Does the representation rationale ease understand?	Design rationale representation understanding (via the user interface)
Does the model ease use by the engineering designers?	Effort required to construct the design rationale
Does this system help or hinder the reasoning process?	Benefit in cost out and other savings to the designer and the organization
Can we review the designers' reasoning process?	Design rationale retrieval through design process
Computational overhead and complexity	Storage Output Reusability

TABLE 6.2: Table of Goal-Question-Metrics for evaluation

the process at the early design phase, which is designing the robot shape, particular robot chassis.

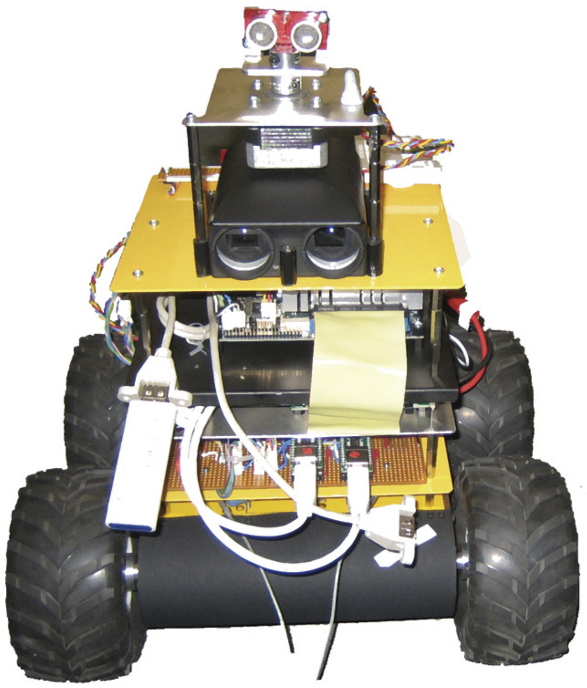


FIGURE 6.1: The robot built for navigation experiment

The requirement of the robot for this project is listed below:

1. The chassis has to be large enough to contain or have mounted on it all the required electronic circuitry and sensors. There should also be some extra space available when the robot is complete to allow for future expansion in the event of the robot's requirements changing as it is used in different projects.
2. The chassis has to be capable of supporting sufficient payload to account for all the circuitry, sensors, motors and batteries that are required for the operation of the robot.
3. The chassis has to be capable of traveling around the school on the floor surfaces that are there contained, primarily industrial carpet.
4. The robot has to be designed that, if required, replica robots can be built from design, description and other information included with the robot.

The author captures the design rationale for choosing the robot chassis, reasons for whether based on previous design or built from available design description and sketch though designers' log book, design reports and interview. A snap shot of the design rationale representation in SW-DRS is shown in Figure 6.2. The requirements on the bottom left of the figure are those given at the beginning of the section. The representation is based on the decision model that described in Chapter 4, which includes a series of design elements: issue, position, argument, constraint, requirement, evaluation and decision. The design begin with the primer issue: the main chassis for the robot, whether to base on existing design or to design, manufacture and build something from scratch. Following that the designer explores six options for solve this design question. Each option is whether support or disagreed by several arguments, in the figures $\longrightarrow$ denotes that reasoning process proceeding, $\boldsymbol{\rightarrow}$ denotes that the argument supports the option, on the other hand, and $\dashrightarrow$ denotes the the argument is object to the option. After fully consideration of the design space, the designer come out a solution which is the designer's option answer the design question. Then during other information such design criteria are considered, the solution is evaluated and finally leading to make the decision. As part of the process the design team evaluated six possible design options, ranging from self-constraint robot based through highly modified trials, to a simple design by University of Southampton. It was concluded that the best option is the *LynxMotion 4WD3* which is shown in Figure 6.1.

6.5 Experience with DRed

The author compares the evaluation result to a similar design rationale system (DRed) which has been fully implemented and employed in industrial applications. DRed has been developed and evaluated over a number of years. DRed was formerly introduced into aero engine company and is used for real world project. The author gained the following knowledge from previous reported work (Bracewell et al., 2004), and Aurisicchio et al. (2006)'s evaluation on DRed.

Within traditional design rationale capture tools, the complete text of the element can only be viewed or edited by invoking a pop up window. Every issue, solution and argument, the user needs to summarize it meaningfully into no more than five or six words. Unlike the existing known IBIS tools using single line labels to represent rationale elements, labels in DRed can be any width and number of lines. An typical screen shot is presented in Figure 6.3. Unlike most graph based tools, producing comprehensive hard copy is not a problem. The outline can be printed as a fully expanded hierarchically structured text document.

Moreover, the following features are identified:

- Simplicity, with a relatively low training requirements.
- No database management system required.

  Tunnel linked DRed files live in the project folder. In DRed terminology a tunnel link is equivalent to a hyperlink.

  Simple to deploy DRed and create a browsable and searchable archive of completed project folders.

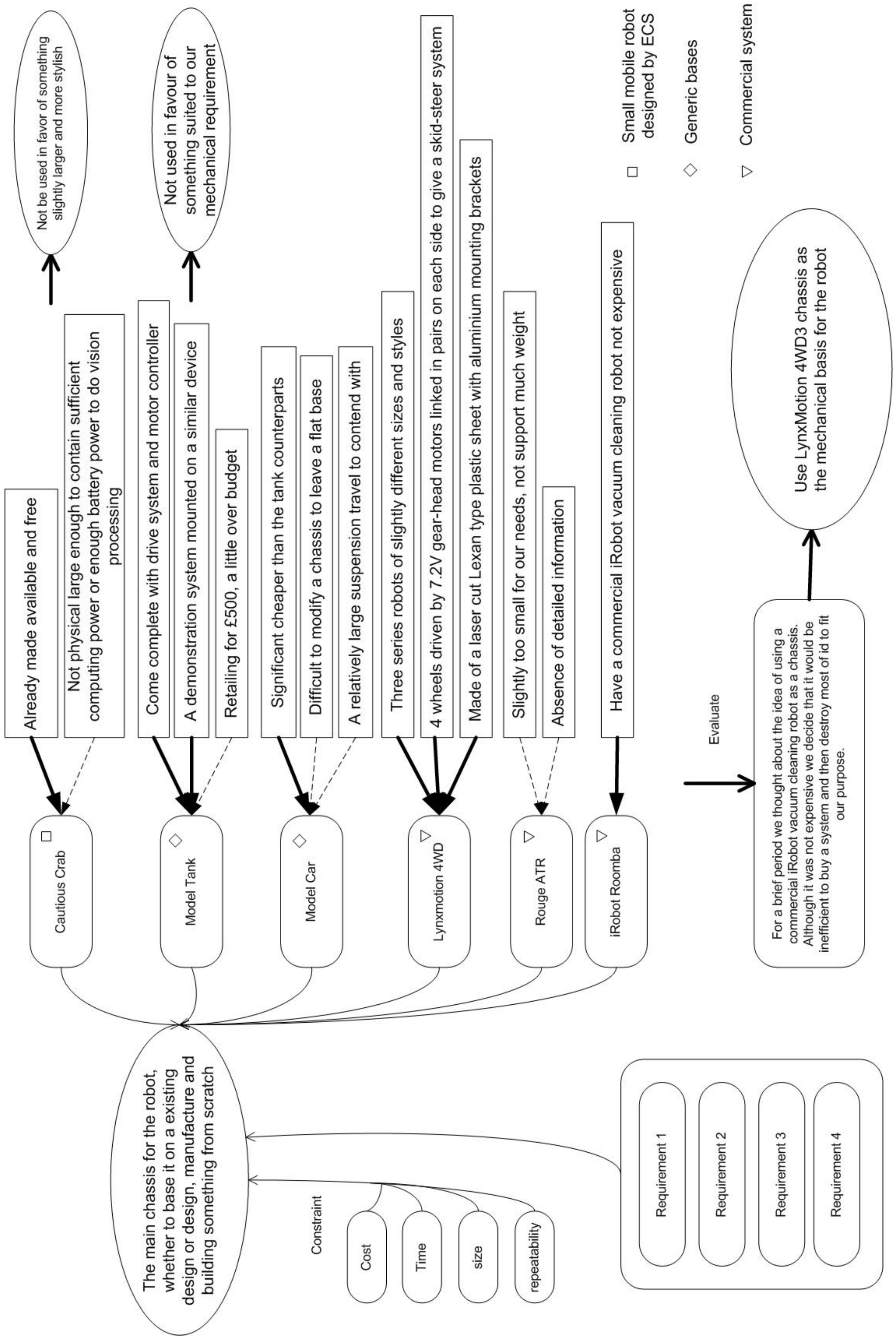
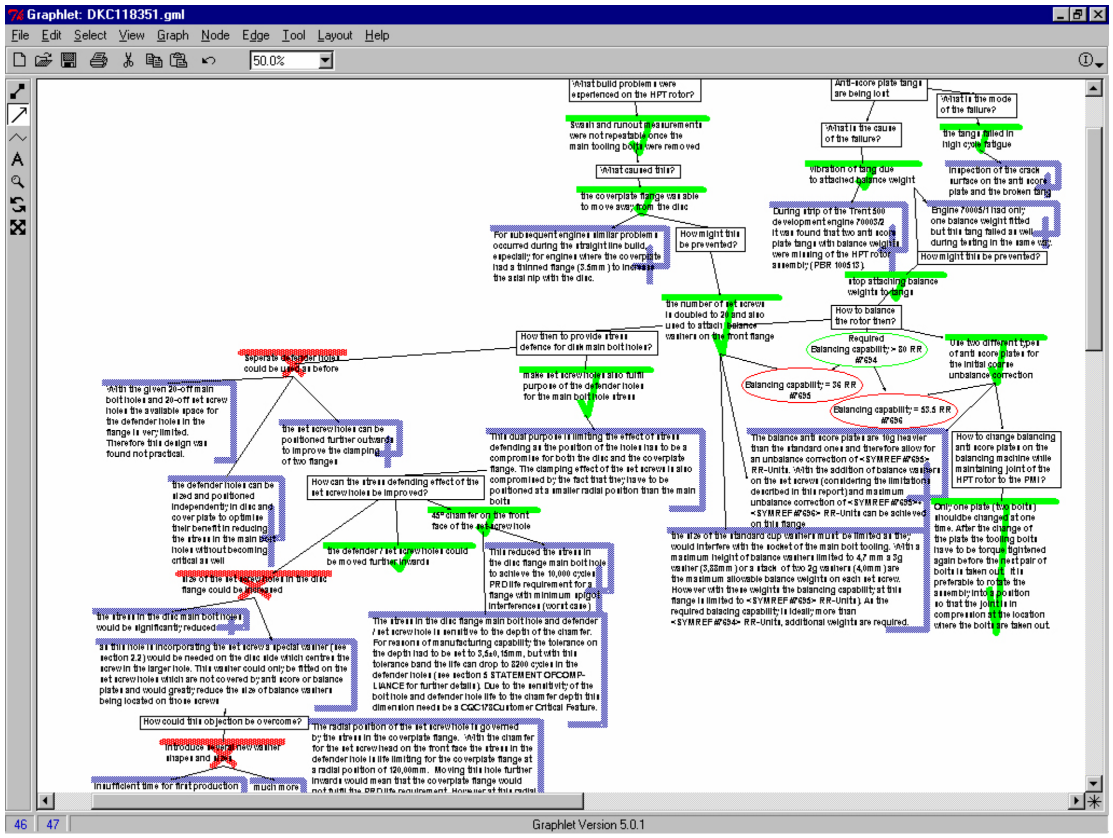

The main chassis for the robot, whether to base it on a existing design or design, manufacture and building something from scratch
Constraint
Cost
Time
size
repeatability
Requirement 1
Requirement 2
Requirement 3
Requirement 4
Cautious Crab
Model Tank
Model Car
Lynxmotion 4WD
Rouge ATR
iRobot Roomba
Already made available and free
Not physical large enough to contain sufficient computing power or enough battery power to do vision processing
Come complete with drive system and motor controller
A demonstration system mounted on a similar device
Retailing for £500, a little over budget
Significant cheaper than the tank counterparts
Difficult to modify a chassis to leave a flat base
A relatively large suspension travel to contend with
Three series robots of slightly different sizes and styles
4 wheels driven by 7.2V gear-head motors linked in pairs on each side to give a skid-steer system
Made of a laser cut Lexan type plastic sheet with aluminium mounting brackets
Slightly too small for our needs, not support much weight
Absence of detailed information
Have a commercial iRobot vacuum cleaning robot not expensive
Not be used in favor of something slightly larger and more stylish
Not used in favour of something suited to our mechanical requirement
□ Small mobile robot designed by ECS
◇ Generic bases
▷ Commercial system
Evaluate
For a brief period we thought about the idea of using a commercial iRobot vacuum cleaning robot as a chassis. Although it was not expensive we decide that it would be inefficient to buy a system and then destroy most of id to fit our purpose.
Use LynxMotion 4WD3 chassis as the mechanical basis for the robot

FIGURE 6.2: The robot chassis design rationale captured by the underlying concepts in SW-DRS



What build problems were experienced on the HPT rotor?

Swash and runout measurements were not repeatable once the main tooling bolts were removed

What caused this?

the coverplate flange was able to move away from the disc

For subsequent engines similar problems occurred during the straight line build, especially for engines where the coverplate had a thinned flange (3.5mm) to increase the axial nip with the disc.

How might this be prevented?

the number of set screws is doubled to 20 and also used to attach balance washers on the front flange

Anti-score plate tangs are being lost

What is the mode of the failure?

the tangs failed in high cycle fatigue

What is the cause of the failure?

Inspection of the crack surface on the anti score plate and the broken tang

vibration of tang due to attached balance weight

During strip of the Trent 500 development engine 70003/2 it was found that two anti score plate tangs with balance weights were missing of the HPT rotor assembly (PBR 100513).

Engine 70005/1 had only one balance weight fitted but this tang failed as well during testing in the same way.

How might this be prevented?

stop attaching balance weights to tangs

How to balance the rotor then?

Required Balancing capability > 30 RR #7694

Balancing capability = 36 RR #7695

Balancing capability = 53.5 RR #7696

Use two different types of anti score plates for the initial coarse unbalance correction

How then to provide stress defence for disk main bolt holes?

make set screw holes also fulfil purpose of the defender holes for the main bolt hole stress

Seperate defender holes could be used as before

With the given 20-off main bolt holes and 20-off set screw holes the available space for the defender holes in the flange is very limited. Therefore this design was found not practical.

the set screw holes can be positioned further outwards to improve the clamping of two flanges

This dual purpose is limiting the effect of stress defending as the position of the holes has to be a compromise for both the disc and the coverplate flange. The clamping effect of the set screws is also compromised by the fact that they have to be positioned at a smaller radial position than the main bolts

The balance anti score plates are 10g heavier than the standard ones and therefore allow for an unbalance correction of <SYMREF #7695> RR-Units. With the addition of balance washers on the set screws (considering the limitations described in this report) and maximum unbalance correction of <SYMREF #7695>+ <SYMREF #7696> RR-Units can be achieved on this flange

How to change balancing anti score plates on the balancing machine while maintaining joint of the HPT rotor to the PMI?

the defender holes can be sized and positioned independently in disc and cover plate to optimise their benefit in reducing the stress in the main bolt holes without becoming critical as well

How can the stress defending effect of the set screw holes be improved?

45° chamfer on the front face of the set screw hole

the defender / set screw holes could be moved further inwards

size of the set screw holes in the disc flange could be increased

This reduced the stress in the disc flange main bolt hole to achieve the 10,000 cycles PRD life requirement for a flange with minimum spigot interferences (worst case)

the size of the standard cup washers must be limited as they would interfere with the socket of the main bolt tooling. With a maximum height of balance washers limited to 4,7 mm a 3g washer (3,88mm) or a stack of two 2g washers (4,0mm) are the maximum allowable balance weights on each set screw. However with these weights the balancing capability at this flange is limited to <SYMREF #7695> RR-Units). As the required balacing capability is ideally more than <SYMREF #7694> RR-Units, additional weights are required.

Only one plate (two bolts) shouldbe changed at one time. After the change of the plate the tooling bolts have to be torque tightened again before the next pair of bolts is taken out. It is preferable to rotate the assembly into a position so that the joint is in compression at the location where the bolts are taken out

the stress in the disc main bolt holes would be significantly reduced

as this hole is incorporating the set screws a special washer (see section 2.2) would be needed on the disc side which centres the screw in the larger hole. This washer could only be fitted on the set screw holes which are not covered by anti score or balance plates and would greatly reduce the size of balance washers being located on those screws

The stress in the disc flange main bolt hole and defender / set screw hole is sensitive to the depth of the chamfer. For reasons of manufacturing capability the tolerance on the depth had to be set to 3,5±0,15mm, but with this tolerance band the life can drop to 8200 cycles in the defender holes (see section 5 STATEMENT OFCOMP-LIANCE for further details). Due to the sensitivity of the bolt hole and defender hole life to the chamfer depth this dimension needs be a CQC178Customer Critical Feature.

How could this objection be overcome?

introduce several new washer shapes and sizes

The radial position of the set screw hole is governed by the stress in the coverplate flange. With the chamfer for the set screw head on the front face the stress in the defender hole is life limiting for the coverplate flange at a radial position of 120,00mm. Moving this hole further inwards would mean that the coverplate flange would not fulfil the PRD life requirement. However at this radial

insufficient time for first production

much more

FIGURE 6.3: A captured design rationale example in DRed

- Easy to produce comprehensive hard copy identical to the screen.
- Traffic-light statuses of all elements, as shown in Figure 3.7
- Anchor links to locations in graphical elements.
- Tunnel linking is more effective for expressing design dependencies than transclusion.
- As the size of the text boxes is uncontrolled, the screen can become clustered, and the users need to scroll down to find an entity.
- No overall map is provided.

6.6 Expert Review

An expert review is an evaluation method moving from a general user review. Expert review evaluation evolved from a Heuristic evaluation (Nielsen, 1992). Heuristic evaluation is a form of usability inspection where usability specialists judge whether each element of a user interface follows a list of established usability heuristics. Expert evaluation is similar, but does not use specific heuristics. This method is to identify usability problems based on established human factors principles. However, as the method relies on experts, the output will naturally emphasise interface functionality and design instead of the properties of the interaction between an actual user and the system.

For the evaluation of SW-DRS, the author proposes to evaluate the content of the system, and not the user interface or HCI aspects. It is advisable to use experts that are or have worked, in an engineering environment for a number of years, in particular, the engineering design environment. Usually two to three analysts evaluate the system with reference to established guidelines or principles, noting down their observations and often ranking them in order of severity. The panel of experts must be established in good time for the evaluation. All analysts need to have sufficient time to become familiar with the system in question along with intended task scenarios. They should operate by an agreed set of evaluative criteria.

Three expert reviewers were chosen because of their experience in the field of engineering design and computer science. The age range of the experts was from 45 years to 55 years, and they have worked in the industry design team for more than 10 years.

In order to evaluate the system usability and utility, three experts were requested to fill out a questionnaire shown in Appendix A on SW-DRS respectively. They also answered a questionnaire on DRed in order to compare with SW-DRS. Here the author used a simple scoring system to quantitive analysis the interview data. The author gives 5 points for answer “yes”, 4 points for “almost”, 3 for “somewhat”, 2 for “not really” and 1 for “no”,and then average the points for each evaluation questions. This leads the conclusion of very poor, poor, ok, good, very good in respect to 1, 2, 3, 4, 5. After the questionnaire procedure, the author also takes several interviews to find out the reasons why the experts would prefer SW-DRS system to DRed, or conversely. Key comments from the interviews are mentioned in Table 6.3. Table 6.3 summaries the comparison result of SW-DRS and DRed as a design rationale capture system. In the scoring system mentioned above, ranging from *Very good = 5* to *Very poor = 1*, summing the score given SW-DRS 29.5 and DRed 26. Hence based on this simple scoring it can conclude the SW-DRS is the better approach.

The results indicate SW-DRS improved or slightly improved the overall view of design process, improved or slightly improved keeping track of design process and improved or slightly improved evaluating and deciding between concepts compared to DRed. However some users found SW-DRS is not very easy for use, especial understand the distinguish between many design element templates during constructing the rationale. As a result, SW-DRS slightly hinder the normal design process compared to traditional design process without design supporting tools. In many cases the designer may not fully understand exactly what is required and therefore may not know what type of expert or information is required to resolve the problem (Crowder et al., 2003).

Metrics	**SW-DRS**	Reason	**DRed**	Reason
Design rationale capture for design task	Very good	Captures rationale using the a full argumentation ontology vocabulary, no missing information	Good	Design rationale constructed during design process with minimal reconstruction
Design rationale representation understanding (user interface)	Good	Comprehensive graphical presentation view on browser	Good / OK	Easy to produce comprehensive hard copy identical to view on screen, screen can be clustered
Computational architecture	Good	Service oriented architecture enable using other web services	Poor	Only hyperlink supports limited services
Storage	Good	Knowledge base stores ontology based information	OK / Poor	No database management system required - simple but not good for potential computational services
Design rationale retrieval through design process	OK	All information based on ontology which can enhance browsability and retrieval if available service enable	Good	Simple to deploy DRed and create a browsable and searchable archive of complete project folder
Reusability	Good	Ontology based architecture enable different level of abstraction and organization	OK	Design rationale captures process of one design only, with some abstraction of the solved and unsolved issues
Output	OK	Simply output rationale graph	Good	Complete output of element status (unsolved or solved), dependency and rationale graph
Design rationale construction overhead	OK / Poor	larger, more expressive notation may take longer to construct	OK	structuring argument as IBIS notation is natural

TABLE 6.3: SW-DRS and DRed comparison. The metrics have been taken from Table 6.2. Scores range from *Very Good/ Good/ OK* to *Poor/ Very Poor*.

6.7 Summary

After a preliminary evaluation and discussion with domain experts, the author finds that by using SW-DRS, the experts conclude that design process can be improved or significant improved, but also it is hard to use. SW-DRS can contribute to make a better comprehensive and effective design, also SW-DRS can easily retrieve rationale if the relevant web service are available which make re-design and reuse possible. However, there are also something negative, currently the interactions in the design process are mainly human, and the use of technology would require a change in the normal ways of working. The designers are only willing to move to automated systems which are accurate and reliable. SW-DRS need to be proved as a accurate and reliable system, which require the evaluation during an ongoing project - a real world project after fully implemented.

Chapter 7

Conclusion and Discussion

This thesis discussed the work undertaken during the two years of the research into the use and design of design rationale system. The author has reviewed several important design rationale systems and based on the current state of the art in capturing design rationale and argumentation structures, the author proposed an integrated formal argumentation model to structure design rationale and also being used during design process. The development of this system for engineering design capture, has demonstrated to improve or slightly improve design rational capture process.

7.1 Conclusion

The research objective of this work is to improve the design reasoning process for engineering design. The author approaches this by firstly understanding the key elements of the process, and then develop an argumentation model based on ontology, a design rationale capture framework with appropriate support mechanisms and tools to facilitate the design process. In this thesis, a number of existed design rationale system, capture method and also application in industry have been reviewed. The author find that the importance of design rationale and basic methodology were identified but there is a lack of efficient capture approach and tool. To achieve these aims, it is necessary to identify the technical and socio-technical factors that influence those design decisions. The research reached the following conclusion:

1. The previous capture models such as the IBIS variant and the Toulmin structure have their own limitations as discussed in Chapter 3. The integration of these models developed in this thesis can improve capturing process.
2. The challenge of design rationale research is to reduce the overload to engineering designers while structuring a rationale, minimize the non-productive effort required

to create design rationale and maximize its utility for those designers creating design rationale. The author make the assumption that all the design documents are formally structured and easy accessible, an automatic or semi-automatic approach to capture rationale will be available by using XML(s), RDF(s) and ontology.

3. The system provided retrieve of the rationale which will facilitate a re-design or related design. This model will provide a foundation for building an effective knowledge based design support system, and will provide for designer more computational services.

Some of the significant issues that the author plans to pursue in our continued research in design rationale are: how can design rationale improve the decision making process, what are the design or system circumstances that influence the use and documentation of design rationale, what mechanism or technology will help make decisions about the level of detail and circumstance under which to document design rationale. The author expects these experimental techniques will enable to discover the answers to the questions above. The results will allow us to develop a design rationale methodology and associated tools to enhance the future use and documentation of design rationale. The initial evaluation results of comparison has indicated SW-DRS had better performance in capture and use design rationale.

7.2 Discussion

The research into design rationale confirms that rationale capture is a challenging problem. The challenge is not to intrude on the designers as they accomplish the design task. Our vision is a design environment with intelligence embedded in the environment, making the designers interact with the design tools in the same way they did before and significantly reduce the time consuming and improve the accuracy for locating the relevant resource.

As discussed in Chapter 4, a framework has been proposed for acquiring, storing and disseminating design rationale. As shown in Figure 4.9, a service-oriented architecture was built to support design liked creative work processes, which enable highly flexible structures and allow the integration of complex of information and specific domain. This is a new idea for building software system, in this architecture, an application's business logic or individual functions are modularized as services for client applications. These services are loosely coupled, which means the service interface is independent of the implementation. Therefore, there is no need for developing all the components to get the system work and new functions can be integrated easily. When the similar situations and features present, the designers will easily reuse the functionality of existing systems rather than building them again. The designers's work has been improved by building the system on top of loosely connected ontology modules which is identified by Semantic Web Portal.

The author will demonstrate the applicability of our model by formalizing an engineering design process from the real case in the future. The Argumentation Ontology Figure 5.1 will also be the basis for capturing design rationale. To support the design process, appropriate tools are needed as semantic web service, formal knowledge representation and knowledge base. The main contribution of this thesis it the first formal argumentation model in web service environment for capture design rationale. This model is an adaption of the IBIS argumentation introduced by Kunz and Rittel (1970). Our methodology will clearly state the arguments in engineering design, capture the rationale more effectively and ease the reinventing when designing.

The work proposed assumes that all the design information is recorded electronically and easy to be access. The proposed system will provide a service to parse the raw data into a simple structured method, because the amount of effort to translate this information into a usable format is considerable. The approach will be to initially implement a rationale representation tool, which can operate without special community knowledge, and the argumentation ontology using OWL. The system will perform increasingly better as more well structured knowledge provided. A hybrid system of semantic web technology and service oriented architecture can be easy and effective for capturing design rationale.

7.3 Future work

Since the model has been built and some part of the system still need to be implemented, a number of works remain to be completed. The main area of the future work is to continue our current work according to what have been discussed in the above chapter and expand some of the other features into the service-oriented system infrastructure. Through the combination of OWL ontology, web service, and knowledge base, a new-generation engineering design environment will be achievable.

One direction of this research is to make this system more usable, which includes building all necessary services that are retrieving the design artifact information through product ontology and exploring the engineering database. The interaction will be mainly implemented by technology rather than face-to-face interaction nowadays. With a human-based system, problems can be discussed and interpreted for the user, making it more likely to proceed with trust. An accurate and reliable system will do exactly the same as the way a designer works, and therefore significantly increase work efficiency.

Another direction is to avoid intruding on the designers as they accomplish the design task. The designers will not be intruded by building the rationale graphs, instead, they work exactly as they used to and the rationale will be captured automatically. That must be implemented as an embedded tool in their design system. There are already a series of projects that capture the rationale about software design by a tool embedded in the software development tools. The author believes such a design rationale system integrated with the CAD or other engineering system will build an automatic or semi-automatic approach.

Appendix A

Questionnaire for expert review

Questionnaire for SW-DRS expert review

Project: Semantic Web Design Rationale Capture
Expert Name:

Please circle what appropriate (sometimes you might want to circle two items) or fill-in:

I. Design rationale representation understanding

i. Did you understand how the system works?
(no/not really/somewhat/mostly/yes)

ii. Did you consider the standard elements useful for expressing your design rationale?
(no/not really/somewhat/mostly/yes)
If no, name the ones that shouldn't have been there, or up to three that the system missed.

iii. Did you consider the graphical representation easy to understand?
(no/not really/somewhat/mostly/yes)

iv. Did you consider the graphical representation easy to learn?
(no/not really/somewhat/mostly/yes)

II. Design rationale retrieval through design process

i. Could you access the captured rationale after design process?
(no/not really/somewhat/mostly/yes)

ii. Do you consider it able to use part of previous design rationale?
(no/not really/somewhat/mostly/yes)

iii. Do you consider it easy to understand the retrievable rationale?
(no/not really/somewhat/mostly/yes)

iv. Do you consider the retrievable rationale useful?
(no/not really/somewhat/mostly/yes)

III. Design rationale capture for design task

i. Do you consider SW-DRS to be a useful system towards daily design task?
(no/not really/somewhat/mostly/yes)

ii. Do you consider the design rationale captured sufficient for design task?
(no/not really/somewhat/mostly/yes)
If no, name the ones that shouldn't have been there, or up to three that the system missed.

iii. Were you pleasing with the way in which design rationale was captured?
(no/not really/somewhat/mostly/yes)

IV. Design rationale reusability

i. Do you consider it useful to be able to look at other designer's rationale during your own rationale creation?
(no/not really/somewhat/mostly/yes)

ii. Do you consider it useful to be able to reuse part of your own rationale?
(no/not really/somewhat/mostly/yes)

iii. Could you reuse previous design rationale during your own design process?
(no/not really/somewhat/mostly/yes)

V. Overhead of design rationale construction

i. Was SW-DRS easy to use?
(no/not really/somewhat/mostly/yes)
ii. Did you know how to build design rationale by the tools in SW-DRS?
(no/not really/somewhat/mostly/yes)
iii. Did you consider the work required for building design rationale acceptable?
(no/not really/somewhat/mostly/yes)
iv. Were you pleasing with the extra work required for building design rationale?
(no/not really/somewhat/mostly/yes)

VI. System computational architecture

i. Did you consider the user interface of SW-DRS sufficient?
(no/not really/somewhat/mostly/yes)
If not, name a maximum of three things that missed or that should be improved (for each interface).

ii. Did you consider the implementation of SW-DRS difficult?
(no/not really/somewhat/mostly/yes)
iii. Did you consider the architecture of SW-DRS reliable?
(no/not really/somewhat/mostly/yes)
iv. Did you consider that the architecture of SW-DRS provides extendability?
(no/not really/somewhat/mostly/yes)

VII. Output of design rationale capture

i. Do you consider SW-DRS output everything you need during design process?
(no/not really/somewhat/mostly/yes)
If no, name the ones that shouldn't have been there, or up to three that the system missed.
ii. Were you pleasing with the format of output design rationale?
(no/not really/somewhat/mostly/yes)
iii. Were you pleasing with the quality of output design rationale?
(no/not really/somewhat/mostly/yes)

VIII. Storage of captured design rationale

i. Did you consider the triple-store reliable?
(no/not really/somewhat/mostly/yes)
ii. Did you consider the triple-store complex?
(no/not really/somewhat/mostly/yes)
iii. Did you consider the enquiry of triple-store effective?
(no/not really/somewhat/mostly/yes)

References

R. Ackoff. From data to wisdom. *Journal of Applies Systems Analysis*, 16:3–9, 1989.

S. Ahmed and K. Wallace. Developing a support system for novice designers in industry. In *13th International Conference on Engineering Design (ICED 01), Design Management*, pages 75–82, Glasgow, 2001.

AKT. (Advanced Knowledge Technologies). http://www.aktors.org/akt/, 2003.

M. Aurisicchio, R. Bracewell, and K. Wallace. Evaluation of DRed a way of capturing and structuring engineering design processes. In *NordDesign*, Reykjavik, Iceland, 2006.

V. Basili, G. Cadiera, and H. Rombach. *Encyclopaedia of Software Engineering*. Wiley, 1994.

G. Bellinger, D. Castro, and A. Mills. Data, information, knowledge, and wisdom. http://www.systems-thinking.org/dikw/dikw.htm, November 2002.

T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May:34–43, 2001.

L. Blessing. *A Process-Based Approach to Computer Supported Engineering Design*. PhD thesis, University of Twente, Netherlands, 1994.

R. Bracewell, S. Ahmed, and K. Wallace. DRed and design folders, a way of capturing, storing and passing on, knowledge generated during design projects. In *Design Automation Conference, ASME*, Salt Lake City, Utah, USA, 2004.

D. Brissaud, O. Garro, and O. Proveda. Design process rationale capture and support by abstraction of criteria. *Research in Engineering Design*, 14(3):162–172, 2003.

B. Chandrasekaran and Y. Iwasaki. Functional representation as design rationale. *IEEE Computer*, 26(1):48–56, 1993.

E. Conklin and K. Burgess-Yakemovic. A process-oriented approach to design rationale. *Human Computer Interaction*, 6(3&4):257–391, 1991.

J. Conklin and M. Begeman. gIBIS: a hypertext tool for exploratory policy discussion. *ACM Trans. Inf. Syst.*, 6(4):303–331, 1988. ISSN 1046-8188.

I. Cooper and R. Crowder. A mobile robot platform for navigation experiments. Technical report, School of Electronics and Computer Science, University of Southampton, 2006.

CORE. (Collaborative Orthopaedic Research Environment). http://www.core.ecs.soton.ac.uk/, November 2006.

R. Crowder, R. Bracewell, G. Hughes, M. Kerr, D. Knott, M. Moss, C. Clegg, W. Hall, K. Wallace, and P. Waterson. A future vision for the engineering design environment: A future sociotechnical scenario. In *International Conference on Engineering Design*, Stockholm, Sweden, 2003.

T. Davenport and L. Prusak. *Working Knowledge*. Harvard Business School Press, Boston, 1998.

D. Englebart. *A Conceptual Framework for the Argumentation of Man's Intellect*. Vistas in Information Handling, Spartan Books, Washington, D.C, London, 1963.

N. Fenton and S. Pfleeger. *Software Metrics, A rigors and Practical Approach*. Internation Thompson Computer Press, 1997.

G. Fischer, A. Lemke, R. McCall, and Morch A. Making argumentation serve design. *Design rationale: concepts, techniques, and use*, pages 267–293, 1996.

G. Fischer, R. McCall, and A. Morch. Design environments for constructive and argumentative design. In *CHI '89: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 269–275, New York, NY, USA, 1989a. ACM Press. ISBN 0-89791-301-9.

G. Fischer, R. McCall, and A. Morch. JANUS: integrating hypertext with a knowledge-based design environment. In *HYPERTEXT '89: Proceedings of the second annual ACM conference on Hypertext*, pages 105–117, New York, NY, USA, 1989b. ACM Press. ISBN 0-89791-339-6.

G. Fischer and J. Oswald. Knowledge management: Problems, promises, realities and challenges. *IEEE Intelligent Systems*, January/February, 2001.

X. Fischer, C. Merlo, J. Legardeur, L. Zimmer, and A. Anglada. Knowledge management and support environment in early phase of design process. In *Design Engineering Technical Conferences and Computer and Information in Engineering Conference, ASME*, Salt Lake City, Utah, USA, 2004.

A. Garcia, H. Howard, and M. Stefik. Active design documents: A new approach for supporting documentation in preliminary routine design. Technical report, Stanford Univ. Center for Integrated Facility Engineering, Stanford, Calif, 1993.

T. Grüber. Towards principles for the design of ontologies used for knowledge sharing. *nternational Journal Human-Computer Studies*, 43:907–928, 1993.

T. Grüber and D. Russell. Derivation and use of design rationale information as expressed by designers. Technical report, Knowledge System Laboratory, Stanford, 1992.

J. Grudin. Utility and usability: Research issues and development contexts. *Interacting with Computers*, 4(2):209–217, 1992.

U. Harigopal and A. Satyadas. Fundamental of knowledge management. *IEEE Tutorial*, 2001.

IDC. (international Data Corporation). http://www.idc.com/, 2006.

Y. Kato and K. Hori. A computational model of argumentative design rationale. In *5th Workshop on Computational Models of Natural Argument, CMNA2005*, Edinburgh, U.K., 2005.

D. Khadilkar and L. Stauffer. An experimental evaluation of design information reuse during conceptual design. *Journal of Engineering Design*, 7(4):331–339, 1996.

R. Kosala and H. Blockeel. Web mining research: A survey. *ACM SIGKDD*, 2:1–15, 2000.

W. Kunz and W. Rittel. Issues as elements of information systems. Technical report, Center for Planning and Development Research, University of California, Berkeley, 1970.

J. Lee. SIBYL: a tool for managing group design rationale. In *CSCW '90: Proceedings of the 1990 ACM conference on Computer-supported cooperative work*, pages 79–92, New York, NY, USA, 1990. ACM Press. ISBN 0-89791-402-3.

J. Lee. Design rationale system: Understanding the issues. *IEEE Expert*, 12(3):78–85, 1997.

J. Lee and K.-Y. Lai. What is design rationale? *Human-Computer Interaction*, 6 (3-4): 251–280, 1991.

R. Lee. Scalability report on triple store applications. http://simile.mit.edu/repository/site/reports/stores/stores.pdf, July 2004.

A. MacLean, R. Young, V. Bellotti, and T. Moran. Questions, options and criteria: Elements of design space analysis. *Human Computer Interaction*, 6:201–250, 1991.

P. Maes. *Designing Autonomous Agents: Theory and Practise From Biology to Engineering and Back*. Cambridge, MA: MIT Press, March 1991.

R. McCall. PHI: A conceptual foundation for design hypermedia. *Design Studies*, 12: 30–41, 1991.

A. Medeiros, D. Schwabe, and B. Feijó. Kuaba ontology: Design rationale representation and reuse in model-based designs. In *Model-Based Designs*, pages 241–255, 2005.

T. Mezher, M. Abdul-Malak, I. Ghosn, and M. Ajam. Knowledge management in mechanical and industrial engineering conculting: A case study. *Journal of Management in Engineering*, July:138–147, 2005.

D. Millard, F. Tao, K. Doody, A. Woukeu, and H. Davis. The knowledge life cycle for e-learning. *Continuing Engineering Education and Lifelong Learning*, 16:110–121, 2006.

J. Molavi, R. McCall, and A. Songer. A new approach to effective use of design rationale in practice. *Journal of architectural engineering, ASCE*, June, 2003.

J. Nanda, H. Thevenot, T. Simpson, and S. Kumara. Exploring semantic web technologies for product family modeling. In *International Design Engineering Technical Conferences and Computers & Information in Engineering Conference, ASME, DETC2004*, Salt Lake City, UT, 2004.

M. Nick and K. Althoff. Engineering experience base maintenance knowledge. *Lecture Notes in Computer Science*, 2176:222, 2001.

J. Nielsen. *Usability engineering*. Boston: Academic Press, 1993.

Jakob Nielsen. Finding usability problems through heuristic evaluation. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 373–380, New York, NY, USA, 1992. ACM Press.

Y. Nomaguchi, A. Ohnuma, and K. Fujita. Design rationale acquisition in conceptual design by hierarchical integration of action, model and argumentation. In *Design Engineering Technical Conferences and Computer and Information in Engineering Conference, ASME*, Salt Lake City, Utah, USA, 2004.

Kurt Nørmark and Kasper Østerbye. Rich hypertext: a foundation for improved interaction techniques. *International Jounal of Human-Computer Study*, 43(3):301–321, 1995. ISSN 1071-5819.

G. Pahng, S. Ha, and S. Park. A design knowledge management framework for active design support. In *Proceedings of DETC'99: 1999 ASME Design Engineering Technical Conferences*, pages 1–8, 1999.

W. Regli, X. Hu, M. Atwood, and W. Sun. A survey of design rationale systems: approaches, representation, capture and retrieval. *Engineering with Computers*, 12 (3-4):209–235, 2000.

Y. Reich. Improving the rationale capture capability of QFD. *Engineering with Computers*, 16:236–252, 2000.

N. Shadbolt and K. O'Hara. *Advances Knowledge Technologies: Selected Papers.* University of Southampton, School of Humanities (History), 2004.

F. Shipman and R. McCall. Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication. *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, 11(2):141–154, 1997.

S. Shum. Design argumentation as design rationale. *The Encyclopedia of Computer Science and Technology (Marcel Dekker Inc:NY)*, 35(20):95–128, 1996.

S. Shum and N. Hammond. Argumentation-based design rationale: what use at what cost? *International Journal of Human-Computer Studies*, 40(4):603–652, 1994.

S. Sigman and X. Liu. A computational argumentation methodology for capturing and analyzing design rationale arising from multiple perspectives. *Information and software Technology*, 45:113–122, 2003.

Y. Sim. *Capturing Organisational Knowledge from Documentation for Expert Finding.* PhD thesis, University of Southampton, United Kingdom, 2004.

C. Soukup and S. Titsworth. A description of Toulmin's layout of argumentation. http://www.unl.edu/speech/comm109/Toulmin/layout.htm, May 1998.

J. Steyn. Data, information, knowledge and wisdom. http://www.knowsystems.com/km/definition.html, 2004.

R. Suresh and C. Egbu. Knowledge management: issues, challenges and benefits for a sustainable urban environment. In *The international construction research conference of the Royal Institution of Chartered Surveyors, COBRA*, Leeds, UK, 2004.

K. Sveiby. What is knowledge management? http://www.sveiby.com/articles/KM.html, March 1996.

S. Toulmin. *The Uses of Arguments.* Cambridge University Press, 1958.

J. Ullman. *Principles of database and knowledge-base systems, Vol. I.* Computer Science Press, Inc., New York, NY, USA, 1988. ISBN 0-88175-188-X.

G. Voland. *Engineering by Design.* Addison Wesley Longman, Inc., 1999.

W3C. World Wide Web Consortium. http://www.w3.org/, 2007.

H. Werner and S. Ahmed. Design capturing with a model system using event triggered procedures. In *International Conference on Engineering Design*, Munich, Germany, 1999.

G. Wills. *The Design and Evaluation of Industrial Hypermedia.* PhD thesis, University of Southampton, United Kingdom, 2000.

S. Wong, R. Crowder, N. Shadbolt, and G. Wills. Knowledge management for a large service-oriented corporation. In *Proceedings of the 6th International Conference on Practical Aspects of Knowledge Management (PAKM)*, 2006.