Demand Support by Virtual Experts

Supporting the Client during the Inception Phase of a Building and Construction Project

Hans Schevers

Demand Support by Virtual Experts

Supporting the Client during the Inception Phase of a Building and Construction Project

PROEFSCHRIFT

ter verkijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.dr.ir. J.T. Fokkema,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen
op 24 juni 2004 om 14.30

door

Hans Albert Josef Schevers

civiel ingenieur
geboren te ‘s-Hertogenbosch

Dit proefschrift is goedgekeurd door de promotors:

Prof.ir. F.P. Tolman

Prof.dr.ir. H.A.J. de Ridder

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof.ir. F.P. Tolman	Technische Universiteit Delft, promotor
Prof.dr.ir. H.A.J. de Ridder	Technische Universiteit Delft, promotor
Prof.dr ir. I.S. Sariyildiz	Technische Universiteit Delft
Prof.dr.ir. B. de Vries	Technische Universiteit Eindhoven
Prof.ir. G. Augenbroe	Georgia Institute of Technology, Atlanta
dr.ir. P.H. Willems	TNO Bouwprocesinnovatie

The research presented in this thesis has been sponsored by TNO Building and Construction Research

Published and distributed by:
Hans Schevers
Lariestraat 25
5482 LH Schijndel
The Netherlands
E-mail: HansSchevers@hotmail.com

ISBN 90-9018215-2
@ Copyright 2004 by Hans Schevers
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior permission ot the author.

To my parents

Albert & Diny

Preface

This thesis presents the results of my PhD study at the Civil Engineering faculty of Delft University of Technology in the Netherlands. The research topic is a natural follow-up of my Masters study where I developed a Hospital Inception Support System that allows hospital facility management to invent and study alternative design solutions. The desire to take an in-depth look into the subject of Inception Support in the Building and Construction industry provided the motivation to start with the research reported here.

A number of people have contributed in various ways to this thesis. First, I am indebted to: Frits Tolman for the opportunity he gave me and for his support during the whole research and Hennes de Ridder for his help and enthusiasm. In addition, I have to thank my colleagues at Delft University of Technology for their input and co-operation: Edwin Dado, Saban Ozsariyildiz, Reza Beheshti, Reinout van Rees, Martien Reniers, Peter van de Veer, Henk Boere, Gina de Kok, Olga van Paassen, Gertjan van der Wel and Tjeerd Dierckxens. Furthermore, I want to thank many people at TNO who I met during my research and contributed someway or the other: Robert Los, Renzo van Rijswijk, Wim Plokker, Bert van Elkhuizen, Luc Soethout, Peter Willems, Michel Böhms, Bart Luiten, Sander van Nederveen, Peter Bonsma, Anneleen Goovaerts, Maurits Dekker and all other members who made my TNO time very pleasant.

I also would like to thank Jan Selen for his illustrations and Ilse Kuijpers for her love and support.

Delft, 2004

Hans Schevers

Summary

1 Introduction

This thesis describes an inquiry into the application of Information and Communication Technology (ICT) to support the client of the Building and Construction (BC) industry. Chapter One presents a first insight into the role of the client. Noticeably, the role of the client is rather weak because he lacks sufficient control over the building process, which can be unpredictable regarding the budget, the expected quality and the duration of the project.

Figure 1 The client lacks sufficient control over the building process.

Chapter One also discusses the potential of ICT for the BC industry as 'enabler' and argues that ICT is able to increase the performance of the BC industry. In this regard, the following initial research question is raised:

- Can ICT improve the client's role in a Building and Construction process?

2 Analysis of the Role of the Client in the Building Process

The client needs to specify a demand (such as product and service requirements) during the building process. The demand is defined in this thesis as the difference between the current situation and the preferred situation including the transformation process. When the client does not have a clear perception of the current and preferred situations, he runs the risk of unsatisfactory results.

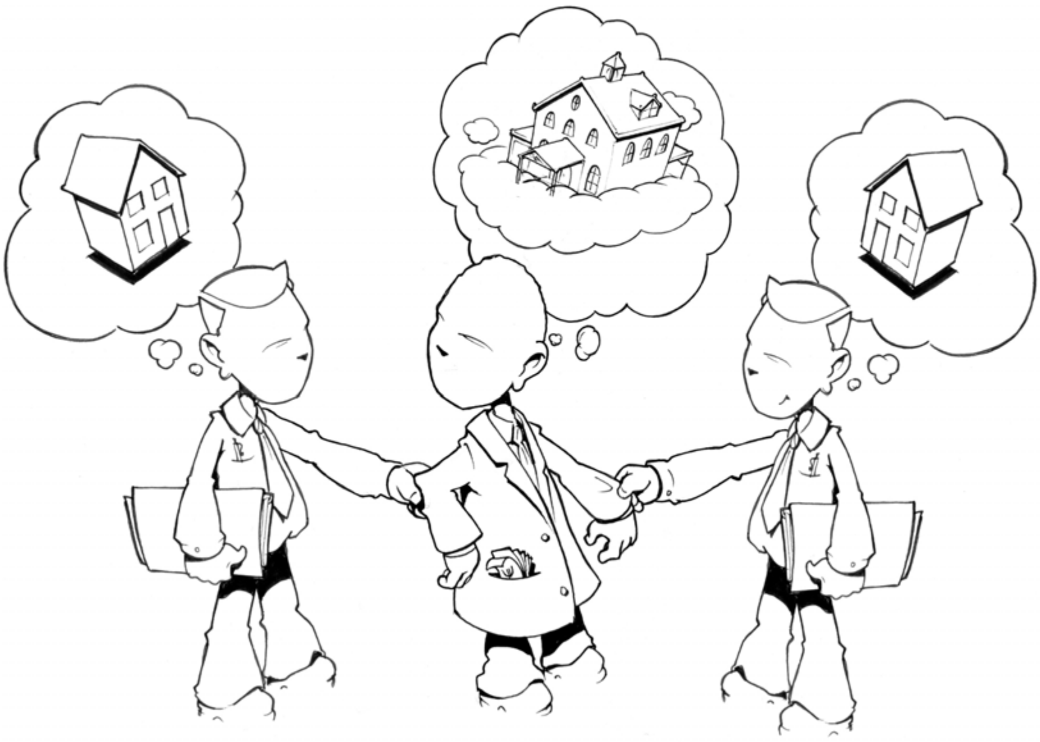


Figure 2 Without clear perceptions on the current and preferred situations, the client runs risks in becoming dissatisfied with the result.

During the project, the client can adjust his perception due to the availability of more relevant information. This change in perception creates a new demand leading to extra costs, while parts of the previously defined demand will become obsolete. Especially, during the inception phase, the client may run high risks on specifying an incorrect demand because relevant information is not available or not sufficient. Although a lot of knowledge is available in the BC industry, several reasons are present why the client hardly gains benefits from that knowledge and consequently lacks the necessary information in the inception phase. Providing the client a suitable access to the Body of Construction Knowledge during the inception phase will support him to develop a demand that brings the expected ‘added value’. This research focuses on the question if ICT is able to provide the client a suitable access to the Body of Construction Knowledge without the presence of experts and without requiring the client to be an expert.

3 Analysis of the ICT-Enabled Access to the Body of Construction Knowledge

Chapter Three analyses the ICT-enabled access to the Body of Construction Knowledge (BoCK) and concludes that the available software applications do not adequately provide the client with access to the BoCK during the inception phase because:

- Only a small and specific part of the BoCK can be accessed using existing applications. Extending or integrating the existing applications is difficult because most of these applications are closed systems.

- ICT-enabled access to BoCK is fragmented while the client requires access to an integrated Body of Construction Knowledge.

- Currently no suitable mechanism is available for dealing with problems related to input information. Current applications only accept input information at a fixed level of detail. Information available during the inception phase can be characterised as scarce and ill-defined, i.e. incomplete, incoherent, conflicting, vague, uncertain and on different levels of detail. Most applications cannot be used in the inception phase because they require a specific set of (detailed) information in order to operate.

- Most existing applications are domain specific and therefore less useful for the client.

- Knowledge is embedded in the software applications and not visible/open for the client or other experts.

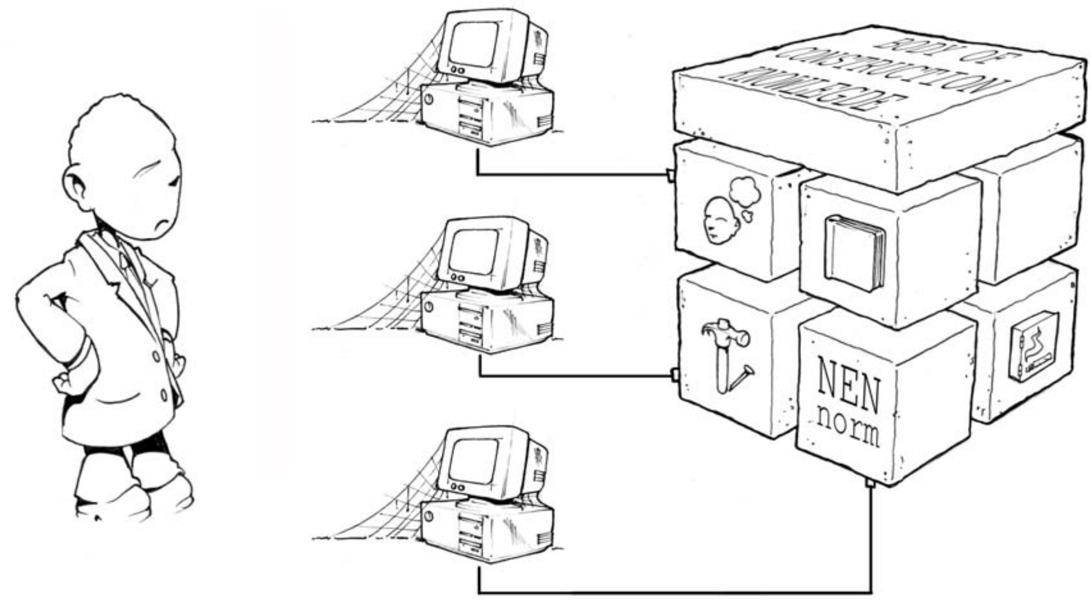


Figure 3 The client does not have a suitable ICT-Enabled access to the fragmented Body of Construction Knowledge.

4 Analysis of State-of-the-Art Knowledge Technology

Knowledge-Based Systems as part of Knowledge Technology, offer new possibilities for reasoning on the BoCK such as reasoning on more and new parts of the BoCK. Many different types of Knowledge-Based Systems are available, each with their own advantages. For example, Knowledge-Based Systems are capable of reasoning and dealing with scarce and ill-defined information. Technologies related to Semantic Web, agents and Web Services support a distributed approach for interoperability between Knowledge-Based Systems. However, a large number of (overlapping) approaches and implementations are available with each their own advantages and disadvantages.

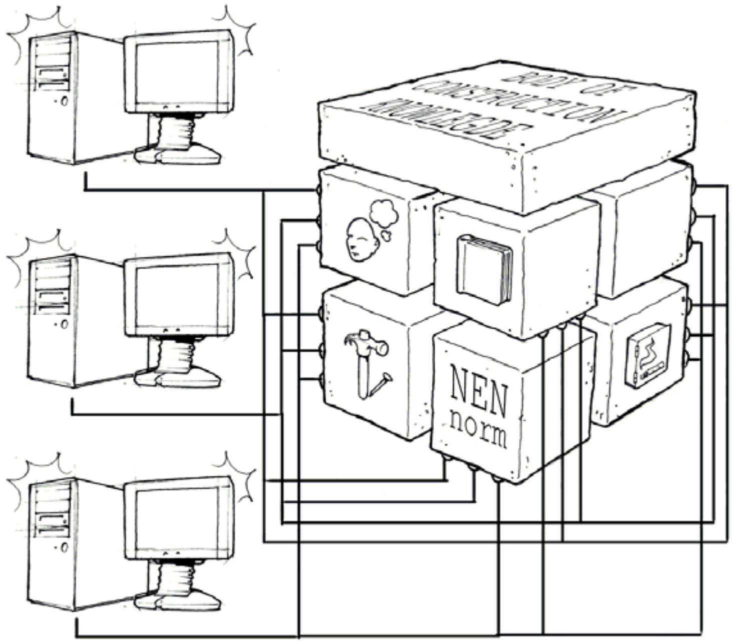


Figure 4 State-of-the-art Knowledge Technology offers numerous possibilities for reasoning on the Body of Construction Knowledge.

5 Detailing the Research Question

To support the client in the building process, he needs access to the BoCK as soon as possible. Therefore, an application is necessary that provides the client access to the BoCK during the inception phase in order to support him with the development of the demand; a Demand Support System. The following five detailed research questions emerge in Chapter Five:

1. *What are the requirements for a Demand Support System?*
2. *Which conceptual architecture is suitable for a Demand Support System?*
3. *How can a Demand Support System incorporate legacy software?*
4. *How can a Demand Support System support the inexperienced client?*

5. *How can a Demand Support System be made that can cope with the upcoming new generation of knowledge intensive computer applications?*

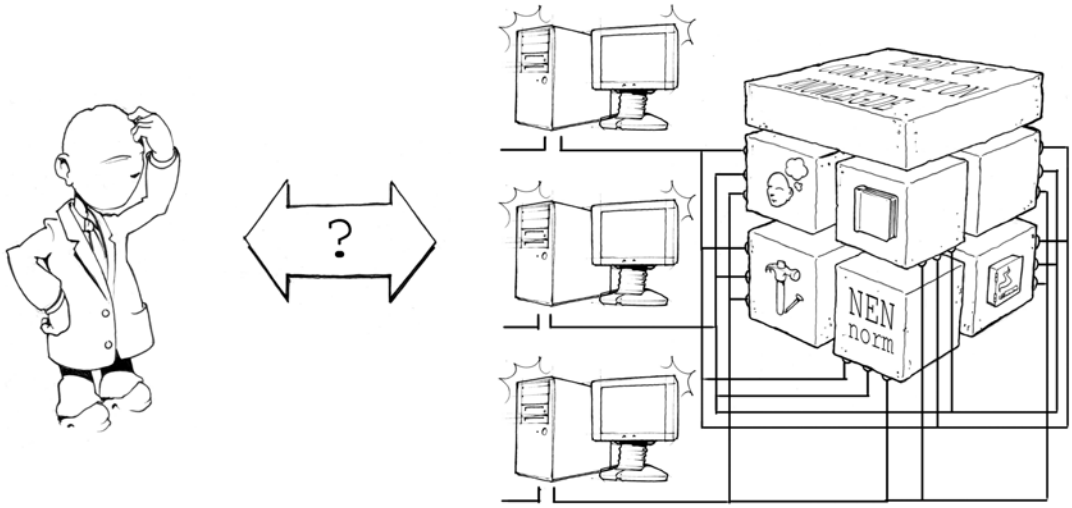


Figure 5 Detailing the initial research question.

6 Functional Design of a Demand Support System Using Virtual Experts

Chapter Six answers the five detailed research questions of Chapter Five and proposes a conceptual architecture for the Demand Support System (DSS). In this conceptual architecture, Virtual Experts proactively use a network of distributed and independent Knowledge-Based Systems in order to infer on the shared Project Model. Therefore, Virtual Experts use Knowledge Components, which provide an interface between the Project Model and a Knowledge-Based System.

The proposed architecture enables the integration of legacy applications and enables the interpretation of technical information into different levels of abstraction in order to make it understandable for the client.

The use of the ontology supports the conceptualisation of Objects of Interests spanning different levels of detail and facilitates flexibility regarding the extension of the DSS. Therefore, a project-type-specific DSS can be created that has the potential to evolve into a larger and more sophisticated system. This evolutionary approach can be supported by making the DSS compliant with the Semantic Web, which enables the DSS to cope with the upcoming generation of knowledge intensive applications.

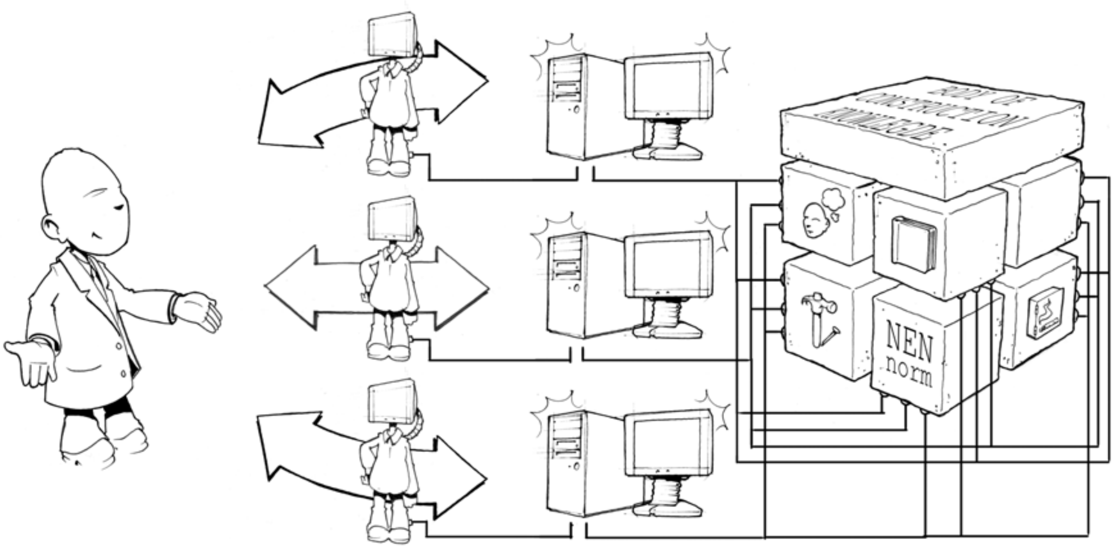


Figure 6 Virtual Experts give the client access to the Body of Construction Knowledge.

7 Implementation of a Prototype Demand Support System

Chapter Seven discussed the implementation of the proposed conceptual architecture. The Project Model, Behaviour Objects and the ontology are implemented and can be shared over the Internet. The ontology is open and its content can be modified. Also the other parts of the conceptual architecture are implemented that include Virtual Experts, Knowledge Components and Pattern search routines.

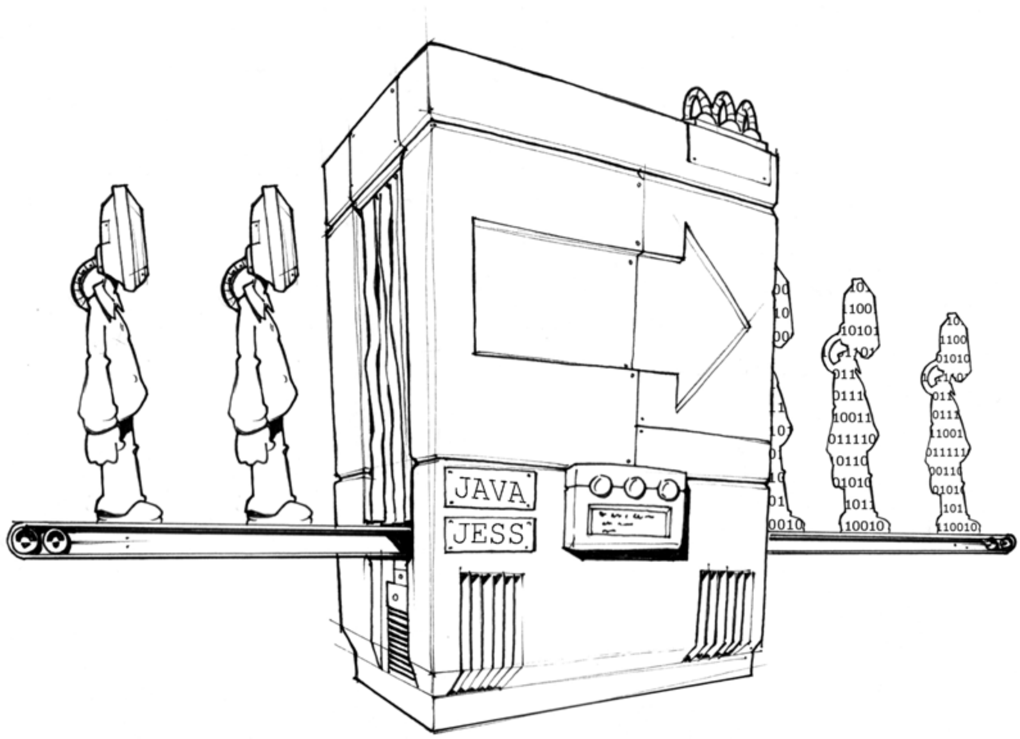


Figure 7 Implementation of Virtual Experts for a Demand Support System.

User interfaces visualising shape related objects in 2D and 3D are created in order to support the creation of a DSS.

To support the creation of a DSS for specific project-types, an ontology editor and a Knowledge Component editor are created. These editors support the creation of Demand Support Systems, without having to implement or change the conceptual architecture or the user interfaces.

8 Case Studies

Chapter Eight demonstrates the potential of the conceptual architecture and of Demand Support Systems in three case studies. By creating project-type-specific ontologies and several Virtual Experts, three Demand Support Systems are developed. These Demand Support Systems are based on the same conceptual architecture and demonstrate various features of the proposed DSS such as reasoning with ill-defined information, the use of legacy applications, the use of multiple Virtual Experts, coping with variable input, supporting different levels of detail, et cetera. Finally, the prototypes demonstrate the usability of the conceptual architecture and the usefulness of the DSS for the (inexperienced) client in the BC industry during the inception phase.

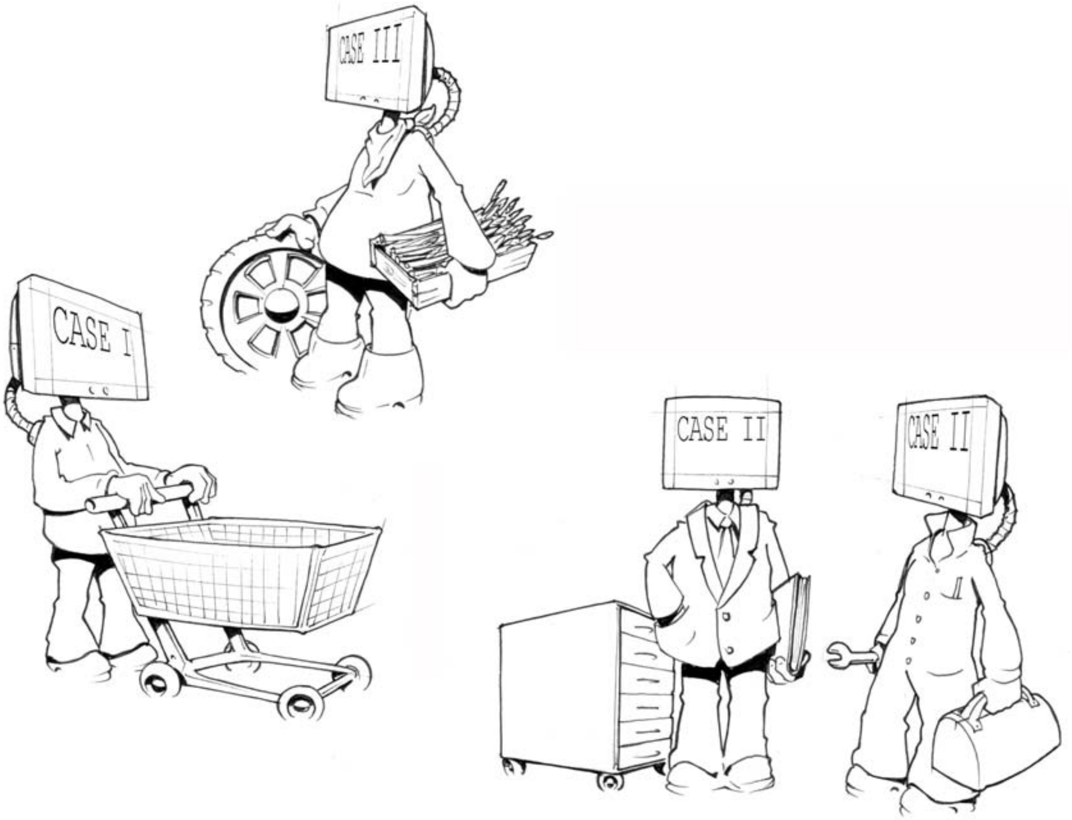


Figure 8 Virtual Experts for the case studies.

9 Conclusions

The Demand Support System using Virtual Experts proposed in this research is an attempt to support the client during the inception phase. The current state of the Demand Support System is not production ready. However, it has been demonstrated that the concept of Demand Support Systems is very promising for supporting the client in order to improve his role in the inception phase.

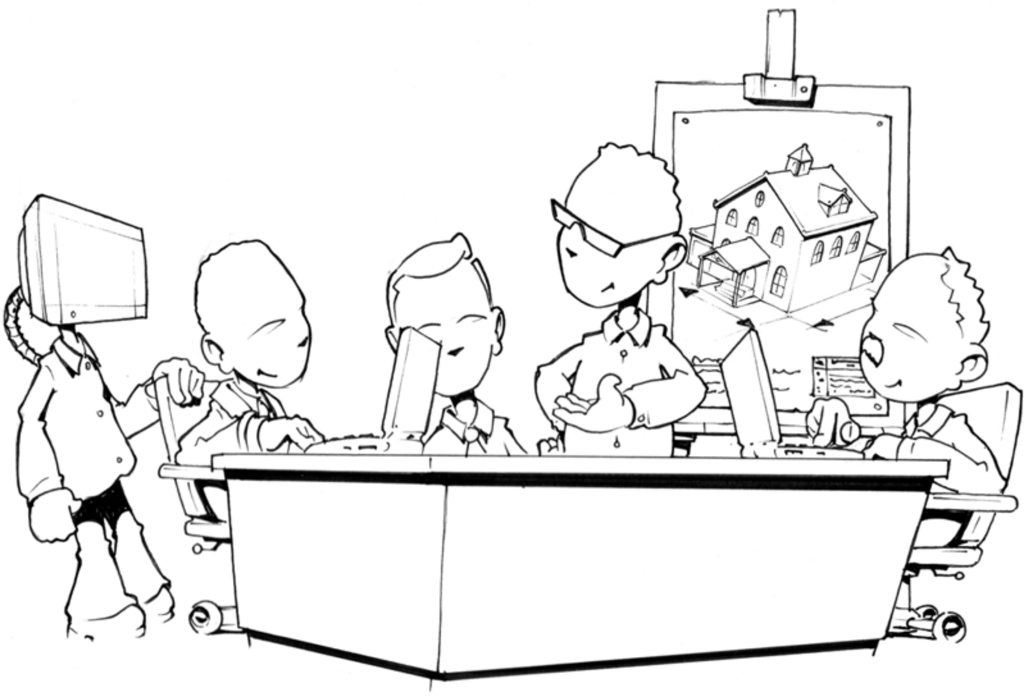


Figure 9 Supporting the client during the inception phase using a Demand Support System with Virtual Experts seems feasible but further research and development is required.

TABLE OF CONTENTS

1 Introduction

This chapter introduces the research topic and gives an overview of the structure of the thesis.

The client lacks sufficient control over the building process.

1.1 AN INTRODUCTION TO THE RESEARCH TOPIC

Products of the Building and Construction (BC) industry often have a prominent appearance and can have a great impact on the environment. The total impact and the value appreciation of the BC industry products involve many stakeholders including the client.

The client of the BC industry is not always able to evaluate the actual product before it is realised. Unlike other industries, making a prototype in the BC industry is normally not feasible. The client is nevertheless involved in a dynamic process in which the product evolves from idea to final product. During this process, the client has to specify requirements, but also has to make decisions on aspects such as contract types, organisation models, et cetera. Most clients are not familiar with this process and therefore may not be aware of the risks they run [Dullaart & Hoeth, 1998]. Many projects are unpredictable regarding the budget, the expected quality and the duration of the project. The role of the client seems to be rather weak because of the lack of sufficient control and involvement in the project. The severity of this weak role can be intensified by the poor performance of the BC industry [De Ridder & Vrijhoef, 2003; Egan et al., 1998].

In order to cope with its poor performance and to increase its efficiency and effectiveness, the BC industry has started to use Information and Communication Technology (ICT). Presently computers play an important role during design, engineering and realisation processes. There is an increasing tendency in applying ICT by many professionals in the course of their activities. This includes but is not limited to Computer Aided Design systems, simulation software, e-mail or the Internet. The uptake of ICT by the BC industry can be characterised as low in comparison with other industries [Korbijn, 1999]. However, an increasing use of ICT by the BC industry in the near future is envisaged by a recent study of the Dutch Ministry of Economic Affairs [Spekkink, 2002b]. In addition, some research projects, such as ICCI [2003] and Roadcon [2003], identify many opportunities for ICT in the BC industry. These projects claim that ICT can increase the performance of the BC industry by reducing the costs (of failures), by improving the quality of the product and by reducing the project delivery time. It seems likely that clients of the BC industry will benefit from ICT as well as the industry itself.

1.2 INITIAL RESEARCH QUESTION

The discussion above leads to the following premises:

- The role of the client within the BC industry is weak.
- ICT is recognized as an enabler for improving the performance of the BC industry.

Based on these premises, the following research question can be formulated:

- Can ICT improve the client's role in a Building and Construction process?

1.3 OUTLINE OF THE THESIS

To deal with the initial research question, Chapter Two embarks on an analysis of the client's role in the building process. Furthermore, Chapter Two formulates the scope of this research that focuses on providing the client with ICT-enabled access to BC knowledge in the inception phase of the building process.

Chapter Three investigates the extent in which the client can access the BC knowledge through ICT-enabled means. This inquiry indicates that the current available ICT-enabled access to BC knowledge is either insufficient or not suitable for the client.

Chapter Four discusses the state-of-the-art of Knowledge Technology and investigates the suitability of this technology for providing a client with access to the BC knowledge.

Chapter Five reflects on the analysis of previous chapters, and decomposes the initial research question into five new research issues.

Chapter Six presents the theoretical body of the thesis and attempts to answer the five research issues mentioned in Chapter Five. This chapter describes the functional architecture of a proposed Demand Support System using Virtual Experts containing amongst others a conceptual architecture and top-level requirements.

Chapter Seven discusses several issues for the implementation of the Demand Support System including Virtual Experts. In this chapter, the implementation of key components will be discussed.

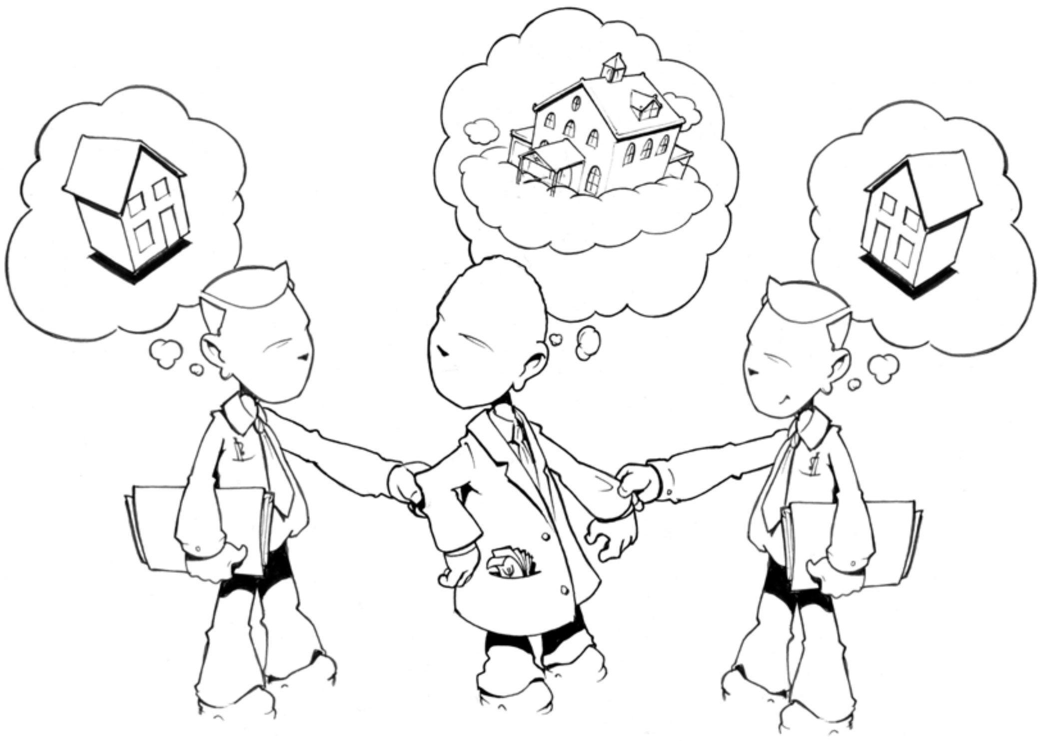
Chapter Eight describes three different Demand Support Systems using different Virtual Experts that are all based on the same architecture but each focusing on a different project type. These case studies provide insight into the usability and potential of the proposed architecture as well as the Demand Support System itself including the Virtual Experts.

Chapter Nine concludes this research and formulates some recommendations for future development of ICT-enabled client support. In this regard, the Demand Support System sketches a framework for Virtual Experts that give the client an ICT-enabled access to BC knowledge, hence offering a mechanism for the client's effective participation during the inception phase.

2

Analysis of the Role of the Client in the Building Process

Chapter Two analyses the role of the client in the building process. Based on this analysis, the scope of this research is defined.



Due to a lack of expert knowledge, the client is not always able to make the correct decisions.

2.1 THE ROLE OF THE CLIENT IN THE BUILDING PROCESS

2.1.1 Who is the Client?

A value demanding and a value supplying party can be distinguished in a Building and Construction (BC) project [De Ridder, 2002]. The value demanding party has an interest in value for money and gains value when the value of the product/services that he demands is higher than the (value of) the price he has to pay for it (Figure 2-1). The difference between value and price is defined as the 'added value'.

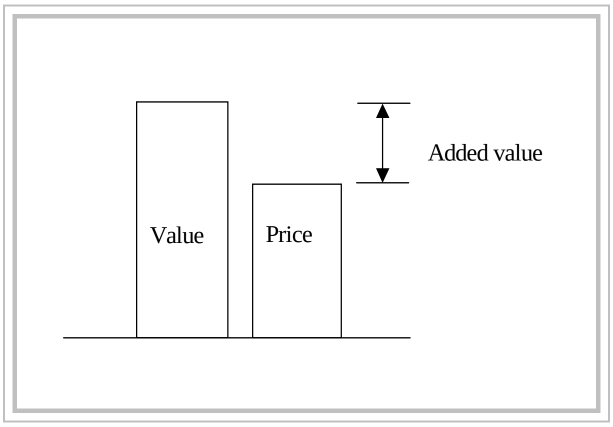


Figure 2-1 Value demanding party asks for value for money.

The value supplying party has an interest in money for value and delivers a product or service for a certain price, which is preferably higher than the costs in order to make profit (Figure 2-2).

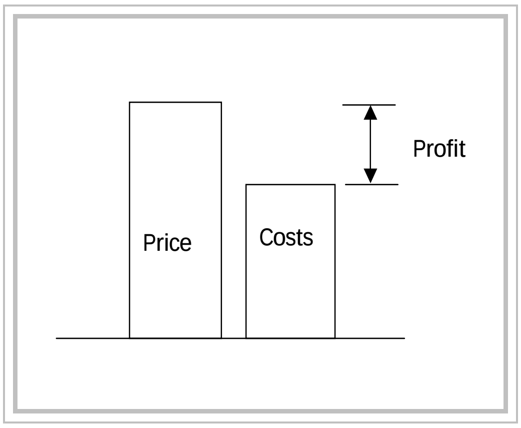


Figure 2-2 Value supplying party asks for money for value by delivering a product or service.

From the start of a BC project, the value demanding party consists of a network of smaller value demanding parties like stakeholders, users, owners, governments, et cetera. Each demanding party has its goals that result in requirements on the BC product. The value supplying party consists of a network of smaller value supplying parties like architects, engineers, contractors, suppliers, et cetera (Figure 2-3). These parties can of course become a value demanding party as well but only to support their own supply.

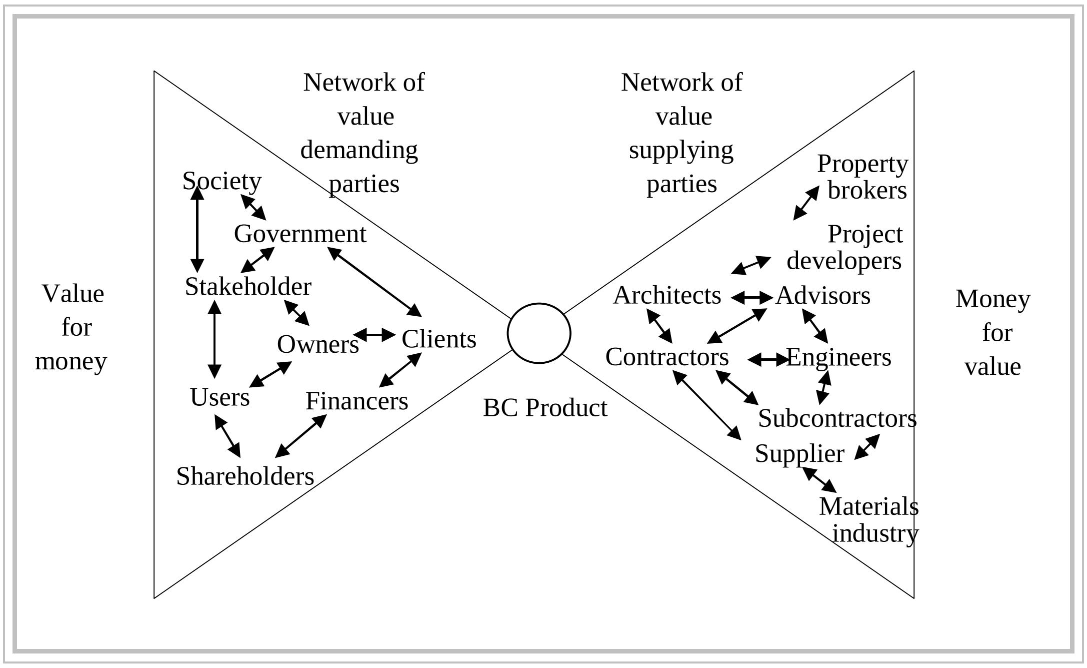


Figure 2-3 Network of value demanding and value supplying parties in the BC industry [Vrijhoef and De Ridder 2003].

In general, all value demanding parties can be regarded as the client of the BC Industry. However, not all demanding parties may be actively involved in the project. In this thesis, the client is defined as the value demanding party who is actively involved in the building process.

2.1.2 The Building Process

The primary interest of the client is to optimise the difference between value and price. This difference should be acceptable by and affordable for the client (Figure 2-4). The primary interest of the value supplying parties is to optimise the profit. Consequently the client has a different primary interests than the supplying parties which may lead to conflicts.

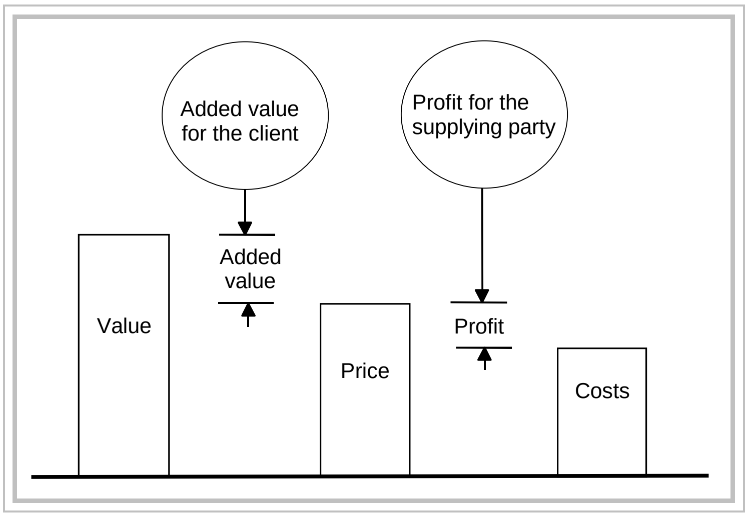


Figure 2-4 Added value in the Value-Price-Cost model.

The client asks for value for money from the value supplying parties using a demand (such as a client's brief or agreeing on services from an expert). The value supplying parties try to match this demand with their products and services (supply) for a certain price and costs [De Ridder, 2002] (Figure 2-5).

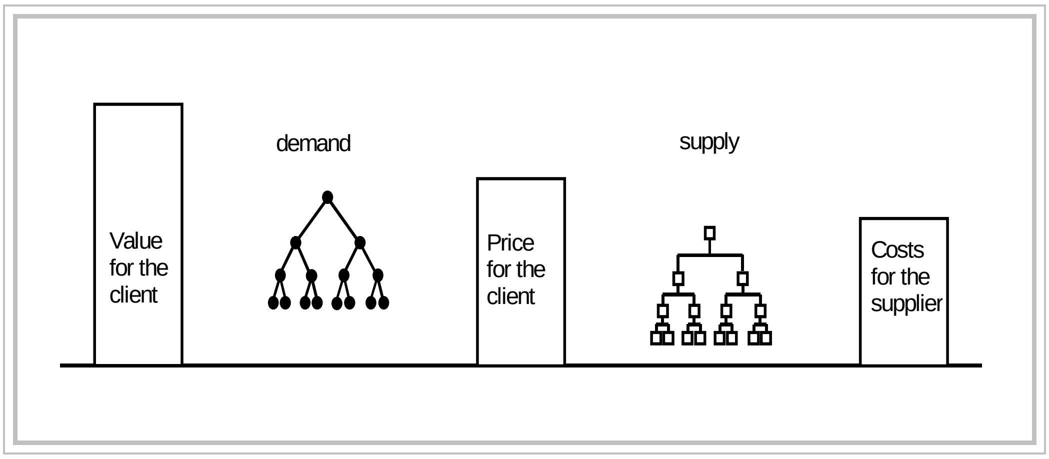


Figure 2-5 Demand and supply in the Value-Price-Cost model [De Ridder, 2002].

During the building process, the client continuously attempts to optimise the ‘added value’ by his demand. The value supplier influences the building process to optimise the profit by his supply. To reach a satisfying situation for both parties, the demand and the supply must be matched. Finding an optimal match is a continuous process where client and the supplier exchange information (Figure 2-6).

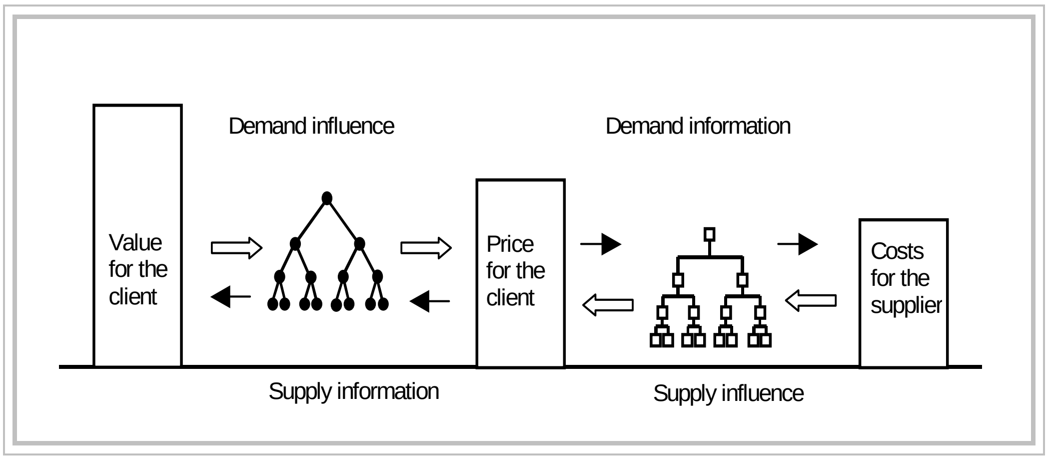


Figure 2-6 The building process (after De Ridder, 2001).

2.1.3 Dissatisfaction of the Client

Dissatisfaction of the client occurs when the client expects more ‘added value’ than he received. In this regard, the Value-Price-Cost model distinguishes three different scenarios [Ang & et al., 2003]. The first scenario occurs when the expected value is lower than the experienced value (Figure 2-7).

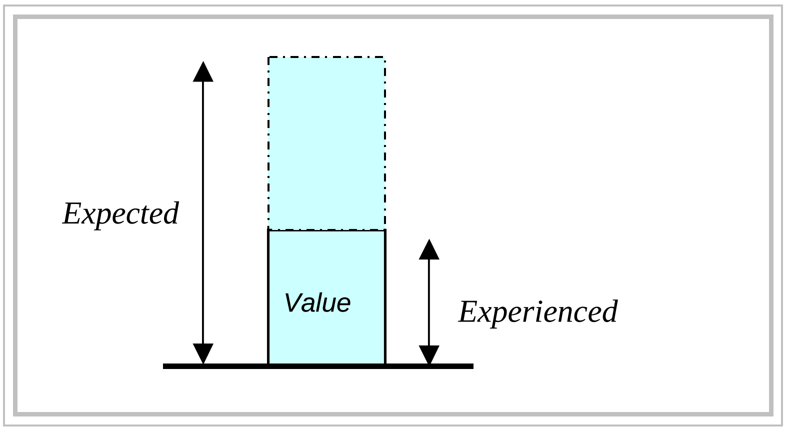


Figure 2-7 First scenario of the client's dissatisfaction.

The second scenario happens when the actual price of the product is higher than the expected price (Figure 2-8).

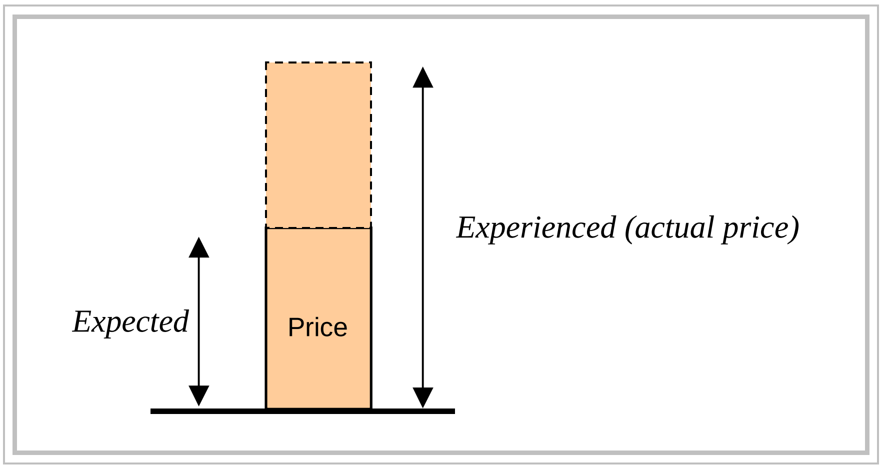


Figure 2-8 Second scenario of the client's dissatisfaction.

The third scenario is a hybrid of afore mentioned scenarios when the client experiences a lower value and a higher price. All three scenarios lead to an unexpected lower 'added value' for the client.

2.1.4 Risks

Figure 2-9 identifies the scope of the primary interest of the client and the primary interest of the supplier within the Value-Price-Cost model.

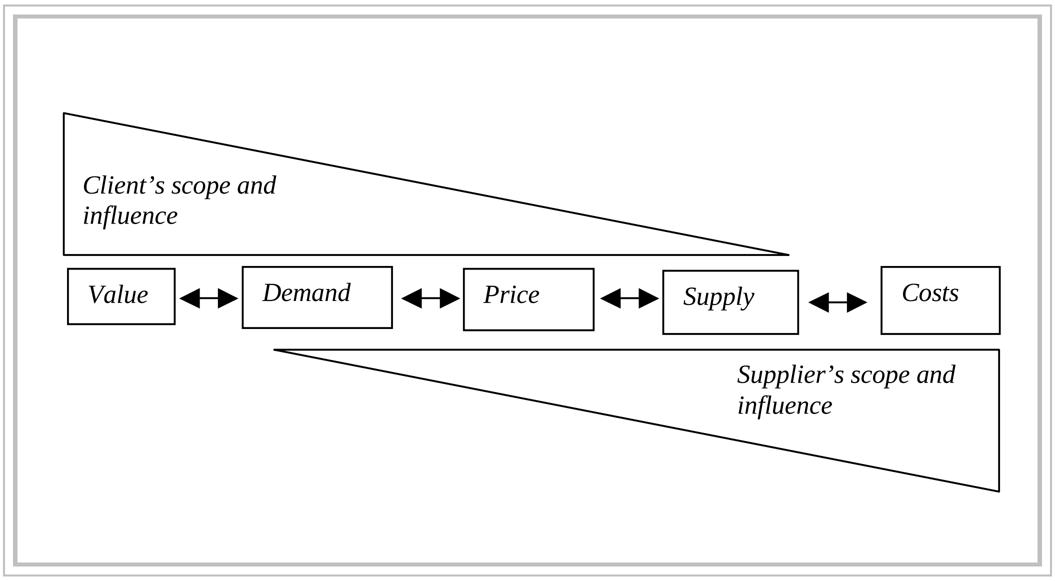


Figure 2-9 A comparison of the scope of the client and supplier.

Based on the scopes of the client and supplier, three different types of risks can be distinguished: value demand risks, transaction risks and production (supply) risks [Reniers & De Ridder, 2003] (Figure 2-10).

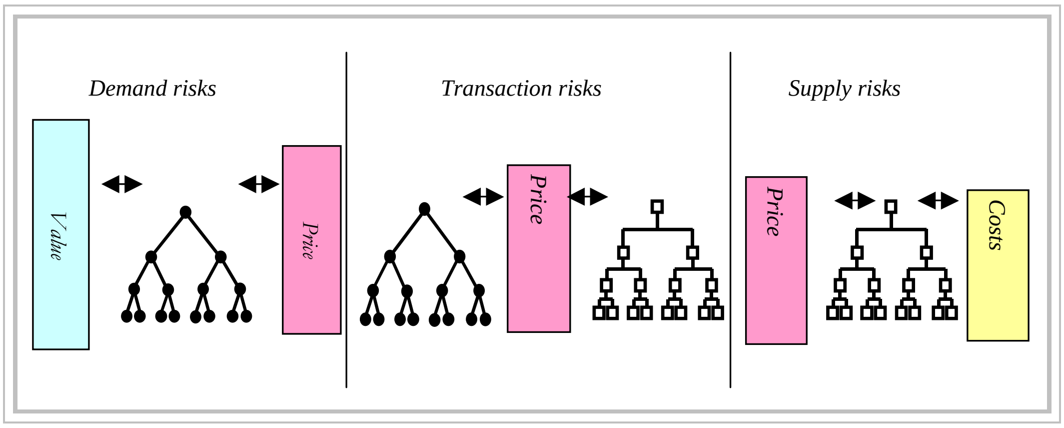


Figure 2-10 Types of risks within the Value-Price-Cost model: demand risks, transaction risks and supply risks.

The 'demand risks' that the client takes occur when the value and the price are not tuned. For example, the client runs the risks for specifying a demand that does not deliver the expected value.

The demanding party and the supplying party run 'transaction risks' when they are involved in any kind of transaction. When both parties have to co-operate, they run for example the risk of miscommunication or working inefficiently.

The supplying party runs 'supply risks' which are the risks involved when the price and costs are not tuned, for instance when extra costs are caused by the construction method employed. In general, risks threaten the profit or the 'added value' and therefore must be avoided or at least brought down to a minimum.

2.1.5 Conclusions

The client wants to optimise the difference between value and price in his favour by influencing the project with his demand. Therefore he runs demand and transactions risks. Both risks threaten the value and the price and therefore threaten the 'added value'.

2.2 RISKS THE CLIENT FACES DURING THE BUILDING PROCESS

2.2.1 Perception problems

The aim of the client is to change the current situation into a preferred situation [after Zeisel, 1981]. In this thesis, the demand of the client is defined as the difference between the current situation and the preferred situation including the transformation process (Figure 2-11).

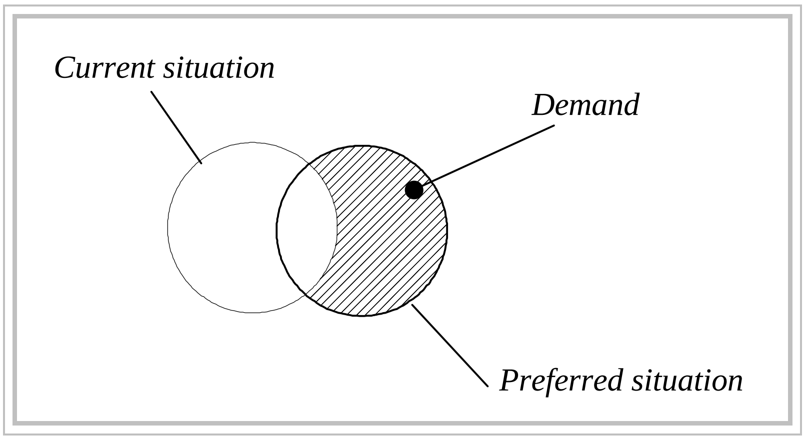


Figure 2-11 The client wants to change the current situation into a preferred situation.

To define the preferred situation, the client needs to have a clear (objective) view of both the current and the preferred situations. However, the client may have his own perception of these situations. When this perceived situation does not correspond with the current situation the client has a perception problem (Figure 2-12). An example of such a perception problem can exist when the demand doesn't match with the supply.

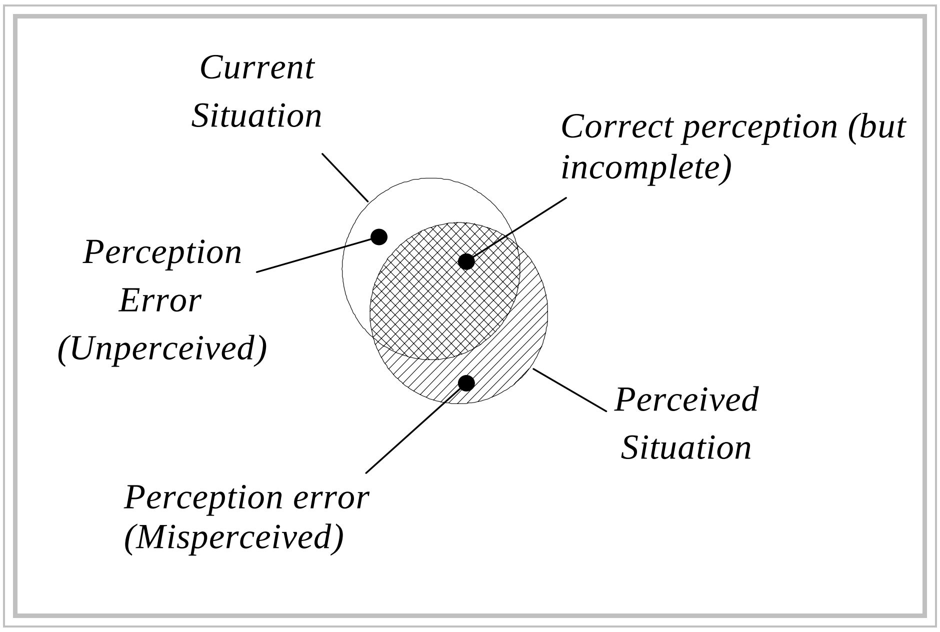


Figure 2-12 Perception problems occur when the perceived situation does not correspond with the current situation.

It is possible that the client perceives both the current situation and the preferred situation incorrectly. Due to these perception problems the client specifies a 'perceived' demand that differs from the 'actual' demand (Figure 2-13).

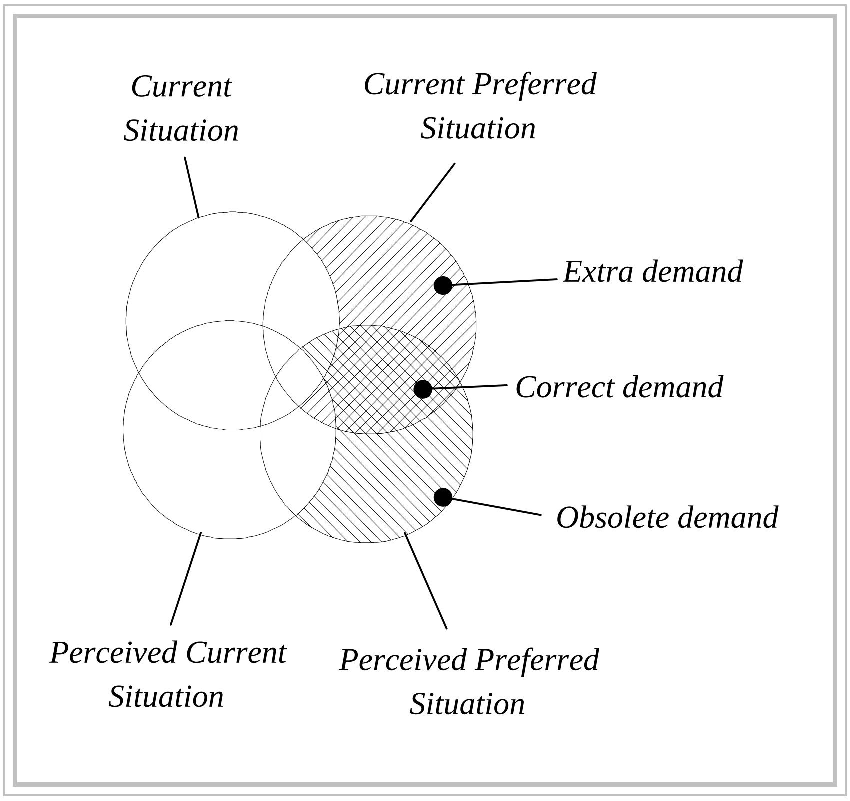


Figure 2-13 Due to perception problems the client specifies an incorrect demand.

2.2.2 Changing Demand during the Building Process

Gielingh [1988] argues that the development of the demand is primarily a top down process. This means that products are designed by first specifying their functionality as a whole. Afterwards, feedback on possible solutions is presented in order to match the demand. During the Building process, the client also receives feedback such as information related to the actual and desired situation. With this feedback the client is able to improve his perceptions on both the current and preferred situations which can result in demand changes (Figure 2-14).

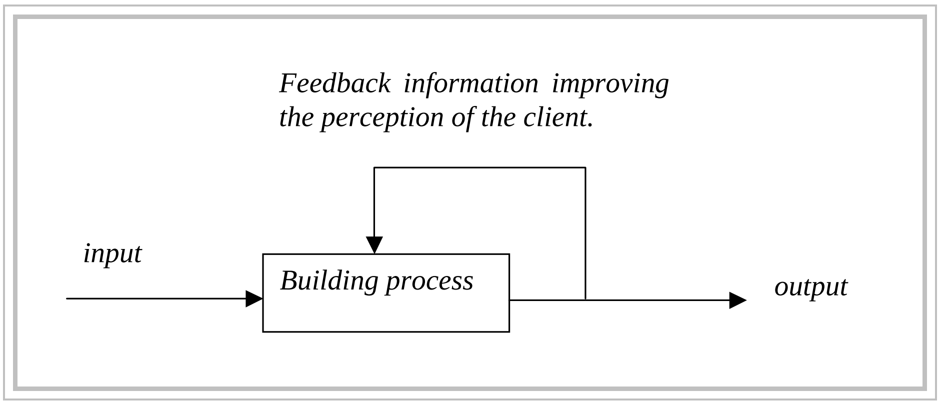


Figure 2-14 During the building process the client receives feedback information enabling him to improve his perceptions.

Consequently two tasks can be distinguished for transforming the current situation into the preferred situation:

1. The perceived current situation can be transformed into the perceived preferred situation (to attempt to solve the problem)
2. The perceived situations can be improved by reducing perception errors for both the current and the preferred situations (in order to attempt to solve perception errors)

The assumption is that the client is able to change his perception and consequently change his demand during the process (Figure 2-15).

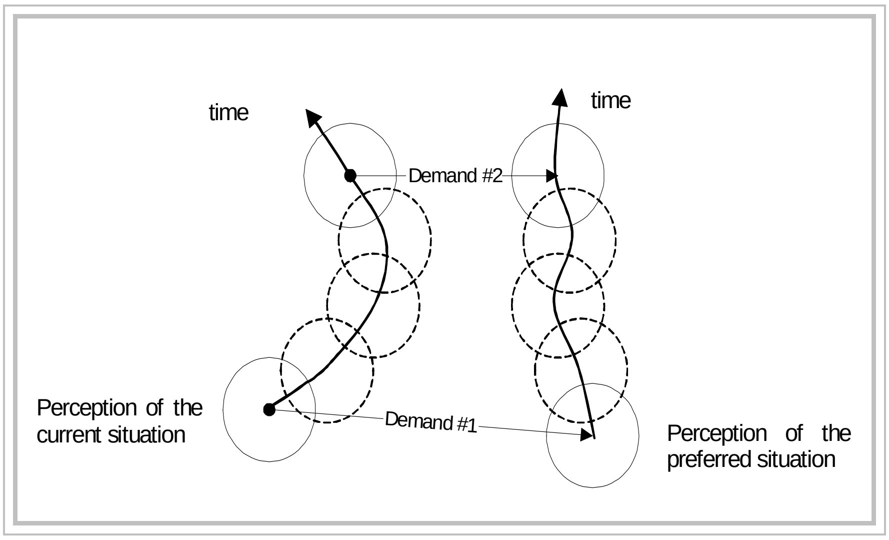


Figure 2-15 Due to changes in perception during the building process, the demand can change resulting in different influences of the client in the project.

The changing demand during the building process leads to extra costs while parts of the previously defined demand will become obsolete. Especially when contracts are sealed changing the demand can become very costly for the client (Figure 2-16).

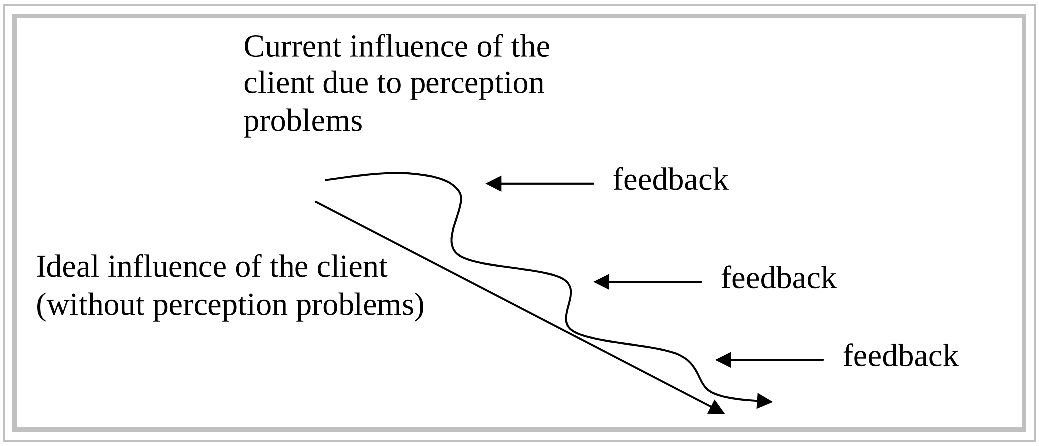


Figure 2-16 Current influence of the client versus an ideal influence.

Spekkink and van Wijk [1998] argue that initial decisions possibly have a large impact on the project (Figure 2-17). Therefore, incorrect initial demands may also have a large impact on the course of the project and can be costly to undo during later phases. Decisions taken in the early project phases, referred to as the inception phase of a project in this thesis, heavily determine the course of the project and influence the outcome of the project. However, the available information in this inception stage is usually very poor which makes it very difficult for the client to diminish potential perception problems.

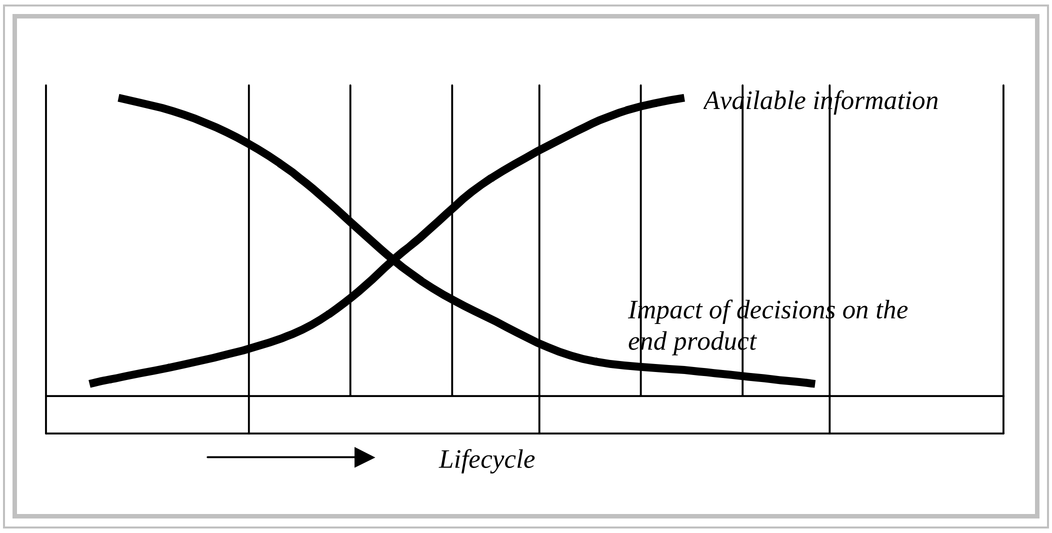


Figure 2-17 The impact of decisions on the end product and the availability of information during the product lifecycle [Spekkink & van Wijk, 1998].

2.2.3 Conclusions

Because the client does not always have a clear perception of the current and perceived situations, he runs the risk that the demand will not provide him with the expected added value. During the project, the client can change his perception due to an increasing availability of relevant information. The changed perceptions create new demands leading to extra costs while parts of the previously defined demand will become obsolete. Especially during the inception phase, the client may run high risks because relevant information is not available or not sufficient yet.

2.3 KNOWLEDGE SUPPORT FOR THE CLIENT IN THE INCEPTION PHASE

2.3.1 Knowledge Position of the Client

A professional client can be defined as a client who has been involved in several projects and therefore has gained a lot of experience in effectively participating in the building process. The knowledge position of the client refers to the client's degree of experience regarding his previous involvements in the building process. However, every project in the BC industry has unique characteristics, involving different partners, contracts, requirements and designs. Making correct decisions (demand) for the client without extra knowledge is therefore very difficult. The consumption of extra knowledge will help the (professional) client to oversee all the consequences of the actions he undertakes. Evidently, when the client is not professional, he will experience more difficulty in overseeing the consequences of his decisions [Dullaart & Hoeth, 1998] and consequently will change his demand during the building process.

2.3.2 Necessary Knowledge for the Client in the Inception Phase

The type of knowledge explicitly required for the client depends on various aspects such as the project type, contract type and knowledge position (the degree of knowledge the client has to successfully influence the building process) of the client. However, the client needs to have an understanding of what is most valuable and relevant for him. He is also required to have an understanding of the possibilities of the supply as well as all transactions. Hence, the client needs access to Demand, Supply and Transaction related knowledge.

Demand related knowledge is necessary for relating value in the Value-Price-Cost model to the demand such as:

- Knowledge about the current and preferred situations
- Knowledge about the goals and ambitions
- Knowledge about the functions of the product
- Knowledge about the primary process of the client and the expected future developments of these primary processes
- Knowledge about user requirements and requirements of other stakeholders
- Knowledge about legislations and regulations

To ensure that the client's demand can be met, he needs to have an understanding of possible solutions and subsequently he may need to have access to Supply related knowledge such as:

- Knowledge about the solution space, i.e. knowledge about the type of solutions that can meet the demand
- Knowledge about the characteristics of individual solutions, i.e. knowledge about the performances of each solution
- Knowledge about the total costs of ownership and the lifecycle costs

As the client has to interact with suppliers, he may need to have access to Transaction related knowledge such as:

- Knowledge about contract types
- Knowledge about risks and responsibilities
- Knowledge about specifications
- Knowledge about financing

The client does not need to become a Building and Construction expert nor does he want to be such an expert. Therefore, he needs to be guarded against any unnecessary knowledge.

2.3.3 Experts Supporting the Client

Experts can help the client in developing his demand. Yet, experts do not always support the client sufficiently because:

- Experts may be involved too late in the process and therefore are not able to influence earlier decisions [Rutten & Trum, 2000]
- Experts usually have domain specific knowledge, which is followed from the fragmentation of the BC industry. Particularly, it is difficult during the

inception phase to determine the necessary type of expertise for specific demands

- The client and experts cannot always communicate effectively [Rezgui et all, 2001]
- Experts may have a tendency at rushing towards (design) solutions without performing a proper analysis of the client's requirements and needs [Van Leusen, 1994]
- Attracting experts may be costly.

2.3.4 The Use of the Existing Body of Construction Knowledge

An enormous amount of knowledge exists in documents (books, reports, formulas etc), regulations and codes of practice. In addition, a great deal of knowledge exists as experience in the heads of experts. It can be observed that the Body of Construction Knowledge is fragmented and is mainly dedicated to the needs of domain experts. Therefore, the client can hardly benefit from the existing Body of Construction Knowledge. For instance, the Body of Construction Knowledge required for predicting the behaviour of a certain product is mostly available but is hardly used in projects.

2.3.5 Conclusions

The client has a weak position because he dependents on knowledge support. Experts are not always able to provide the appropriate knowledge support for the client in the inception phase. Furthermore, the client often does not seek assistance of a particular expert when formulating demands as long as he is not certain of the right course of action that he has to take.

2.4 SCOPE OF THIS RESEARCH

2.4.1 As-Is Situation

Because of the weak knowledge position of the client, he runs the risks of specifying an incorrect demand. Particularly the non-professional client is often unable to formulate his demand properly. In addition, the 'value supplying parties' have problems with matching the demand and the supply when for example presenting the characteristics of their supply. This also contributes to the unsatisfactory current situation for the client who does not have access to the extensive Body of Construction Knowledge that is available today (Figure 2-18).

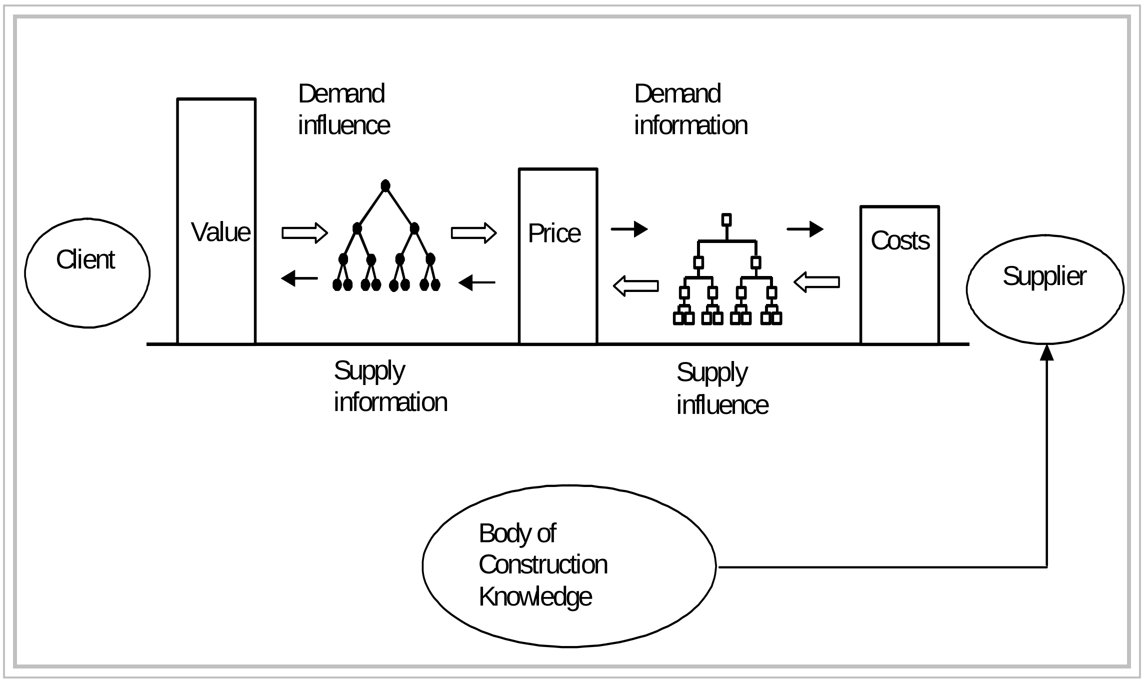


Figure 2-18 The As-Is situation of the current position of the client.

2.4.2 Related Topics

To improve the current situation in the BC industry the following solutions are available.

Dynamic Control

The Ridder argues that the client should be able to make changes in his demand during the building process. Therefore, agreements are necessary for all parties involved in order to match the value and the price of the product [De Ridder, 2002].

Functional Specifications

By functional specifications, the client does not specify a detailed product but rather the performance specifications of the product. This allows the BC industry to match the demand with the supply [van den Brand and Nijland, 1999].

Design Sessions

Design sessions are a relatively new form of collaboration and can be used in the inception phase. The idea is to ensure that all relevant expertise is available in meetings for an integrated approach. In this regard, process management is also necessary for the project to benefit from the available expertises [Friedl et al., 2001].

2.4.3 Focus of this Research

The research hypothesis of this research suggests that it could be helpful to provide the client with access to (a part of) the available Body of Construction Knowledge. In theory, this can lead to a more accurately formulated demand, which incorporates a better match of the demand and the supply, as well as the occurrence of fewer problems caused by unforeseen changes in the demand. Of course, the ‘value supplying parties’ could also benefit from an improved access to the Body of Construction Knowledge. The latter is out of scope of this research.

This research attempts to provide support for the client using Information and Communication Technology (ICT). ICT can help to find the right information and knowledge and is helpful in adapting and presenting the information and knowledge in the right (i.e. understandable) form or format (i.e. as input for a computer application). Therefore, the client should be provided with an ICT-enabled access to the Body of Construction Knowledge (Figure 2-19).

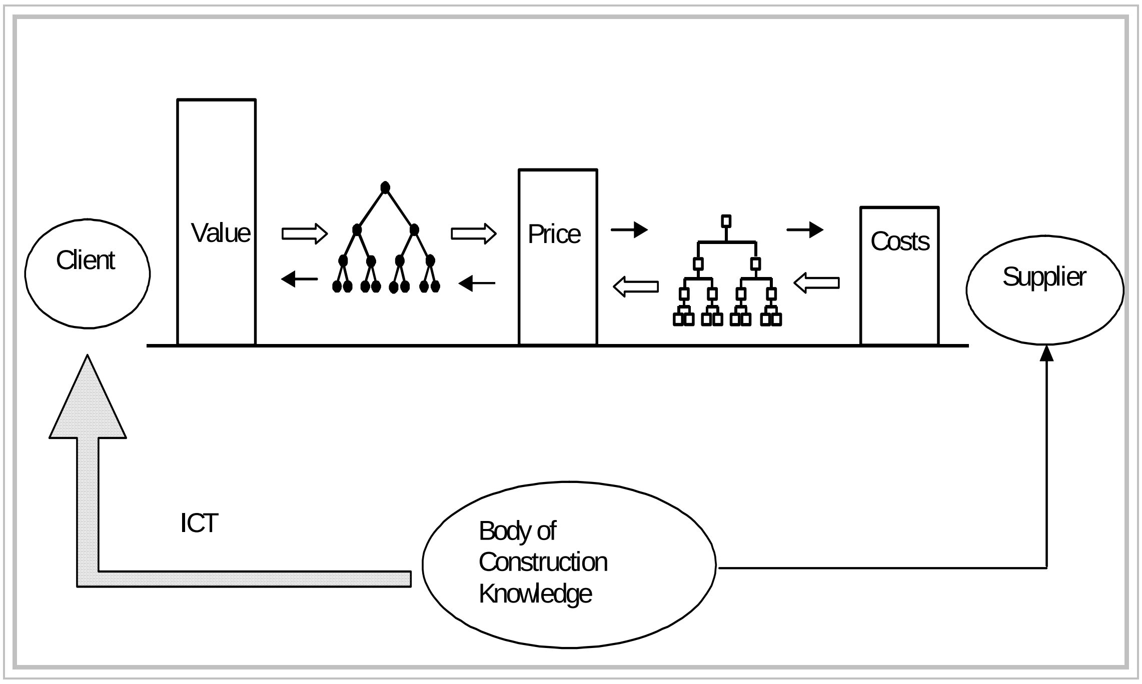


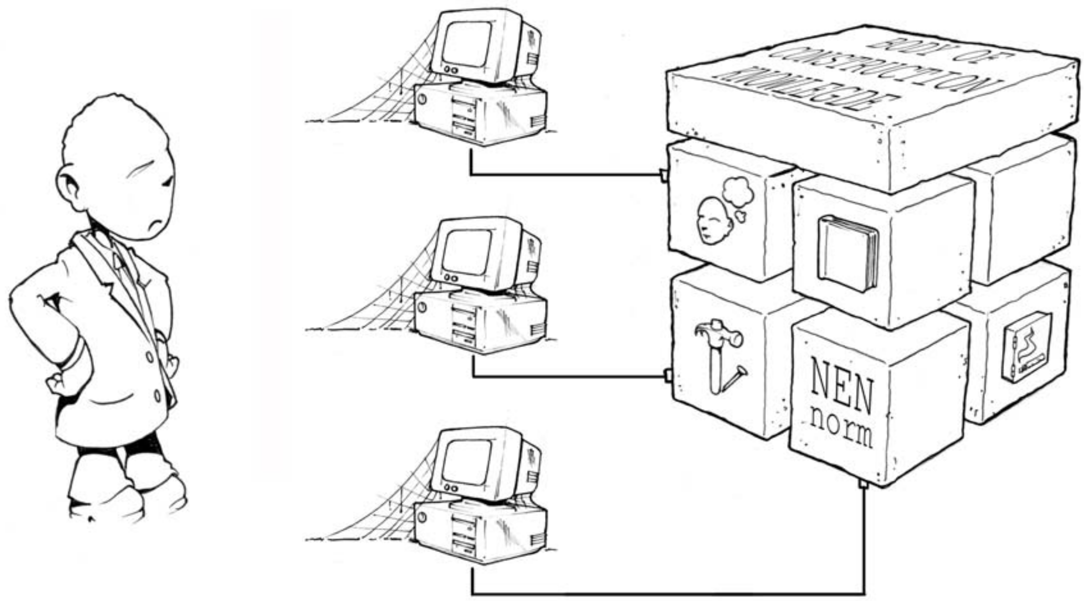
Figure 2-19 The To-Be situation. The large arrow identifies the problem area of this research.

If available in the right form, early access to the Body of Construction Knowledge could help to reduce the risks. Therefore, the focus of this research is on the inception phase where important decisions are taken and useful information is scarce.

3

Analysis of ICT-Enabled Access to the Body of Construction Knowledge

This chapter discusses the suitability of existing software applications for the client in order to access the Body of Construction Knowledge. Furthermore, a number of research projects are investigated that have been specifically engaged in issues related to the client's access to the Body of Construction Knowledge.



The client lacks a suitable ICT-enabled access to the fragmented Body of Construction Knowledge.

3.1 INTRODUCTION

The previous chapter argued that it could be helpful to provide the client with ICT-enabled access to the Body of Construction Knowledge, particularly during the inception phase.
This chapter discusses the suitability of software applications for the client in order to access:

- Demand related knowledge
- Supply related knowledge
- Transaction related knowledge

This chapter is particularly interested in the reasoning capabilities of software applications for accessing the Body of Construction Knowledge. In this regard, a document management system is out of scope of this research. Such a system is not able to reason about the information contained in a document and therefore can neither provide the client with custom-made support on the project nor can it shield the client against information overload.

3.2 ICT-ENABLED ACCESS TO DEMAND RELATED KNOWLEDGE

The client needs to have a good understanding of the current and the preferred situations in order to be able to specify a demand that will bring the expected added value. In addition, the client needs to have an understanding of the demand of other stakeholders like user requirements, requirements from society like sustainability and rules and regulations, etc. Software applications are required in order to facilitate the client's ICT-enabled access to demand related knowledge. This section analyses some of these software applications.

3.2.1 Requirements Engineering Tools

Requirements Engineering (RE) is a discipline concerned with the development of the requirements. Processes and software applications available address the following problems associated with defining requirements:

- Requirements are not well understood
- Requirements can be incomplete
- Requirements may specify unnecessary functionality
- Requirements can be vague, uncertain and/or ambiguous

A large number of Requirement Engineering tools are available like DOORS and Catalyze. Currently, most RE tools use an Object Oriented approach to capture and organise requirements. Each requirement is then modelled using an object, which can be linked to other objects (requirements). Properties can be associated to these objects describing a requirement in more detail. Based on such an object model, configuration management, publication of requirements on the web, collaboration support and prioritisation functionality can be offered.

Evaluation

Presently, many different RE tools are available offering a wide range of functionality. However, the captured requirements are currently not computer interpretable because of the unavailability of reasoning functionality in such applications. Consequently, these applications hardly provide the client with access to the Body of Construction Knowledge.

3.2.2 Norm Worm

The Norm Worm is an application focusing on conformance checking with the Dutch building design rules. The user has to provide input information of the building like the areas of building, the function of the area, information related to building components (materials and such) and geometric information (manually or semi-automatically by using CAD drawings). All information is stored in one computer interpretable model in order to be used by several applications. These applications use the data in the model to check the design against several rules and regulations. For example, the Norm Worm is able to check building designs on daylight, energy performance and ventilation regulations. An automatically created report of the results of these checks can be used when applying for a building permission.

Evaluation

Although, a very small part of the building code has been implemented in the Norm Worm, several actors in the building process like architects, local governments, et cetera found it to be useful. The model used in the system is not an open model and therefore it is difficult to extend, modify, or use it for other purposes.

3.2.3 Solibri Model Checker

The Solibri Model Checker (SMC) exposes flaws and problems in a building information model. The SMC uses a building information model (the Industry Foundation Classes). Based on this model, several checks can be performed automatically such as checking the interference of components (clash detection) or checking the escape routes. In addition, the SMC users can modify constraints used by the system.

Evaluation

The SMC provides a platform where constraints can be imposed on the building information model allowing various checks. Currently, only a few checks have been implemented that can be performed automatically.

3.2.4 Conclusions

This section has analysed only a few applications that provide access to demand related knowledge. Except of Requirement Engineering software, only a limited number of applications are available for accessing demand related knowledge. The Requirement Engineering applications are the most developed of the available software applications but only supports a requirement elicitation process and can hardly provide access to demand related knowledge.

3.3 ICT-ENABLED ACCESS TO SUPPLY RELATED KNOWLEDGE

The Construction Industry Computing Association (CICA) lists approximately 1800 software applications that can be used to support the BC industry from the inception phase of the design through the construction process extending to the management and maintenance of properties. This section analyses software applications that provide ICT-enabled access to supply related knowledge.

3.3.1 Parametric Design Tools

Parametric data describes information about geometry and topology of objects such as height, depth, amount of floors, et cetera. Parametric design refers to the application of parametric data embedded within 3D objects. The available parametric design tools are capable of generating complete designs. These tools are mostly custom-built systems using existing CAD software that are also dedicated to the design of specific products. These systems allow users to specify a design based on a set of parameters. Most of these systems are product specific and are able to generate alternative solutions for the same product type. In this regard, the system generates a design using predefined design information. A great amount of assumptions is embedded into such systems, hence they are not transparent to the user. Despite the latter, the user can generate complete alternative design solutions. Figure 3-1 shows a screenshot of a parametric design application. In this example, the application can generate alternative viaducts based on three user-defined parameters.

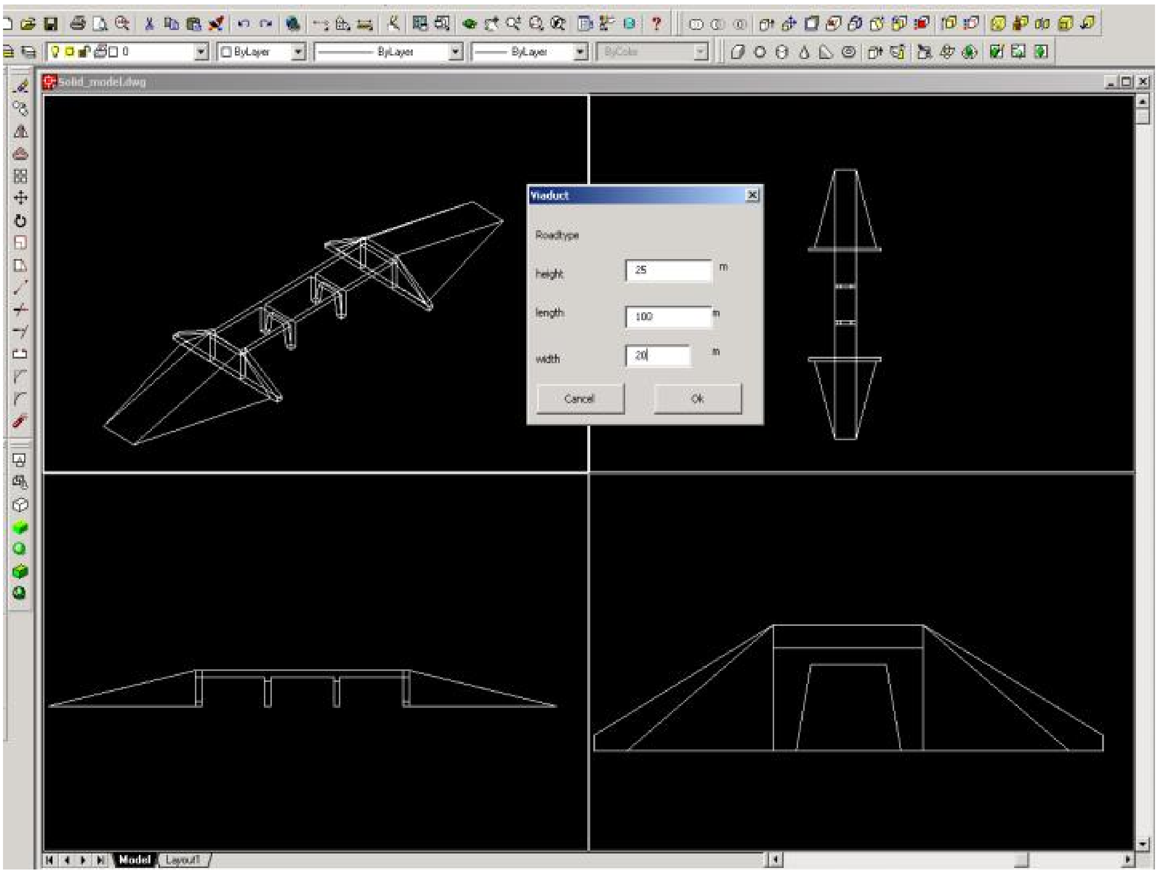


Figure 3-1 Screenshot of an application for parametric design of viaducts.

Evaluation

Parametric design applications support the design of products based on some predefined parameters. In effect, parametric design prevents repetitive design works while providing a certain degree of freedom providing the client with fast feedback on alternative design solutions.

A disadvantage is that other design alternatives requiring other parameters than the pre-defined ones are not possible because such systems operate within a fixed level of detail. Therefore, these systems may quickly become too rigid. Furthermore, it is difficult to retrieve the design rationale. Therefore, it is difficult to modify the knowledge in the system and the decisions made are not always well documented. Communication with these systems can be very client friendly. The parameters can be defined in terms that the client can understand. In addition, due to the fast feedback and the visualisation of the design alternatives, the client is able to experiment with the value of the parameters and can study the consequences associated with them.

3.3.2 Computer Aided Design

Designers use CAD systems to document design information. Several CAD systems support a Building Information Model (BIM). ArchiCAD and Architectural Desktop, for example, support the parametric creation of walls, beams, and floors. These objects adjust themselves to changes in the model. For example, changing the height of the walls will change the location of the roof automatically. Furthermore, the BIM enables features like sunlight evaluations, or the generation of bills of quantities. Probably, more features will be incorporated in such systems in the future.

Evaluation

CAD systems do not offer complete design solutions but support experts to document their own design. These systems do not provide access to supply related knowledge and therefore cannot support the client. Noticeably the BIM is able to cope with changes in the design. However, the BIM is not useful during the inception phase because the BIM uses elements such as beams and columns, which are not relevant for the inception phase.

3.3.3 Design Evaluation Tools for Engineers

A great deal of design evaluation tools is available such as programs for evaluating ventilation systems, energy consumption, fire safety, lighting and acoustic, structural safety, et cetera. Most of these applications address only one aspect and/or type of a design solution. Experts use evaluation software applications for analysing and understanding the complex behaviour of the designed product. These applications provide access to overlapping parts of the Body of Construction Knowledge. For example applications for estimating heat-loss and cooling load like DOE-2 and TRNSYS, while having a similar scope, they are slightly different regarding their applicability, certainty, correctness and internal working.

Evaluation

Most design evaluation tools have a strong mono-disciplinary character. Their focus lies primarily on the detailed design phase. Furthermore, most of these tools use a fixed level of detail. Consequently, these tools need complete and coherent design information in order to be used. These programs use domain specific terms to support experts, which make them unsuitable for the client.
In addition, these programs differ in the level of accuracy and methods used as well as their applicability in different situations. Only the *interpretation* of the output of these programs by an expert can be interesting for the client. These applications need detailed information, and therefore are not relevant to the inception phase.

3.3.4 Conclusions

A great deal of supply related knowledge is captured by a diversity of software applications that operate on different project types, life cycle stages, level of detail, et cetera. This knowledge is however fragmented and not harmonised due to fragmentation of the BC industry itself. This means that either knowledge is not available or it exists in variations depending on the focus of the application. In addition, this is highly specific knowledge for experts and hardly accessible for (inexperienced) clients.
Furthermore, the client during the inception phase is only able to provide information that is vague, incoherent, incomplete, conflicting, et cetera. Existing applications can only deal with a fixed level of detail and are therefore not suitable for the client during the inception phase.

In addition, the client cannot communicate with these tools because they are highly specific and expert oriented.

3.4 ICT-ENABLED ACCESS TO TRANSACTION RELATED KNOWLEDGE

The client needs to have sufficient knowledge about how to deal with transactions. For example, the client needs knowledge about, building specifications, contract types, risks and responsibilities, selection of suppliers, et cetera. This section analyses existing applications that provide access to transaction related knowledge.

3.4.1 ICT-Enabled Access to Transaction Related Regulations

Several websites give access to regulations regarding transactions between the client and suppliers. For example, the Dutch ministry for Spatial Planning, Housing and the Environment [VROM] uses their website (www.vrom.nl) to give its users access to knowledge about building applications forms. Users of the website can retrieve the correct application form by answering questions about their project. Similarly, the Eurasbo website (www.eurasbo.com) gives its users access to a part of the European tendering rules. Multiple-choice questions determine the applicable regulations and gives access to relevant procedures that should be followed.

Evaluation

The advantage of this type accessibility is that the knowledge provider can coordinate all changes to the information and/or knowledge. A pre-requisite is that the client needs to be aware of the existence of these sources.

3.4.2 Cost Estimation Software

Several cost estimation software applications are available like 'Kraan Bouw Plus', or Estimation Desktop (EDT). Most software tools use product elements to estimate the price of the product and offer a database with price information per element. Using a CAD application that supports a building information model, building elements can be extracted and related to so-called cost recipes. These cost recipes contain cost estimations like labour costs and material costs per element. By using the elements derived from a building information model, automated cost estimations are possible.

Evaluation

Although automated estimations of the costs is probably interesting for the client, this may not be possible during the inception phase when using estimations based on elements because of the unavailability of detailed information on the product.

3.4.3 Kubus STABU

Kubus STABU is an application that supports the Dutch STABU system for writing construction documents (building specifications), offering a database of predefined elements. Users of the system can browse through the indexed catalogue of elements and can select these elements for the project at hand. Furthermore, the system supports a file format enabling others to see the construction documents by using a simple viewer.

Evaluation

Experts can use Kubus STABU for faster and consistent preparation of construction documents. The software is not suitable for the client. Nevertheless, the program can facilitate communication with the client.

3.4.4 Conclusions

Applications that provide access to 'transaction related knowledge' are scarce. For example, applications for accessing knowledge related to contract types, or selecting experts is not available. Reasoning on regulations is available but only on a limited scale. Using web services for reasoning enables organisations to up-date their own application whenever it is necessary. However, the client may not know about the existence of these websites and consequently needs a more proactive knowledge support. Cost estimation and construction documents related software offer too much detail and can only be used by experts.

3.5 RELATED RESEARCH

Supporting the client in the inception phase of BC projects is a relatively new research topic. Therefore, there are not many research projects in this area. This section analyses a number of research projects attempting to support the client in the inception phase by providing ICT-enabled access to the Body of Construction Knowledge.

3.5.1 Parap

Parap is a software tool, which has been developed to support the Dutch Governmental Building Agency (RGD) to analyse their office buildings. Parap uses a relatively small set of parameters for modelling an office building in order to give multi-disciplinary feedback. Parap distinguishes three sets of parameters, namely: organisational parameters, location related parameters and building parameters. Based on these parameters, Parap is able to give feedback on several aspects like construction costs, maintenance costs, energy consumption, etc., using indexes that are derived from the study of previous projects.

Evaluation

Parap can be used to support the client in the inception phase because of the user-friendly interface and the fast multi-disciplinary feedback. Manipulation beyond the parameters is however not possible. Currently Parap does not support any data exchange mechanisms to re-use the information in other software tools. Furthermore, Parap cannot be consulted by other applications.

3.5.2 Concur

The European Concur project (Concurrent Design and Engineering in Building and Civil Engineering) 'demonstrates concurrent design and engineering focusing on electronic information exchange from the inception (client brief) stage to tendering and construction planning stage' [CONCUR]. The project addressed the integrated use of information and software applications using Product Data Technology (PDT). The main goal of the Concur project was to develop a two-way information exchange mechanism that allows actors to handle project information in the first stages of a construction project. This means that the information covers the relevant aspects and detail levels spanning the early stages of a project life cycle from inception to tender (Figure 3-2).

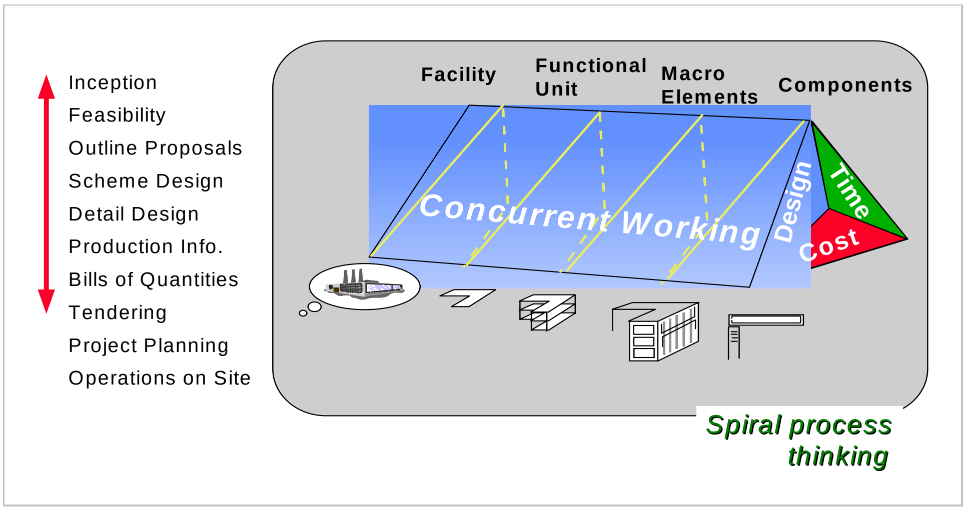


Figure 3-2 Spiral thinking in the Concur project.

The project developed, implemented and demonstrated a three level architecture used by Fortum Engineering, a Finish vendor of turnkey power plants. The three levels could communicate information (such as concept designs) in both directions. For the knowledge integration aspect, a lot of effort has been put into the development of well-structured knowledge bases [Ozsariyildiz, n.d.]. Regarding the ontology aspect, it is interesting to note that the development of STABU's LexiCon started in Concur.

Evaluation

The Concur project demonstrated the successful application of a three level inception-to-tender support environment. The environment supported a 'level of detail' approach for inception that naturally follows the design process and used 3D visualisation for representing concept designs. The client of Fortum Engineering is better off because the Concur environment offers, after a number of cycles, a near-optimum power plant concept.

3.5.3 SEED

The SEED project is a software environment to support early design phases in building design. The goal of this project is to provide computer support for the preliminary design phase of buildings such as visualisation, analysis and evaluation and the generation of design alternatives. The SEED project argues that existing software is unable to support exploration activities of early design phases, i.e. the fast generation of alternative design concepts and their rapid evaluation against a broad spectrum of relevant and possibly conflicting criteria. The SEED environment consists of several applications supporting several aspects of the design exploration. For example, 'SEED Layout' supports designers in generating a space layout plan. Another application is 'SEED Database', which tries to capture snapshots of the design communication for tracking the design process. The latter can also be used for case based reasoning.

Evaluation

The SEED environment focuses on supporting the architect. One important aspect for an architect is communication with the client. In this regard, SEED can offer interesting possibilities for supporting the client for instance the fast generation of design alternatives.

3.5.4 Design Assistant

A lot of design related knowledge is documented in books such as "Bauentwurfslehre (Architect's data)" from Neufert [2000]. This book contains a set of precedents for different architectural domains like residential buildings, sport facilities and office buildings. The precedents have been generalised into types, which offer a blue print for a design. Achten [1997] argues that the building type is a knowledge structure that is recognised as an important element in the architectural design process. During a master thesis, a research is carried out to provide ICT-enabled access to this type of knowledge. Van der Wel [2002] developed, in this thesis, a prototype implementation called Design Assistant, which uses this knowledge to automate design tasks enabling the creation of a design, based on a building type (Figure 3-3).

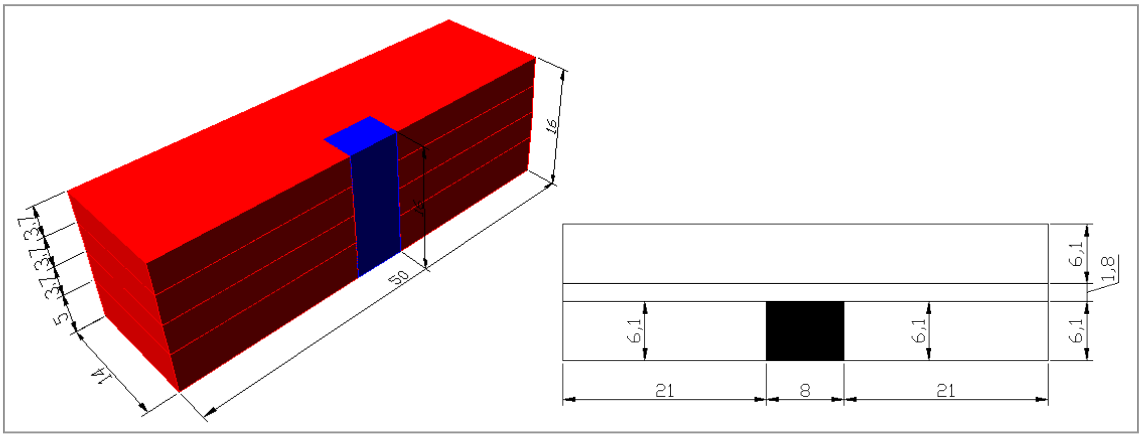


Figure 3-3 The Design Assistant uses a building type to create a design.

The system offers the user the choice of several building types and the choice of several levels of detail (e.g. building level, floor level, and room level). Not only building types, in an architectural sense, are accessible but also load-bearing structures have been addressed in the same way. The user can choose several types of load bearing structures (Figure 3-4). The system is able to automate the design task in order to generate a suitable load-bearing structure into the already available building solution.

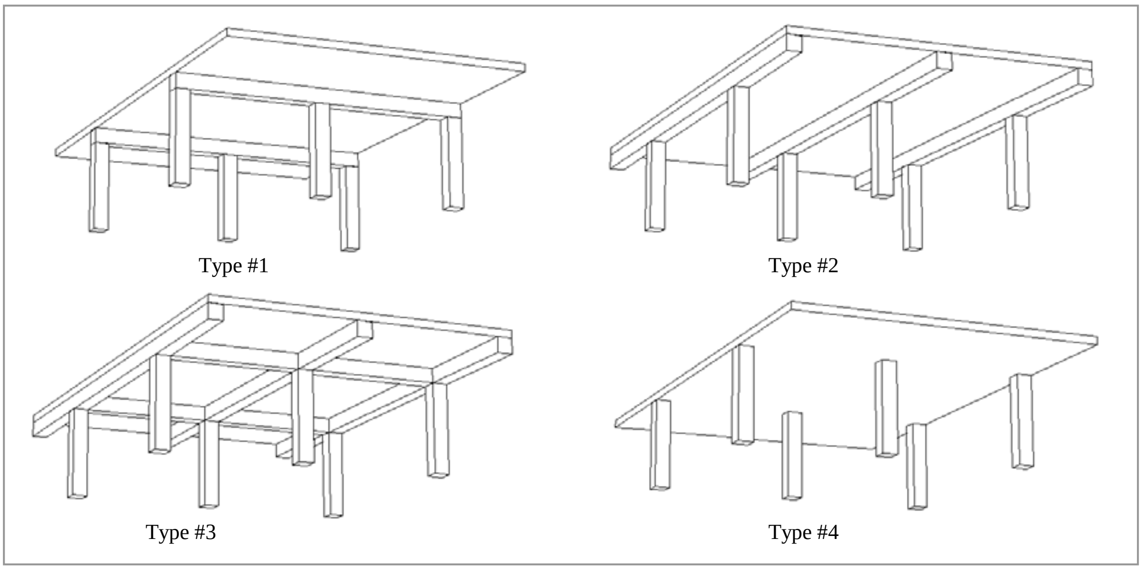


Figure 3-4 The Design Assistant offers several choices for structural solutions using 'types'.

Evaluation

Presenting different buildings types and different load-bearing types allows the user of the system to make important design decisions. Consequences of user-defined choices can then be visualised because the system is capable of automating the design process using requirements like the size or the number of storeys. A disadvantage of this application is that the knowledge is hidden in the code of the software and is difficult to extend or change without expert intervention.

3.5.5 Hospital Inception Support System

The Hospital Inception Support System (HISS) is developed to support early design processes of new hospital facilities [Schevers, 1999]. This system supports the creation and evaluation of different conceptual design solutions. In the HISS, different objects like buildings, parking lots, green areas and such can be placed in a 3D space. The user can input present and desired characteristics of these objects, such as the function of a building (Figure 3-5).

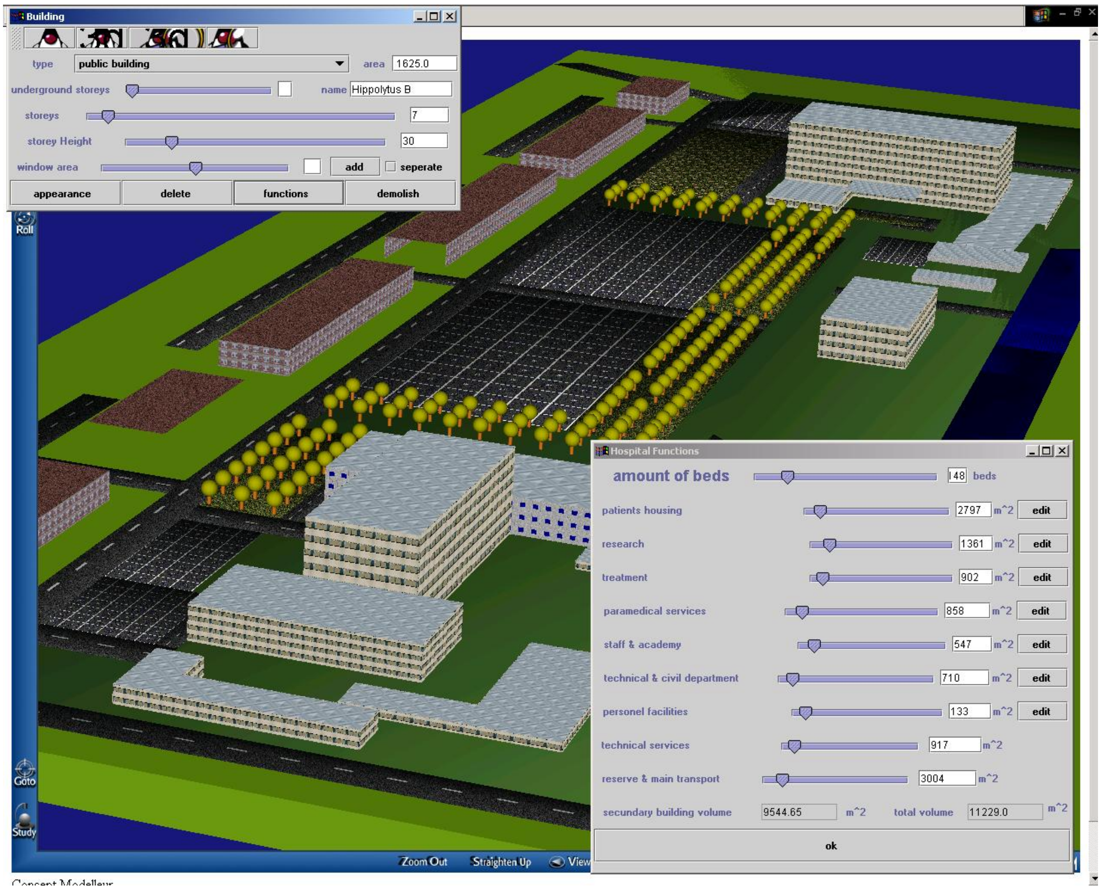


Figure 3-5 The existing situation of the "Reinier de Graaf" hospital in Delft including functions.

The idea is to model alternatives and to evaluate their properties such as costs, construction time, energy usage et cetera. Using parameters like length of the building and the amount of floors, the user can manipulate the size of the building. The system is able to guarantee the desired volume. Therefore, the system automatically calculates the number of stories required (Figure 3-6).

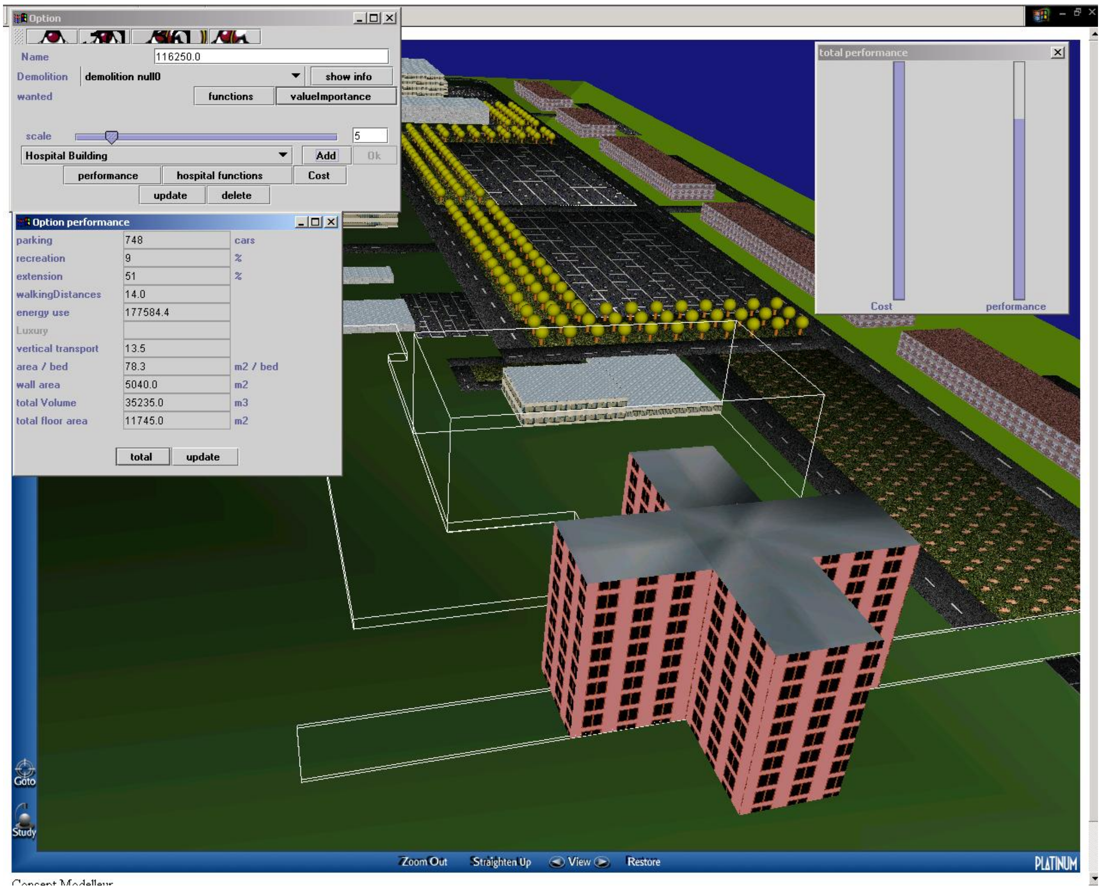


Figure 3-6 Characteristics of an alternative.

Different alternatives can be compared with each other based on their properties. In order to support the evaluation of multiple alternatives, all properties can be normalised and displayed. For the normalisation process, the client has to rate the properties

Evaluation

The hospital modeller supports a complete design cycle, i.e. development of the requirements, generation and evaluation of designs alternatives. The HISS supports the development of the requirements in a two-step strategy starting from the number of beds to more specific space requirements. The design itself is only supported on one level of detail. A disadvantage of the HISS is that the knowledge is hidden in the code of the software and consequently hard to extend or change.

3.5.6 Conclusions

Most research projects propose a 3D environment for interacting with and visualising design alternatives. Some of these projects deal with requirements that are interesting for the client in the inception phase. These research projects provide access to a small portion of the Body of Construction Knowledge.

3.6 CONCLUSIONS

Currently available applications do not adequately provide the client with access to the Body of Construction Knowledge during the inception phase:

- Only a small and specific part of the Body of Construction Knowledge can be accessed using existing applications. Extending or integrating the existing applications is difficult because most of these applications are closed systems.

- ICT-enabled access to the Body of Construction Knowledge is fragmented while the client requires access to an integrated Body of Construction Knowledge.

- Currently, no suitable mechanism is available for dealing with problems related to input information. Current applications only accept input information at a fixed level of detail. Information available during the inception phase can be characterised as scarce and ill-defined, i.e. incomplete, incoherent, conflicting, vague, uncertain and on different levels of detail. Most applications cannot be used in the inception phase because they require a specific set of (detailed) information in order to operate.

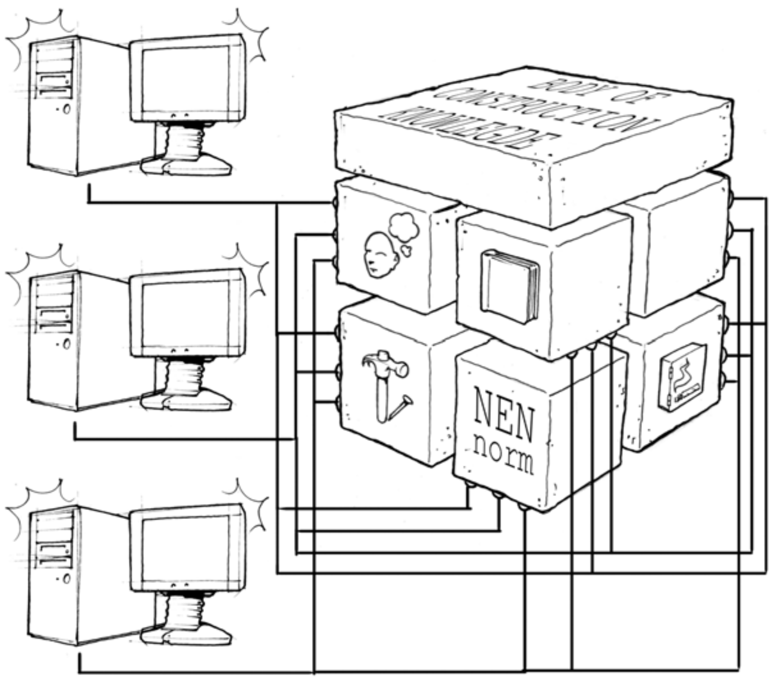
- Most existing applications are domain specific and are therefore less useful for the client.

- Knowledge is embedded in the software applications and is not visible/open for the client or other experts.

Research carried out in this field demonstrates the potential of ICT for supporting the client during the inception phase. For example, Virtual Reality and 3D presentations can be used to support communication with (inexperienced) clients. Rapid generation and presentation of design alternatives provides the client with the ability to evaluate the design alternative as well as the ability to redefine the requirements if necessary. In addition, research projects demonstrate the potentials of ICT-enabled access to a multi-disciplinary part of the Body of Construction Knowledge.

4 Analysis of State-of-the-Art Knowledge Technology for Accessing the Body of Construction Knowledge

This chapter analyses the state-of-the-art knowledge technology and investigates the suitability of this technology for providing the client with access to the Body of Construction Knowledge during the inception phase.



State-of-the-art Knowledge Technology offers numerous ways for reasoning on the Body of Construction Knowledge.

4.1 INTRODUCTION

This chapter attempts to analyses the state-of-the-art Knowledge Technology (KT) in order to be able to draw conclusions on the suitability of this technology for providing the client with access to the Body of Construction Knowledge (BoCK) during the inception phase. One problem is that the existing applications are only capable of reasoning on a small and specific part of the BoCK. Another problem is that existing applications are not able to reason with scarce and ill-defined information (incoherent, vague, conflicting, et cetera). This chapter investigates the suitability and applicability of Knowledge-Based Systems for the client.

Furthermore, an *integrated* access to the BoCK is necessary to support the client. As the BoCK is large and fragmented, multiple systems are necessary for reasoning purposes. The section 'state-of-the-art interoperability' analyses the ability of KT to provide a suitable infrastructure for integrating multiple applications in order to cope with:

- The unawareness of clients which parts of the BoCK they need
- The lack of skills for operating specialised software systems: usually clients are for example not able to interpret and integrate technical information

4.2 STATE-OF-THE-ART KNOWLEDGE-BASED SYSTEMS

Any application capable of reasoning can be regarded as a KBS [Guida, 1994]. However, the term Knowledge-Based System is mostly used for software systems that use a knowledge representation language for representing the knowledge and a problem-solving method, which is able to reason on the knowledge representation. Many different knowledge representations are available with their own characteristics and advantages for reasoning on a certain type of knowledge. Figure 4-1 roughly relates different types of knowledge representations to different types of knowledge. The axes denote respectively the data intensiveness and the clarity or complexity of the data.

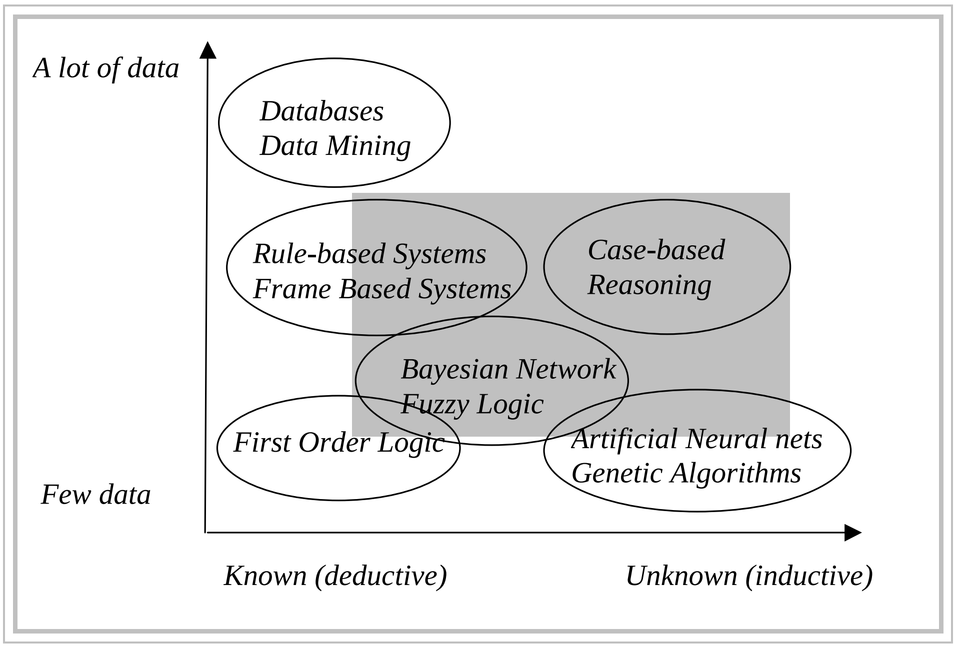


Figure 4-1 The applicability of knowledge representations.

The client is not able to specify his demand because the relationships between the client's values and product properties are not clear. The process to clarify these relationships during the inception phase requires knowledge that can be characterised as uncertain, incomplete, complex, vague and based on assumptions and experience. Therefore, knowledge representations capable of dealing with rules of thumb, fuzzy logic, case-based reasoning, et cetera are relevant (grey area in Figure 4-1).

The following section investigates the suitability of Knowledge-Based Systems for reasoning on respectively demand, supply and transaction related knowledge that is required by the client, and takes into account the characteristics of the input: scarce and ill-defined.

4.2.1 Demand Related Knowledge

This section analyses several types of Knowledge-Based Systems for reasoning on Demand Related Knowledge and discusses their suitability.

Frame-Based Systems

A frame is a logical collection of knowledge related to a single concept. A frame uses slots containing names and values for capturing stereotypical information of the concept. By linking frames to other frames, a network of frames can be created. Frames have procedures that can carry out knowledge intensive tasks, which use and influence the slots. Nowadays, these systems can easily be implemented using any Object Oriented (OO) language. The use of OO programming languages also enables the use of inheritance. Consequently, new frames can be defined as subtypes of other frame classes.

Frame-based systems are used to improve the requirements elicitation process [Scott & Cook, 2003]. It is likely that frame-based systems can also be used in the requirement elicitation process for the Building and Construction industry. For example, a frame-based system can be used in an office-building project to estimate the required size of rooms by using information related to the organisation that the building has to accommodate. Figure 4-2 illustrates the necessary frames for such a system described as a UML class diagram.

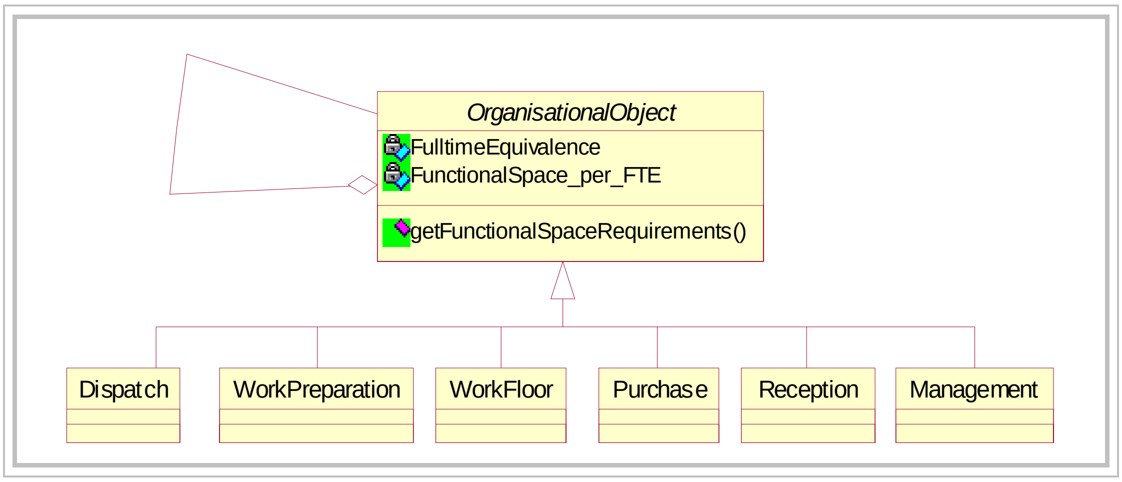


Figure 4-2 UML Class diagram of a Frame-Based System capable of capturing an organization chart for estimating the functional space requirements.

By creating instances of classes, a network of frames can be created capturing the organization chart. Rules of thumb and indices can be used in the procedures to calculate the required space.

Evaluation

Frame-Based Systems are useful for the requirements elicitation process and are useful in the inception phase. In order to use these systems for requirements elicitation process, the requirements have to be pre-defined in the system. This may be a large task but the incremental extension possibilities of these systems support an evolutionary approach.

Rule-Based Systems

A rule-based system is a Knowledge-Based System working with a set of rules, facts, and an interpreter, often called 'inference engine' [Guida, 1994]. The rules define the behaviour of the system, while the inference engine provides the methods to use the rules as well as taking care of the selection and execution of the rules. The knowledge representation is in the form of a 'If-Then' construction. The 'If' part defines a condition and the 'Then' part defines a conclusion. In order to consult the system, users have to provide facts, which will be stored in the memory of the system. The inference engine will search for applicable rules. When the system finds a match, the rule will be executed[1] which in turn could result in new facts. Therefore, the execution of a rule can cause a chain-effect.

A useful aspect of rule-based systems is the ability of these systems to generate more information. Consequently, these new facts enable the system to span different levels of detail. Figure 4-3 illustrates the generation of more information for hospital requirements. When the number of people that the hospital has to provide for, is known, the rule-based system is able to

[1] In rule-based reasoning terminology, this is referred to as 'fired'.

estimate the number of hospital beds. Based on the number of beds, the system can estimate the size of the hospital.

Logic	Example rule
A ➔ B	Hospital for x people ➔ x/1000 beds
B ➔ C	Hospital for x beds ➔ 80 * x m^2 area
C ➔ D	Area x m^2 ➔ x/15 amount of operation rooms

Figure 4-3 Rule-based systems can use their inference engine to generate more information, which in return can trigger new rules.

Another interesting feature of rule-based systems is that the user is allowed to randomly input facts. When such a system has a large amount of facts, a lot of flexibility can be offered to users regarding their (initial) input. For example to get the numbers of operation rooms using the rules in Figure 4-3, the user can choose to input 1) the number of people for which the hospital has to provide for or 2) the number of beds.

Evaluation

The inference possibilities of Rule-Based Systems enable to reason on limited information, which may vary heavily. Therefore, these systems can be very useful for inferring on the information that the client is able to provide during the inception phase.

Fuzzy Logic Systems

Fuzzy logic is a multi-valued logic that allows intermediate values to be defined between conventional evaluations like ‘yes/no’, ‘true/false’, 'black/white', etc. Fuzzy logic systems address the imprecision of the input and output variables by defining fuzzy numbers and fuzzy sets that can be expressed in linguistic variables (e.g. 'small', 'medium' and 'large'). For example, Figure 4-4 illustrates the 'fuzzification' of the notion low, medium, and high.

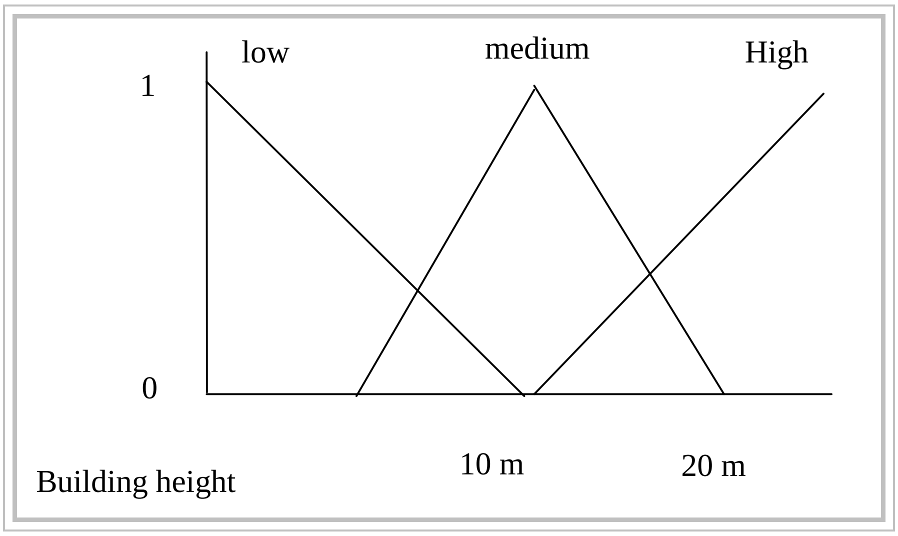


Figure 4-4 Fuzzy reasoning notions.

In this example, the value '12' is according to Figure 4-4 part of the 'medium' set but also part of the 'high' set. Nevertheless, this value belongs 'more' to the 'medium' set than to the 'high' set. Therefore, an index is used to defining the likelihood of the value '12' as a member of a certain set. Using these fuzzy sets, knowledge using crisp values can be consulted but also fuzzy knowledge using fuzzy terms can be used.

Fuzzy reasoning can be closely related to rule-based systems and can have similar constructs. Fuzzy rules can be a 'If-Then' construct that has the general form of "If A Then B" where A and B are (collections of) propositions containing fuzzy variables like 'rather warm'.

Evaluation

During the inception phase, the client is not always able to explicitly define the requirements in fixed and measurable terms but rather in ill-defined values that can appropriately be dealt with by fuzzy logic systems.

Case-Based Reasoning

Case-Based Reasoning (CBR) is a problem-solving paradigm organized in the form of stored cases representing previously solved problems. CBR is based on a simple premise: similar problems have similar solutions. New problems can be solved by finding similar cases and by adapting these cases to match the problem. CBR systems store characteristic features of cases without exact knowledge about the interrelation between these features.

Evaluation

The BC Industry has produced many buildings and a lot of information that can be used as cases in a CBR system. During the inception phase, the client has a limited set of parameters to work with. A building CBR system can provide a list of detailed requirements based on this set of parameters by using information from stored cases.

Similarly, it can be observed that CBR systems are also applicable and appropriate for reasoning on both supply related knowledge (e.g. design alternatives) and transaction related knowledge (e.g. contracts).

4.2.2 Supply Related Knowledge

This section analyses several types of Knowledge-Based Systems for reasoning on Supply Related Knowledge and discusses their suitability.

Genetic Algorithm

Genetic Algorithm (GA) is a directed random search technique. When the solution space increases non-polynomial with the number of parameters, an exhaustive search could be unacceptable. GA randomly provides a set of solutions of a problem that are called chromosomes. These chromosomes contain genes that represent the values for the parameters. By using a fitness function, the found solutions can be measured. By selecting the chromosomes that have the highest fitness, the bad chromosomes will be discarded (survival of the fittest). New solutions and crossed-over chromosomes will be re-evaluated using the fitness function several times until a satisfactory solution has been found. In situations where the solution area is very large, GA can find a solution or several solutions within a reasonable or controllable span of time.

Evaluation

GA has been used to generate alternative space layouts. Figure 4-5 shows the functional requirements for a space layout. When the spatial design for these requirements is articulated mathematically, then genetic algorithms can be used to find solutions.

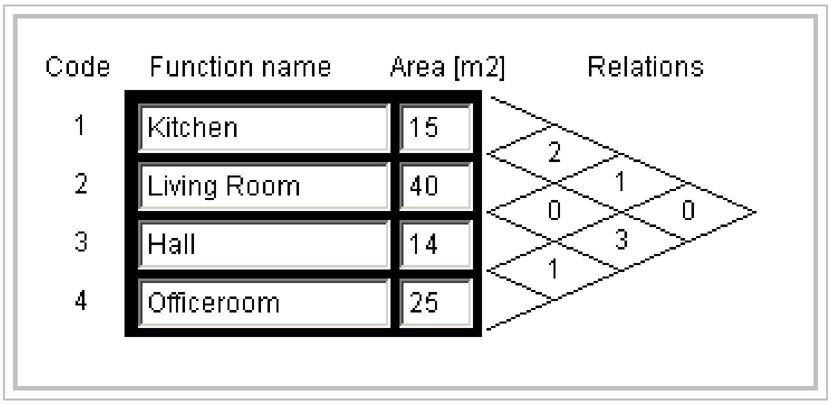


Figure 4-5 Functional requirements for a space layout.

Constraint-Based Systems

A constraint describes a relationship that should be maintained. For example the equation 'A + B < C' holds a relationship between three variables. A Constraint-Based System (CBS) is capable of solving or maintaining a set of constraints including soft constraints. The latter are associated with preference or importance levels and may be left unresolved. To solve a set of constraints different approaches are available including genetic algorithms.

Evaluation

A CBS can support the client with exploring design solutions by giving him the opportunity of changing the solution while maintaining constraints. The CBR ensures that the client is not able to make illegal changes. In addition, constraints can be used to maintain the relationship between requirements and design solutions. While the user attempts to change requirements or design solutions, the CBR warns when a constraint is broken or alternatively automatically corrects the problem. For example, CBS can support the development of space layouts where such constraints have to be rigorously observed [Damski & Gero, 1997]. Figure 4-6 shows a screenshot of a system where layouts may be changed. Similarly, a CBS can cope with conflicting constraints. In this case, the system may not be able to resolve the conflict, but can identify and report the conflict.

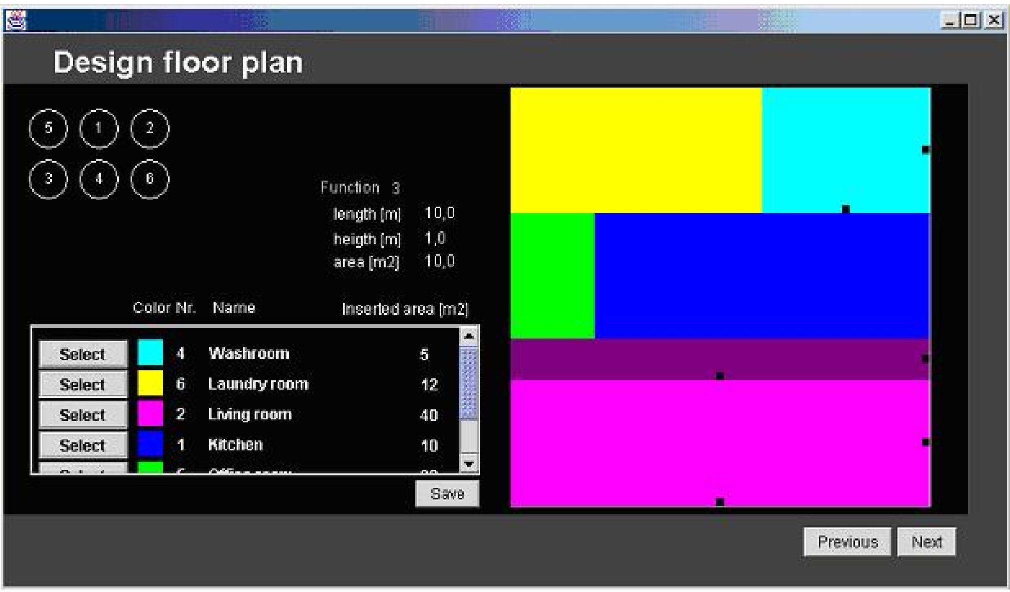


Figure 4-6 A screenshot of a system where constraints maintain spatial relationships.

4.2.3 Transaction Related Knowledge

This section analyses several types of Knowledge-Based Systems for reasoning on Transaction Related Knowledge and discusses their suitability.

Artificial Neural Networks

Neural networks learn to solve problems by example. Neural networks are made up of nodes, which are interconnected. Each node has a numeric weight and each node has a set of input and output that are linked to other nodes. Several layers of nodes are available such as an input layer, output layer and hidden layers. The input layer contains the nodes, which will be activated by input parameters. The nodes in the input layer will determine their activation level using the input parameters and their numeric weight. The activation level determines what message is sent to other nodes. Consequently, the input parameters will change via a network of nodes, into the output parameters. Figure 4-7 illustrates the working of an artificial neural network where the input data x1, x2, x3 will activate a network of nodes which will collectively compute the output y1 and y2.

To determine the numeric weight of nodes, training data is necessary. Using an example set of data, the nodes can correct their computation skills. Offering them more data enables them to 'learn' from this data and consequently adjust their activity. Therefore, when more data is available, the behaviour of the nodes can become more refined and consequently more accurate.

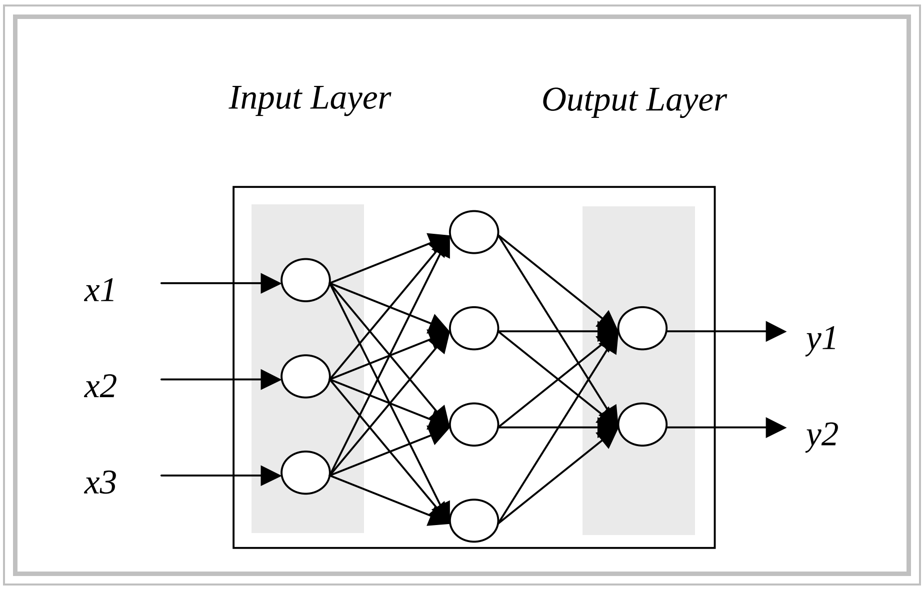


Figure 4-7 The working of a neural network.

Evaluation

Artificial neural networks can for example be used for price estimations by training the system using data on designs and prices.

Evaluation of Rule-Based Systems for Transaction related knowledge

Rule-Based Systems described earlier, can also be applied to transaction related knowledge. For example, rule-based systems can be used to convert one set of facts into another. Declaring explicitly the relation between the elements of both sets will enable rules to convert one set to another (mapping). Figure 4-8 illustrates how rule-based systems can map one format to another by using a set of facts that instruct the system on how to map individual elements. The inference engine will use the facts to translate inserted elements to elements of the new format. As many different experts use different software applications, mapping of information in order to use their own applications is necessary. Rule-based systems may automate certain mappings.

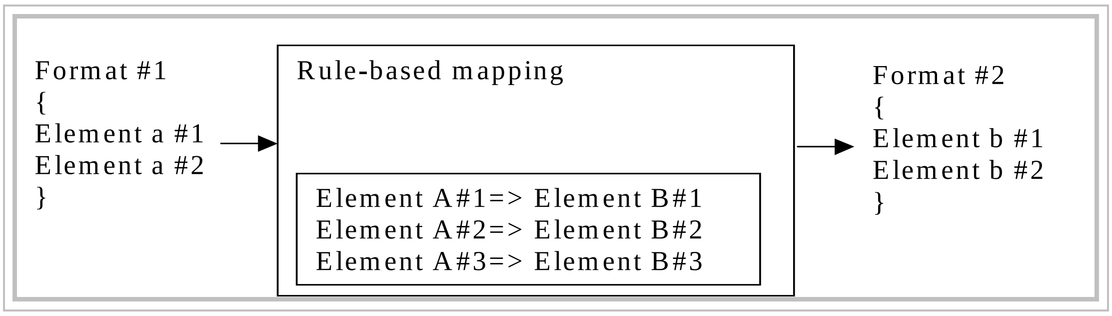


Figure 4-8 A rule-based system for mapping files from one format to another.

4.2.4 Conclusion

This section presents the analysis of a small number of different Knowledge-Based Systems. Already the limited survey shows that Knowledge-Based Systems may well provide ICT-enabled access to portions of the BoCK that are currently out of reach. The following main observations can be made:

- Different types of Knowledge-Based Systems are necessary for capturing and reasoning on different portions of the BoCK.
- The open structure of most Knowledge-Based Systems enables the user to extend the system in a simple and incremental fashion.
- The reasoning capability of these systems is able to deal with ill-defined information and can generate more information.

4.3 STATE-OF-THE-ART INTEROPERABILITY

The Body of Construction Knowledge is vast and fragmented. Therefore, it is crucial to provide interoperability between different Knowledge-Based Systems. This section analyses the state-of-the art interoperability. Technology providing for a low level interoperability such as DWG and IGES are in the scope of this section.

4.3.1 Industry Foundation Classes

The Industry Foundation Classes (IFC) is an initiative of the International Alliance of Interoperability (IAI). The IFC standardises the exchange of information about construction projects using ISO10303 Standard Exchange Protocols (STEP). The ultimate goal of the IFC is to enable interoperability between different applications by exchanging standardised project information. Therefore, the ambition of the IFC is to standardise the exchange of product data throughout the project lifecycle for all disciplines. At present, the inception phase receives less attention.
To cope with the complexity of standardising the exchange of product information, the IFC uses different layers of information models. These different information models relate to each other forming one integrated model. Several CAD vendors have implemented only a small and specific part of the IFC, related to shape and simple building objects like beams, walls and roofs. To date, however, there has been almost no attempt to implement other parts of the IFC [Froese, 2003].

Evaluation

In general, the uptake of the IFC has been marginal and particularly for the inception phase hardly any applications are available.

4.3.2 LexiCon

STABU has produced the LexiCon, a formalised taxonomy for the BC industry providing definitions of concepts to support the creation of a common computer interpretable language. In a single specialisation hierarchy, the BC industry objects are conceptualised and discriminated by their properties. These objects can be product related but also related to spaces and processes [Woestenenk, 2000].

Evaluation

The LexiCon provides a user-friendly interface for conceptualising BC objects. The current content of the Lexicon is detailed and explicit and therefore is not suitable for the inception phase.

4.3.3 Problem Solving Methods and Ontologies

According to Motta [2001] a new approach that explicitly distinguishes a universe of discourse and a Problem Solving Method (PSM), is required to improve the scalability and to support the development of Knowledge-Based Systems. Re-usable universes of discourse, often called ontologies, serve as sources of information for a PSM. Figure 4-9 illustrates this new layout of Knowledge-Based Systems.

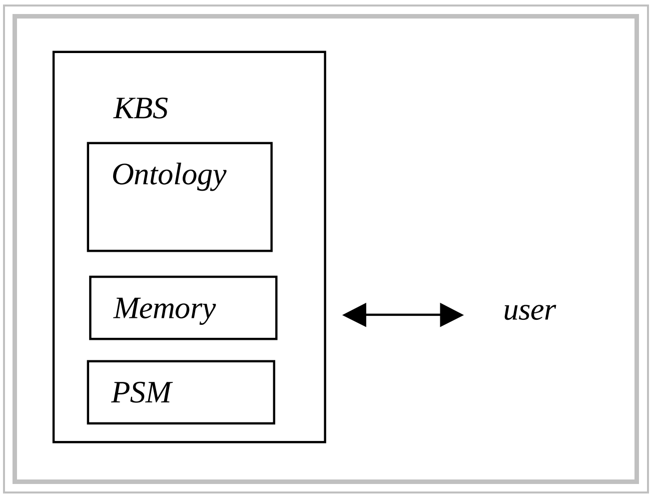


Figure 4-9 The layout of Knowledge-Based Systems using an ontology and a problem solving method.

Evaluation

The re-use of ontologies and different problem solving methods, supports the creation of Knowledge-Based Systems. Furthermore, different PSM's can be used on the same ontology increasing the inference capabilities of Knowledge-Based Systems.

4.3.4 Knowledge Interchange Format

The Knowledge Interchange Format (KIF) is a language for the interchange of knowledge amongst (disparate) programs (created by different programmers, at different times and in different languages). The language uses declarative semantics and it is possible to understand the meaning of expressions in the language without appealing to an interpreter for manipulating those expressions. Typically, when a computer system reads a knowledge base in KIF, it converts the data into its own internal format upon which the computation is done. When the computer system needs to communicate with other systems, it can map its internal data structures into KIF. In addition, several related formats are available such as RuleML for communicating rules using XML-technology or FIPA-ACL, etc.

Evaluation

Formalizing knowledge in the BC industry using KIF (or similar formats) may ensure that the knowledge can easily be transferred to other (upcoming) systems.

4.3.5 Software Agent Systems

Software agents mimic the behaviour of human agents. Key aspects of agents are their autonomy, their ability to perceive, to reason and to act in their surrounding environments, as well as to cooperate with other agents. Many other features can be ascribed to agents like mobility and learning capabilities.

As agents can perform knowledge intensive tasks, they are able to use Knowledge-Based Systems for reasoning purposes [Weiss, 2000]. Multi agent systems where agents can have different goals and different Knowledge-Based Systems are able to resolve conflict situations using negotiations without agents having to share their knowledge. To support the negotiations between agents a standardised language has been formalised called the Knowledge Query and Manipulation Language (KQML). This language enables the communication between non-homogeneous agents [Weiss, 2000]. Another important aspect of software agents is their ability to react proactively. By using their sensors, they perceive their world and determine when to act. There are many definitions of agents and agent properties. Therefore, different independent implementations and approaches are available.

Evaluation

The proactive and autonomous properties of an agent can provide the client with support similar to human experts. A multi-agent system is interesting for reasoning on different parts of the Body of Construction Knowledge. In this regard, each agent can provide access to a specific part of the BoCK. Furthermore, agents can rely on their negotiation tactics to resolve conflicts without having to share their knowledge. Therefore, the Knowledge-Based Systems that an agent can use do not necessarily have to be compliant with other Knowledge-Based Systems operated by other agents.

Furthermore, agents are able to operate Knowledge-Based Systems and adapt the outcome to either the requirements of the client or the phase of the lifecycle.

4.3.6 Semantic Web

The Semantic Web uses web services, which are software applications that can be discovered, described, and accessed through the Internet [Daconta et al, 2003]. As web services are software applications, they can be used for knowledge intensive tasks. A technology such as Universal Description, Discovery and Integration (UDDI) allows other applications to find (discover) the web services. The Web Service Definition Language (WSDL) describes how to communicate with web services and the Simple Object Access Protocol (SOAP) enables the actual communication with web services. In general, these technologies allow the registration, discovery, and use of web services.

The Semantic Web is based on the idea that machines can have a certain understanding of what is communicated by using ontologies. One widely cited definition of ontology is 'A specification of a conceptualisation' [Gruber, 1993]. In this regard, Web services are able to understand each other by using compliant ontologies. They can also perform knowledge intensive tasks, which can be weaved together and hence increase the intelligence of the Web.

To create the Semantic Web several layers of technology have been proposed and are currently under development (Figure 4-10).

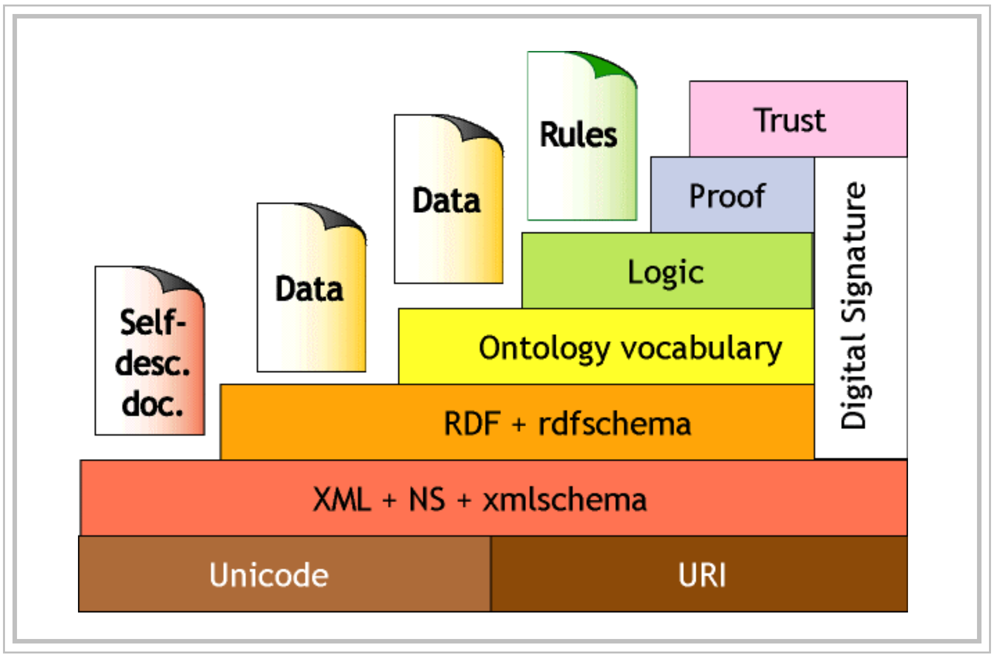


Figure 4-10 The layered approach for the Semantic Web [Berners-Lee et al, 2001].

RDF (Resource Description Framework) and OWL (Web Ontology Language) provide standardised languages for conceptualising semantics and, consequently, support the creation of ontologies. These languages enable the creation and re-use of ontologies in a networked environment that can be interrelated (Figure 4-11). The interrelations and extendibility of these ontologies support an evolutionary approach. Knowledge intensive applications can be interrelated in order to create larger and more sophisticated applications. Currently the Semantic Web does not address standardisation of concepts.

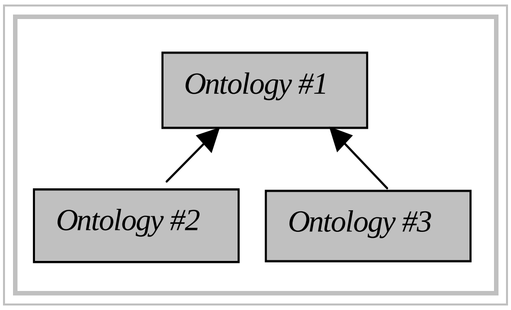


Figure 4-11 A network of ontologies.

Evaluation

Web services are useful for offering services of suppliers through the Internet. Semantic Web services in combination with interrelated ontologies offer possibilities for a distributed approach for reasoning on the BoCK. The infrastructure supports the creation of different ontologies that may suite the BC industry because the Industry consists of many relatively independent and fragmented disciplines, each with their own vocabulary. This creation of domain specific ontologies provides for a platform for interoperability and knowledge sharing.

4.3.7 Conclusions

Semantic interoperability between Knowledge-Based Systems in the BC industry is hardly available. New approaches are available using ontologies that support the development of Knowledge-Based Systems. The Semantic Web adopts this approach and offers standardised languages for creating these ontologies. Semantic Web offers the ability to interrelate distributed ontologies resulting in a network of ontologies. This network supports the distribution of cooperating web services, which can be weaved into a more sophisticated system. Related technology for finding and using these web services on the Internet support this distributed approach for knowledge intensive computing. In addition, agent technology supports the distribution of knowledge intensive computing and offers amongst others proactive and autonomous properties. It seems that the Semantic Web is able to bundle many technologies into a network of knowledge intensive computing.

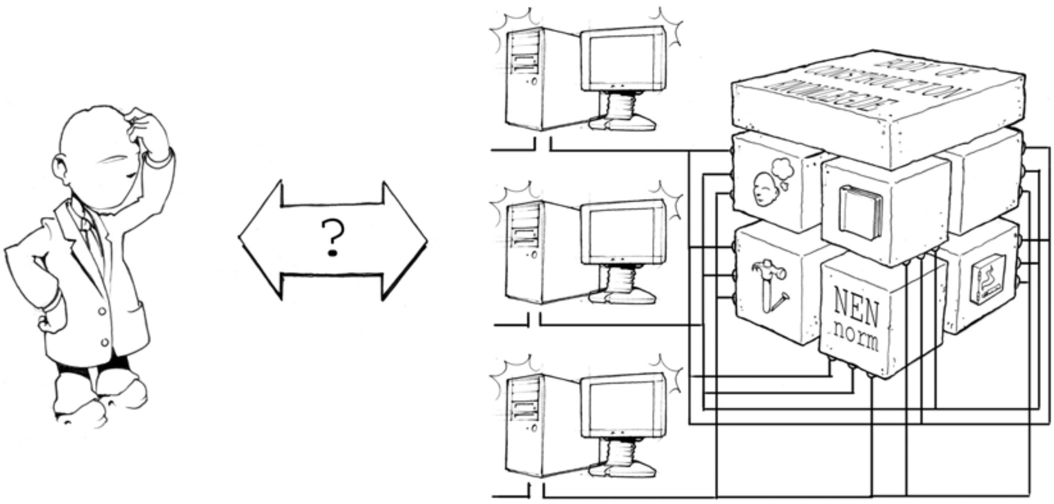
4.4 CONCLUSIONS

Knowledge-Based Systems as part of Knowledge Technology, offer new possibilities for reasoning on the BoCK such as reasoning on more parts of the BoCK than the existing applications. Many different types of Knowledge-Based Systems are available each with their own advantages. In addition, Knowledge-Based Systems are capable of reasoning and dealing with scarce and ill-defined information. Technologies such as Semantic Web, agents and Web Services support a distributed approach for interoperability between Knowledge-Based Systems. However, a large number of (overlapping) approaches and implementations are available with each their own advantages and disadvantages.

5

Detailing the Research Question

This chapter elaborates on the initial research question resulting in several detailed research questions.



Detailing the initial research question: how can ICT provide the client during the inception phase with access to the Body of Construction Knowledge.

5.1 INTRODUCTION

This chapter elaborates the initial research question based on the analyses presented in the previous chapters. This leads to the emergence of five detailed research questions requiring further research.

5.2 CONCLUSIONS OF THE PREVIOUS CHAPTERS

Chapter One gives a first insight into the role of the client in the BC industry and concludes that the role of the client is rather weak because he lacks sufficient control over the building process.
In addition, Chapter One discusses the potentials of ICT for the BC industry. It states that ICT is able to increase the performance of the BC industry. In this regard, ICT may effectively support the client. Subsequently, the following initial research question is raised:

- Can ICT improve the client's role in a Building and Construction process?

Chapter Two analyses the role of the client in the building process. The client needs to specify a demand during the building process. The demand (such as the requirements or services of experts) is defined as the difference between the current situation and the preferred situation including the transformation process. When the client does not have a clear perception of the current and preferred situations, he runs the risk of unsatisfactory results. The client's perception can change by the availability of more relevant information resulting in adjustments of his demand. During the inception phase, the client does not always have sufficient information except when he can access the Body of Construction Knowledge (BoCK). This research points to the capabilities of ICT for accessing the BoCK without the presence of experts.

Chapter Three analyses ICT-enabled access to the BoCK and concludes that the available applications do not adequately provide the client with access during the inception phase because:

- Only a small and specific part of the BoCK can be accessed using existing applications
- ICT-enabled access to the BoCK is fragmented
- Currently no suitable mechanism is available for dealing with problems related to input information
- Most existing applications are too domain specific and are therefore less useful for the client during the inception phase
- Knowledge is embedded in the software applications and not visible/open for the client or other experts

Chapter Four analyses the state-of-the-art Knowledge Technology for providing the client with access to the BoCK. Many different Knowledge-Based Systems are available which offer new possibilities for reasoning on the BoCK. It seems that these Knowledge-Based Systems are able to reason on more parts of the BoCK than the existing applications. Each type of Knowledge-Based System has its own characteristics and usefulness, therefore different types of Knowledge-Based Systems are necessary for providing the client with access to the BoCK. Furthermore, Knowledge-Based Systems are also able to deal with scarce and ill-defined information, which is characteristic for the inception phase. Technologies such as Semantic Web, agents and Web Services support a distributed approach for interoperability between Knowledge-Based Systems.

5.3 ELABORATION

The client needs access to the BoCK during the inception phase in order to improve his demand, which may be enabled by ICT. In this regard, it is necessary to develop an application that can facilitate the access to the BoCK and support the client with his demand, hence a Demand Support System (DSS). In order to develop such a system, more specific requirements (than providing access to the BoCK) are necessary for such a system. The following research question emerges from this discussion.

- What are the requirements for such a Demand Support System?

Knowledge-Based Systems can provide ICT-enabled access to specific portions of the BoCK that are currently out of reach. Different types of KBS are useful in order to provide the client access to the BoCK. In addition, lot of different technologies and approaches are available for integrating non-homogenous Knowledge-Based Systems. The following research question emerges from this discussion.

- What is a suitable architecture for a Demand Support System?

Most existing applications are domain specific and have to be operated by experts. To give the in-experienced client access to the BoCK, he needs to be able to communicate in his own language. The following research question emerges from this discussion.

- How can the Demand Support System support the in-experienced client?

KT offers possibilities for providing the client with a suitable access to the BoCK. As the BoCK is very large, it is not feasible to build a complete set of new applications for the client. However, a lot of legacy applications are available but they only provide access to the BoCK for experts. The following research question emerges from this discussion.

- How can the Demand Support System use legacy applications in order to support the client?

The Semantic Web stimulates the development of a new generation of knowledge intensive applications. The following research question emerges from this discussion.

- How can the Demand Support System be made that can cope with the upcoming new generation of knowledge intensive computer applications?

5.4 CONCLUSIONS

In Chapter One the initial research question is formulated:

- Can ICT improve the client's role in a Building and Construction process?

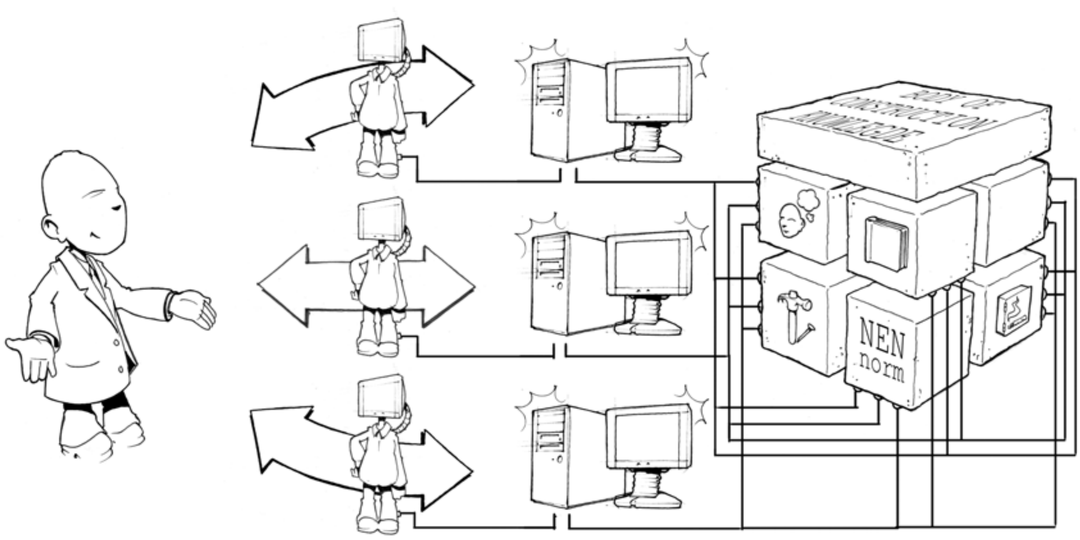
In this chapter the initial research question is elaborated and the following five detailed research questions are emerged:

1. *What are the requirements for a Demand Support System?*
2. *Which conceptual architecture is suitable for a Demand Support System?*
3. *How can a Demand Support System incorporate legacy software?*
4. *How can a Demand Support System support the inexperienced client?*
5. *How can a Demand Support System be made that can cope with the upcoming new generation of knowledge intensive computer applications?*

6

Functional Design of a Demand Support System Using Virtual Experts

This chapter attempts to answers the research questions theoretically.



Virtual Experts give the client access to the Body of Construction Knowledge

6.1 INTRODUCTION

This chapter describes the top-level requirements for a 'Demand Support System' (DSS) and proposes a suitable conceptual architecture for such a system. Furthermore, the chapter answers the research questions formulated in chapter 5. The structure of this chapter closely follows these research questions:

1. What are the requirements for a Demand Support System?
2. Which conceptual architecture is suitable to create such a system?
3. How can a demand support system incorporate legacy software?
4. How can a demand support system support inexperienced users?
5. How can a Demand Support System be made that can cope with the upcoming new generation of knowledge intensive computer applications?

6.2 WHAT ARE THE REQUIREMENTS FOR A DEMAND SUPPORT SYSTEM?

This section presents the requirements for a Demand Support System by first specifying the goal of the system followed by its functional, communication and knowledge support requirements.

6.3 GOAL OF THE DEMAND SUPPORT SYSTEM

The goal of the DSS is to support the client, during the inception phase, with access to the Body of Construction Knowledge. With this access, the client can increase his insight. The increased insight can be used to diminish perception problems and to improve the match between demand and supply. Of course, the client only needs access to a relevant part of the BoCK. To prepare for the real project, the client consults the system with for instance a

'What-If' scenario. The goal of the DSS is not to automate the inception phase but to increase the client's insight.

6.3.1 Functional Requirements for a Demand Support System

To increase the client's insight, he has to be able to interact with the system. The client must be able to communicate with the system in order to:

- Enter the current situation
- Enter the demand, for instance functional and performance requirements
- Enter the his own ideas about possible solutions

To increase the client's insight, the DSS must be able to offer:

- Demand Support, i.e. the development of the requirements or maintaining their consistency
- Supply Support, i.e. offering insight into various feasible concept solutions and their consequences
- Transaction Support, i.e. offering insight into various transaction possibilities such as contract types, tendering processes, etc.

Meeting all the functional requirements for such a system at once is quite ambitious. Therefore the system needs to be able to evolve in time. Figure 6-1 shows the full functional requirements for the Demand Support System using a UML Use Case diagram.

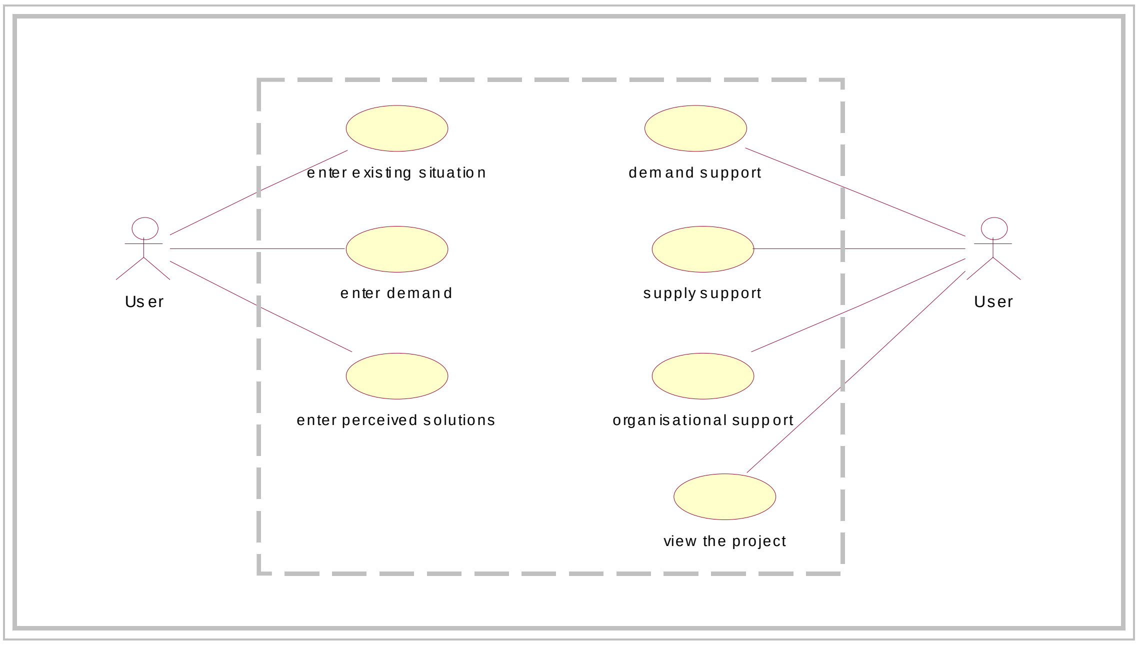


Figure 6-1 A UML Case diagram for the Demand Support System

6.3.2 Communication Requirements

The following communication requirements seem applicable:

- *Capture the Available Project Information Spanning Different Levels of Abstraction*

The system must be able to capture the project specific information. Starting in the inception phase, the client can only provide ill-defined information. The project information will increase in quantity and details during the course of the project. The DSS must support this.

- *Understandable for Inexperienced Clients*

The communication must be understandable for the inexperienced client and therefore in a language that he can understand, like 3D models, virtual reality, hiding technical details, et cetera.

- *Open and Extendable*

The communication with the client must be open and extendable. In this regard the system must allow new user interfaces to be inserted without modifying the system.

6.3.3 Knowledge Support Requirements

The following knowledge support requirements are identified:

- *Automation of (Design) Tasks*

The system must be able to automate (design) tasks for providing the client insight into the consequences of his decisions.

- *Proactive Support*

The DSS must proactively support the client in determining the usefulness of knowledge. For example, the system must warn the user for illegal designs.

- *Multi Disciplinary Support*

The client requires multi disciplinary support coherently.

- *Supporting the Identification of Conflicts*

Conflicts form an important part of the inception phase. Resolving them is necessary for clearing up perception problems and for matching demand and supply. The DSS should provide a suitable basis for identifying these conflicts in order to enable different conflict resolution techniques.

- *Cope with the Fragmented Body of Construction Knowledge*

The DSS must be able to provide access to fragmented, domain specific and overlapping knowledge. Many different Knowledge-Based Systems should be allowed in the DSS.

- *Re-Use of Existing and Future Systems*

The BC industry comprises of different disciplines that independently contribute to the BoCK. The DSS must be able to use existing and future Knowledge-Based Systems that are developed for different purposes.

6.3.4 Conclusions

The requirements for a DSS are similar to those of an expert when supporting the client during the inception phase. The more technical requirements are based on the shortcomings of existing applications.

6.4 WHICH CONCEPTUAL ARCHITECTURE IS SUITABLE TO CREATE A DEMAND SUPPORT SYSTEM?

This section presents an overview of the architecture of the DSS starting with providing ‘the definitions of terms’ followed by the architecture.

6.4.1 Definition of Terms

The following terms are used for describing the architecture of the DSS:

- *Virtual Expert*

A Virtual Expert mimics the behaviour of a human expert. A Virtual Expert proactively supports the client with domain expertise and can autonomously infer from the project model.

- *Ontology*

One widely cited definition of an ontology is “an explicit specification of a conceptualisation of a domain” [Gruber, 1993]. In this thesis, the ontology specifies domain related concepts and their relationships for two purposes. The first purpose is to supply the definitions in order to be able to capture the available project information. The second purpose is to conceptualise a domain of discourse for knowledge models.

- *Project Model*

The Project Model captures the project information from the client as well as the Virtual Experts in a computer interpretable model. This Project Model is by definition compliant with the ontology and contains 'behaviour objects', which enable the Project Model to respond to events.

- *Model Object*

The Model Object is a persistent abstract super type of several objects that form the elements of a Project Model. The Model Object provides containers that can be filled by the semantic definitions specified in the ontology. The Model Objects can be interrelated in order to create a network of objects forming the Project Model.

- *Objects of Interest*

Objects of Interest (OoI) are composed and instantiated Model Objects using the definitions in the ontology.

- *Knowledge Component*

A Knowledge Component is a persistent object containing Meta-Knowledge and Knowledge Models. Virtual Experts use Knowledge Components to infer from the Project Model.

- *Library*

The Library contains a set of Knowledge Components.

- *Meta-Knowledge*

Meta-Knowledge is an explicit specification of conditions for using a Knowledge Model.

- *Knowledge Model*

A Knowledge Model is a computer interpretable model capable of reasoning on a specific knowledge domain.

- *Pattern*

A Pattern is minimal subset of the Project Model, which meets with the conditions defined in the Meta-Knowledge.

- *Behaviour Object*

A Behaviour Object is a persistent object integrated in the Project Model, which determines the behaviour of a predefined part of the Project Model.

6.4.2 Overview of the Conceptual Architecture

The DSS plays the role of several experts, e.g. architects or engineers and consequently must mimic the behaviour of these experts. Therefore, the DSS uses Virtual Experts, which behave autonomously and proactively just like human experts. These Virtual Experts collectively support the client in for example finding an optimal design solution. The conceptual architecture of the DSS contains these Virtual Experts who give the client access to the BoCK. As the BoCK is fragmented, the Virtual Experts offer support in their own knowledge domain without knowing other domains. This approach enables every discipline to create knowledge models independently. To reason on a significant part of the BoCK, several Virtual Experts are necessary, representing various (overlapping) parts of the BoCK. The Internet can provide the necessary infrastructure required for maximizing the scalability of the system. Virtual Experts of different domains (maintained by different organisations) can reside on different locations on the web. To act proactively, Virtual Experts need to have access to the project information. As Virtual Experts can reside on different web locations, the project model can be shared on the Internet. Every Virtual Expert has access to up-to-date project information and is able to react whenever appropriate. Figure 6-2 illustrates the overview of the conceptual architecture of the DSS. In this example, several Virtual Experts residing on the different web spaces access the shared project model and infer from it proactively using their own expertise.

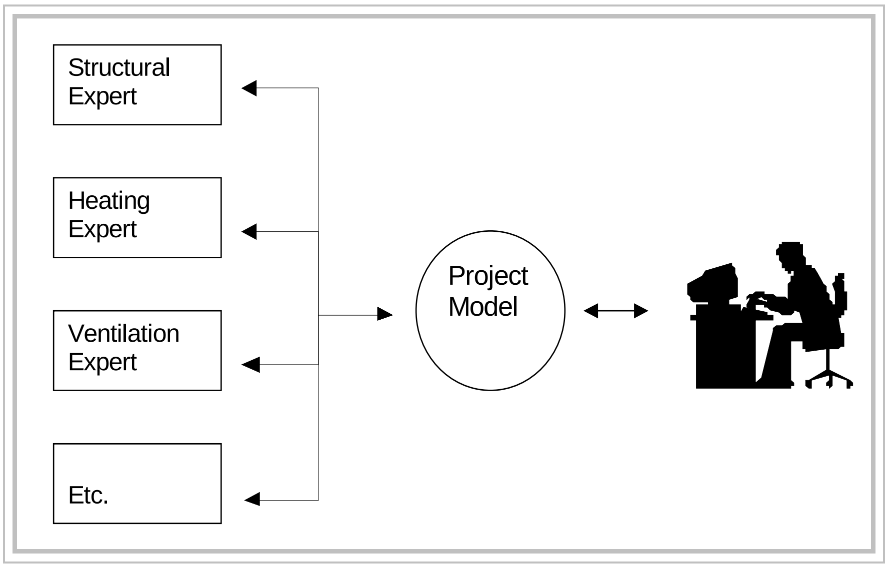


Figure 6-2 Overview of the conceptual architecture of the DSS.

6.4.3 Architecture of the Project Model

To capture the project information, an ontology and a meta-model containing semantic-free Model Objects are used to create the Objects of Interest (OoI). The ontology provides the definitions of the OoI's like the semantics, the properties and the possible relations each OoI may have with other OoI's. The Model Objects provide the necessary containers for these definitions. Figure 6-3 shows the UML Class diagram of the architecture of the Project Model. The abstract Model Object has the subtypes Building Object, Relation Object and Property Object. The Relation Object is able to relate two Model Objects. The Property Object has subtypes like Float, Integer, Enumeration and String to define different types of the properties.

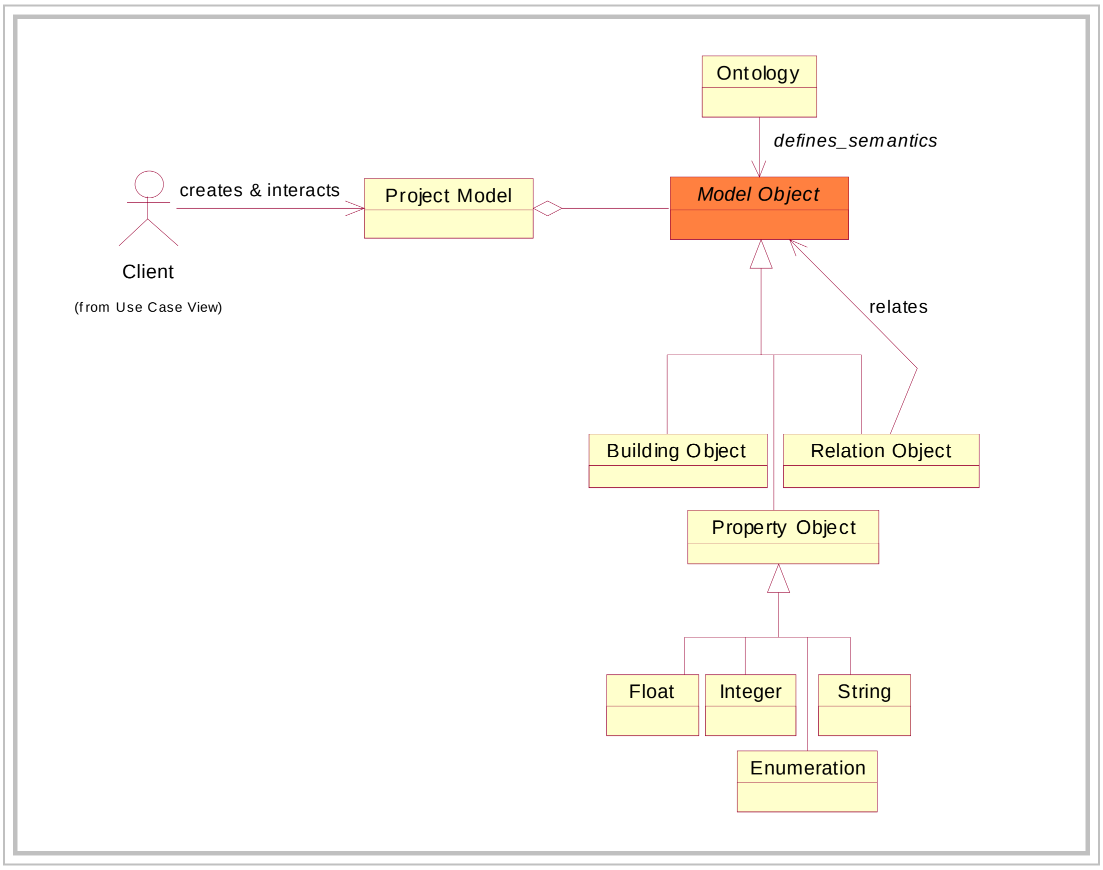


Figure 6-3 The ontologies define semantics and possible interrelations between Model Objects. Project Models consist of Model Objects with subtypes Building Objects, Property Objects and Relation Objects.

The explicit separation between Model Objects and the ontology enables the modification or addition of concepts. These concepts can be directly used in the Project Model without changing the DSS. Furthermore, the ontology is able to conceptualise definitions of OoI's that span different levels of detail. Consequently, project information can be captured on different levels of detail. In addition, the Project Model allows the modification of Model Objects at run-time such as adding or deleting properties. This supports the development of each OoI and consequently enables capturing the natural flow of information. The separation between the Model Object and the ontology also offers flexibility to the user of the DSS. The user is able to determine which OoI to create and can therefore select the appropriate definitions.

6.4.4 Architecture of a Virtual Expert

Virtual Experts can use different types of Knowledge Models. However, these Knowledge Models may use a different ontology than the one used in the Project Model. To make these Knowledge Models compliant with the Project Model, Virtual Experts use Knowledge Components. A Knowledge Component consists of a Knowledge Model interface and Meta-Knowledge, which defines *when* the Knowledge Model is useful. As the Meta-Knowledge is compliant with the ontology used for the Project Model, Virtual Experts are able to infer from the Project Model. Using the Meta-Knowledge, Virtual Experts can search for Patterns in the Project Model. These Patterns contain a minimum subset of the Project Model, which are compliant with the Meta-Knowledge. When a Virtual Expert finds such a Pattern, the Knowledge Model associated with the Meta-Knowledge is by definition relevant to that Pattern. Figure 6-4 shows the UML Class diagram of the architecture of a Virtual Expert, which uses a Library of Knowledge Components.

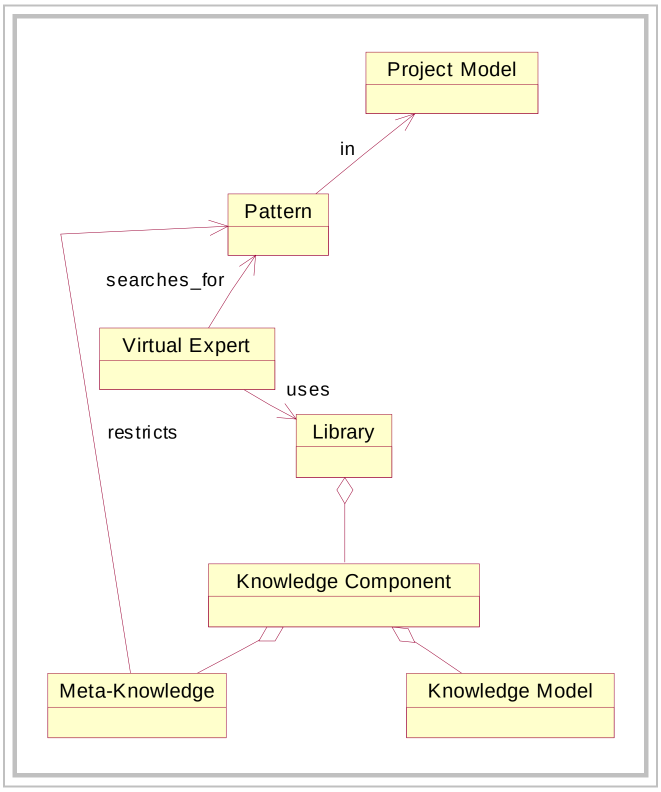


Figure 6-4 The architecture of a Virtual Expert. Virtual Experts use the Library of Knowledge Components to search for Patterns in a Project Model.

Virtual Experts use different knowledge models to infer from the Project Model. The influence of this inference can vary from changing the Project Model topologically like inserting new information, or influencing values of the Project Model. The influence may originate from different Virtual Experts; hence, it is necessary to manage all these influences because they may cause conflicts. Therefore, Behaviour Objects are used to identify which part of the Project Model is under influence of a Knowledge Model. When a Virtual Expert finds a Pattern, it creates a Behaviour Object and connects it to the appropriate part of the Project Model. As the Behaviour Object is part of the Project Model and can have its own properties and relations with other Model Objects, it is therefore a (new) subtype of Model Object. Figure 6-5 shows the UML Class diagram of Virtual Experts, Knowledge Components, Behaviour Objects, et cetera.

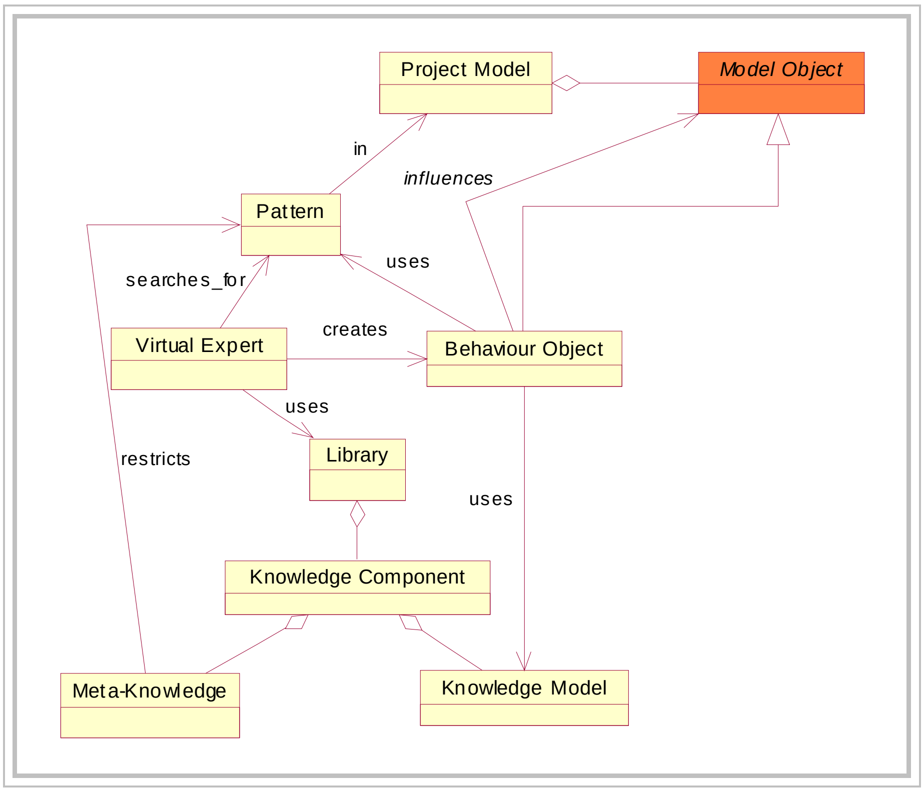


Figure 6-5 Architecture of a Virtual Expert including the Behaviour Object. Virtual Experts create Behaviour Objects to influence Model Objects.

6.4.5 Architecture of Knowledge Components

The Meta-Knowledge of a Knowledge Component refers to a Model Object that can relate to other Model Objects resulting in a network of Model Objects. This network defines when the Knowledge Model can be used. Several types of constraints for the Model Objects can be added to the network such as property value constraints and topological constraints. This network (meta-knowledge) should be connected to the Knowledge Model. Figure 6-6 presents a UML Class diagram of the architecture of a Knowledge Component. The internal working of the Knowledge Model is not relevant for this architecture. Therefore, all knowledge models can be used when they are able to specify their input and output.

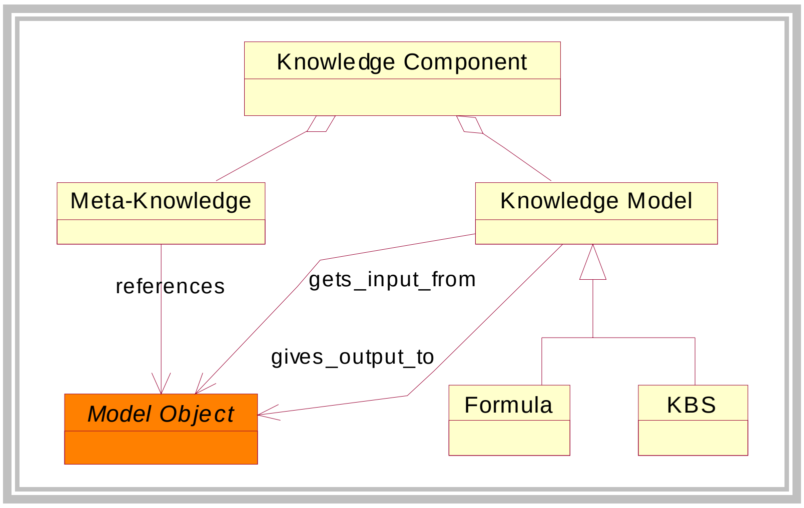


Figure 6-6 Knowledge Component. The Meta-Knowledge comprises of a network of Model Objects, which specify the in- and output of the Knowledge Model.

6.4.6 Architecture of Behaviour Objects

As described in the global architecture, Behaviour Objects define explicitly what the influence of a knowledge model is on the Project Model. When changes occur in the Model Objects to which a Behaviour Object is associated with, the Behaviour Object may react by consulting its knowledge model. Three types of relations are defined to relate the Behaviour Object to other Model Objects. The first one is the '*determines'* relation that specifies the

Model Objects, which are influenced by the knowledge model. The *'invokes'* relation specifies the Model Objects, which are necessary for consulting the knowledge model. Finally, the *'context'* relation is used to validate the use of the knowledge model. Changes in Model Objects (that are related to the Behaviour Object by the *'invokes'* relation), will cause the Behaviour Object to re-consult its knowledge model. Figure 6-7 illustrates an example using UML Object diagram where a Model Object 'building' has decomposition relations with Model Objects 'storey space'. By increasing the property 'height' of the Model Object 'storey space', the Behaviour Object will affect the 'height' of the Model Object 'building'.

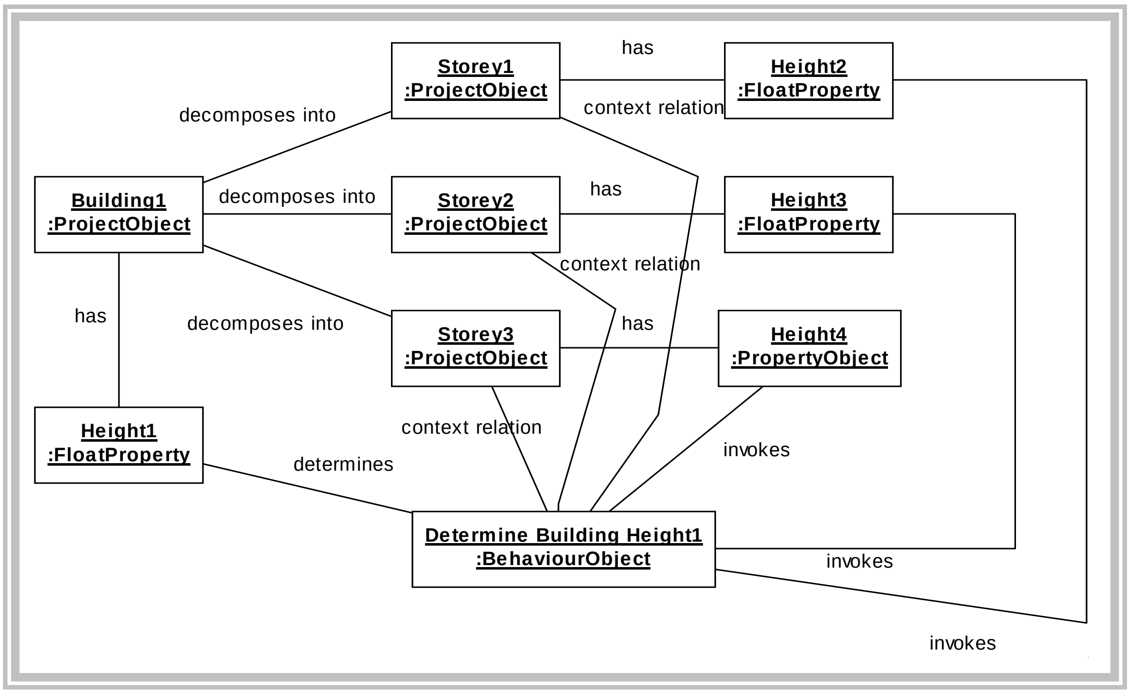


Figure 6-7 Three types of relations connect the Behaviour Object to the Project Model. Whenever the height property of the 'storey space' changes, the Behaviour Object can directly calculate the total height of the 'building'.

Explicitly defining which knowledge model influences what part of the Project Model, enables management of influences such as automatically finding conflicts. When the Project Model contains many Behaviour Objects, Model Objects can be connected to more than one Behaviour Object. For example, a property value can be determined by a Behaviour Object and at the same time it can be connected to another Behaviour Object using the *'invokes'* relation. When the first Behaviour Object is invoked, it may change the property value, invoking another Behaviour Object. Subsequently, it could result in a chain of reactions. Figure 6-8 shows an UML Object diagram of a Project Model with two Behaviour Objects. In this situation, the Behaviour Object may propagate changes in the Project Model.

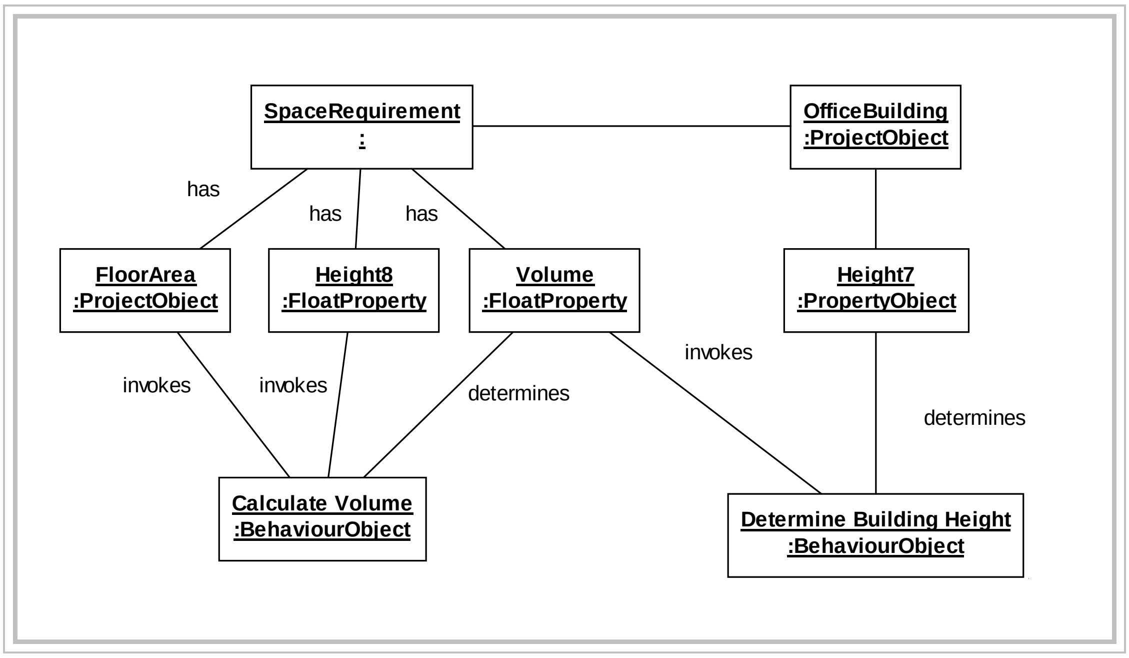


Figure 6-8 UML Object diagram of a Project Model with two Behaviour Objects.

It is required to manage the propagation of changes when either two or more Behaviour Objects influence the same Model Object (Figure 6-9) or during an iterative chain of reactions (*Figure 6-10*).

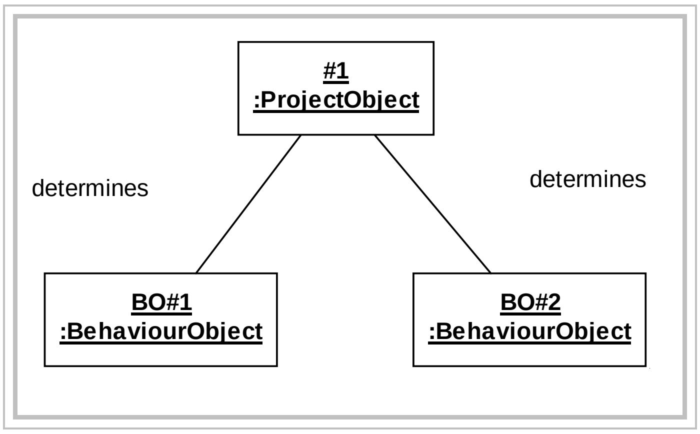


Figure 6-9 UML Object diagram of two Behaviour Objects influencing the same property (a potential conflict situation).

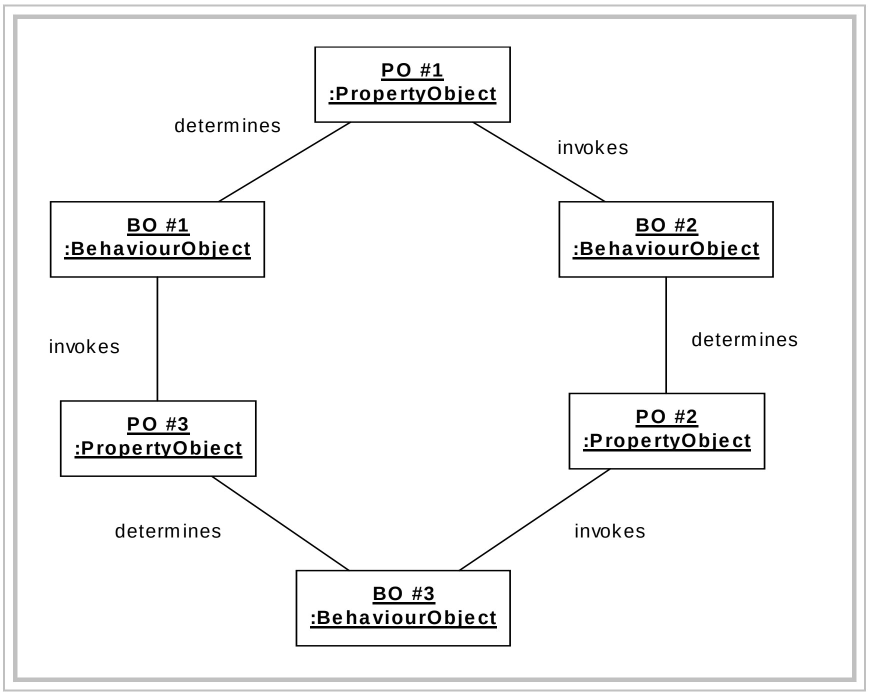


Figure 6-10 UML Object diagram of an iterative chain (a potential conflict situation).

In an iterative situation, the circular chain of reactions can converge to a balanced situation where no conflicts are present. However, non-converging cycles are also possible. Therefore, iterative chains of reactions must be monitored and controlled for convergence.

All these situations require a form of knowledge management. Different approaches are viable such as enabling users to overrule the influence of Behaviour Objects or (semi-) automatic conflict handling using agent negotiation, et cetera. A discussion of these approaches and their suitability is out of scope for this thesis.

6.4.7 Reasoning of a Virtual Expert

A proactive approach requires that Virtual Experts continuously search the Project Model for Patterns. Figure 6-11 illustrates an UML Activity diagram of one of the main activities of such a Virtual Expert.

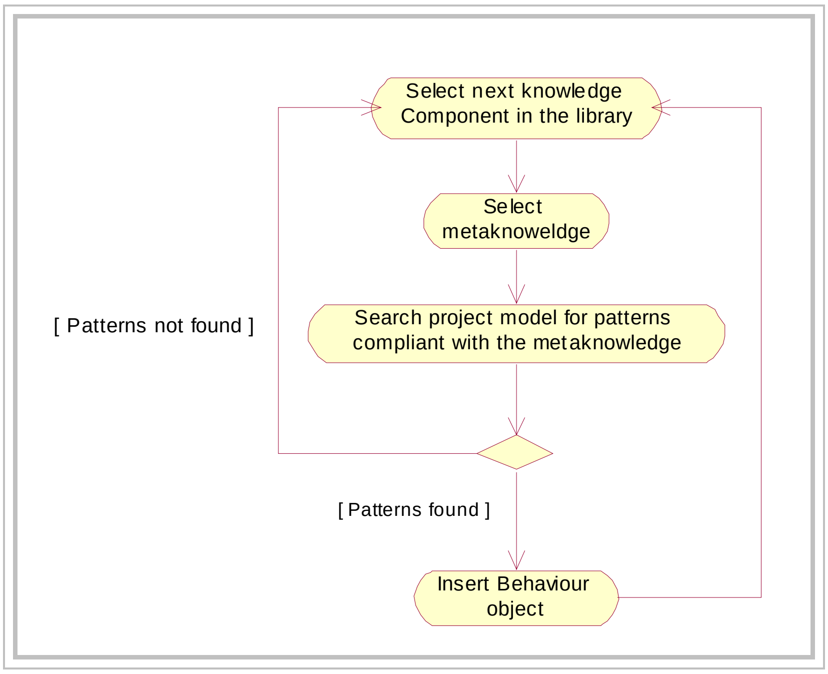


Figure 6-11 UML Activity diagram illustrating the continuous search of a Virtual Expert for Patterns.

Virtual Experts continuously search the Project Model for Patterns. During the search, the Virtual Expert may need to access the ontology in order to check if a subset of the Product Model is compliant with a specified Meta-Knowledge. For instance, an office building defined as subtype of a building is compliant with conditions prescribing a building.

6.4.8 Conceptual Architecture

The previous sections gave an overview of the conceptual architecture and discussed several parts of the architecture in detail. The full conceptual architecture is shown in Figure 6-12.

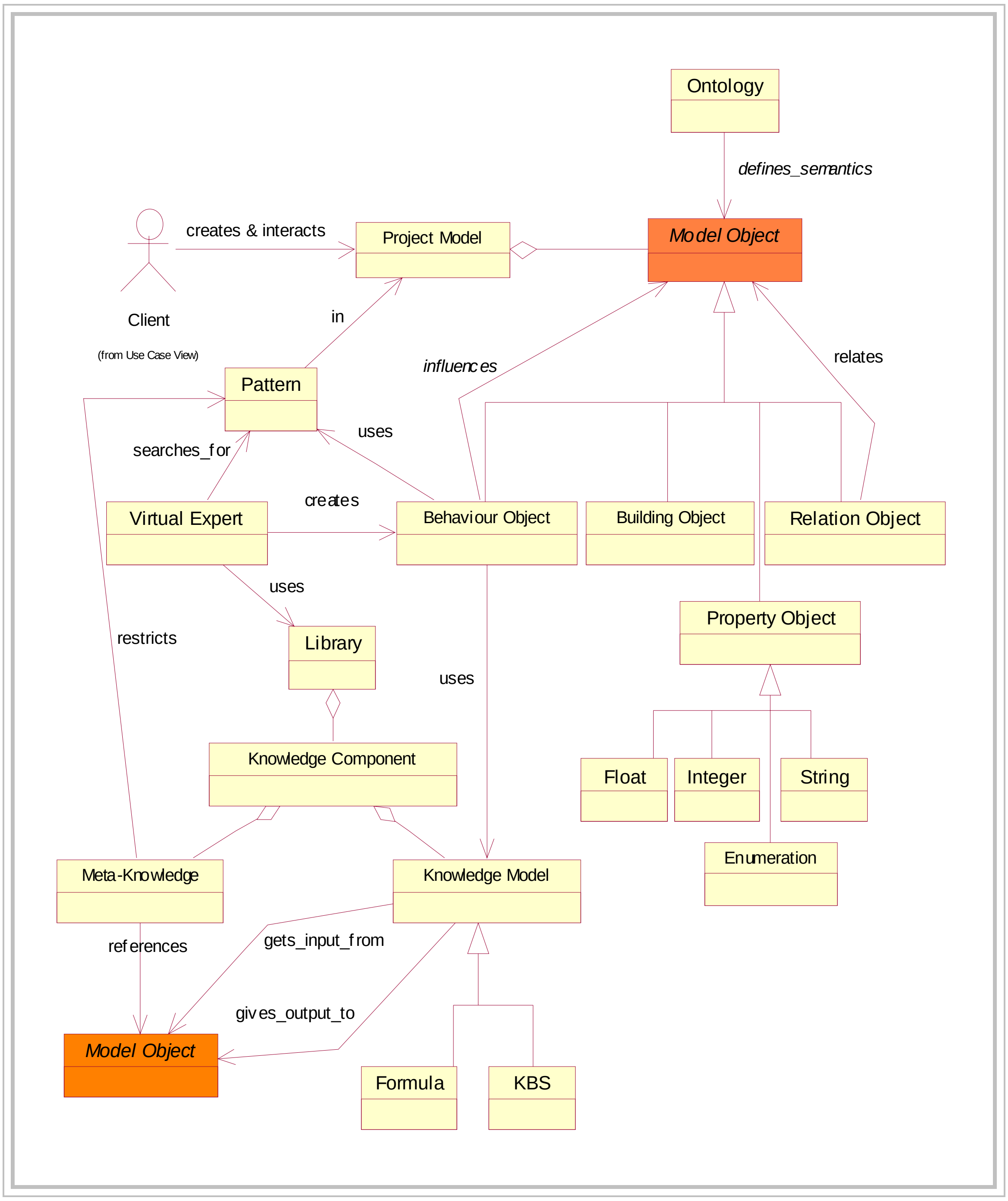


Figure 6-12 UML Case diagram of the proposed conceptual architecture of the Demand Support System.

6.4.9 Conclusions

This section proposes a conceptual architecture for a DSS and elaborates the second research question[2]. The proposed architecture offers support for capturing project information and uses Virtual Experts for reasoning. The latter are able to reason independently and use Knowledge Components for accessing Knowledge-Based Systems. The conceptual architecture of the DSS uses Behaviour Objects in order to identify all influences on the Project Model.

6.5 HOW CAN A DEMAND SUPPORT SYSTEM INCORPORATE LEGACY APPLICATIONS?

A significant amount of software applications are developed for the BC industry. Creating new Knowledge-Based Systems dedicated to the DSS without re-using the 'legacy' applications would be inappropriate. This section discusses how the proposed conceptual architecture is able to handle these legacy applications.

6.5.1 Incorporating Legacy Applications

A Knowledge Component can interface to a legacy application when the input and the output of the latter are defined (Figure 6-13). First, the necessary input has to be supplied, and then the legacy application has to run. The interface can integrate the output of the legacy application with the Project Model.

[2] 'Which conceptual architecture is suitable to create such a system?'

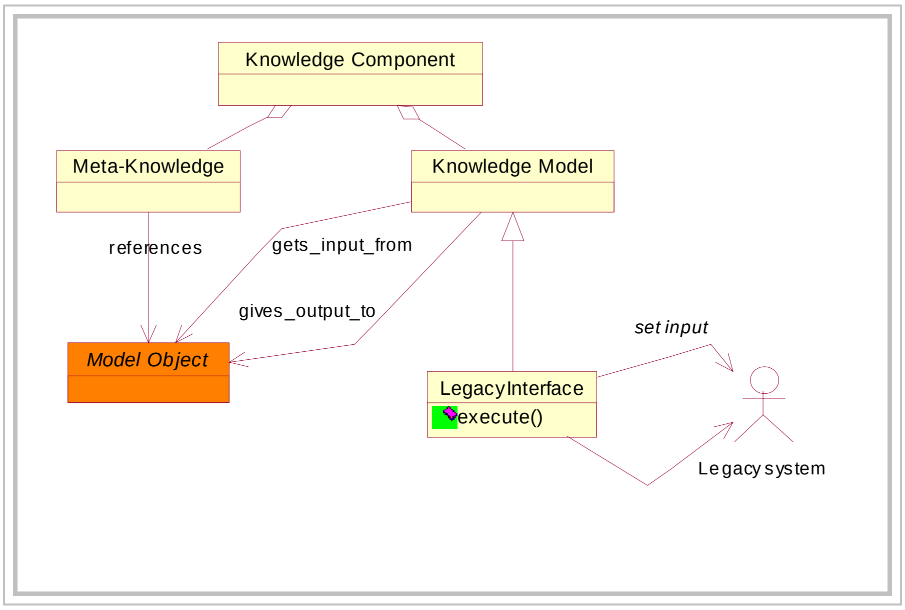


Figure 6-13 UML Class diagram of a Knowledge Component using an interface to a legacy application.

6.5.2 Guided Extension

Construction knowledge (and legacy systems incorporating the knowledge) is available on all the levels of detail of Building-Construction projects, from the building in its environment, to the way buildings are divided in floors, transportation areas, and such, to the way individual spaces can be grouped, shaped and equipped, to the various ways to connect a window in a wall.

One way to solve this problem is to start on a high level of abstraction (no details) and tries to abstract construction knowledge into simple rules of thumb. As knowledge abstraction obviously is very difficult and inaccurate it seems a good idea to look into the reverse possibility, i.e. generate detailed information from less detailed information. This way legacy systems and existing construction knowledge can be readily applied, no further abstraction being necessary. Instead of random detail generation, the idea came up that it often is possible to constrain the solution space during the generation process. In the case of a building complex, it might be possible to use a compact factor that indicates the spatial appearance of the complex, or maybe use some other typology of preferred solutions. In the case of an office building design it is already more or less clear how many square meters would be ideal for each workspace. Applying that type of guidelines makes the generation of details much more focused. As the generation of details can be seen as steps in the Intension-Extension dimension (extensions of meaning captured in the intension), the approach has been termed Guided Extension.

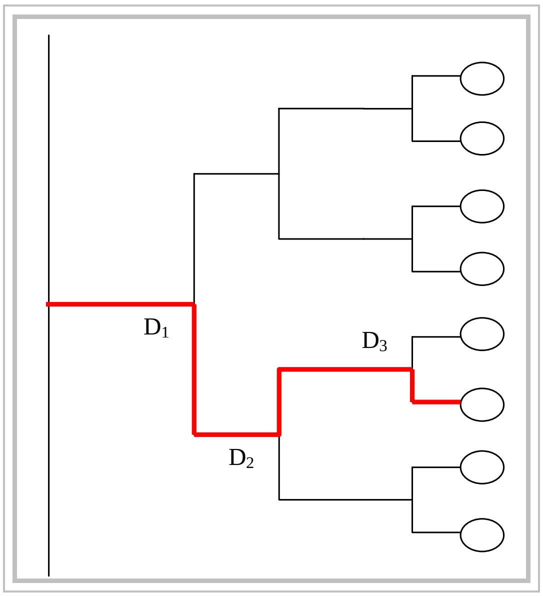


Figure 6-14 Virtual Experts will generate extra information to create a design with enough detail to supply the legacy application by making the decisions D1, D2 and D3.

6.5.3 Conclusions

The conceptual architecture is capable of integrating legacy applications, as part of a Knowledge Components, using a legacy interface. Guided Extension enables the usage of these legacy systems during the inception phase. The intension information is extended by knowledge rules in order to supply the legacy application with the necessary information. During the demand development the same legacy application can be used assuring a high level of consistency compared to using different knowledge models on different levels of detail for similar evaluations.

6.6 HOW CAN A DEMAND SUPPORT SYSTEM SUPPORT INEXPERIENCED USERS?

The (inexperienced) client may have difficulty interpreting technical information depending on amongst others the knowledge position of the client. The architecture must be able to support communication with the inexperienced client. This section discusses how the conceptual architecture of the DSS supports the communication.

6.6.1 Communication with the Client

Virtual Reality and 3D models can facilitate appropriate communication between the DSS and the client regarding the presentation of design solutions.

The Project Model must be able to capture shape related information in order to use VR and 3D models for presenting design solutions. Defining a shape model in the ontology enables capturing the shape related information. A converter using shape related information creates 3D content, which can be visualised (Figure 6- 15). In addition, the converter can use the semantics of the OoI's to convert the related shape models into 3D content.

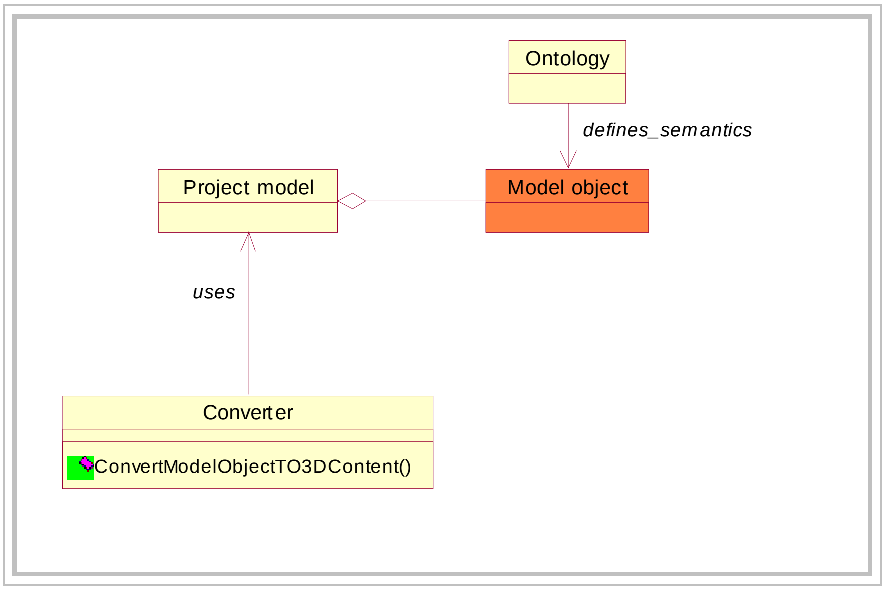


Figure 6- 15 UML Class Diagram for visualising the Project Model.

6.6.2 Interpreting Technical Information

The conceptual architecture enables Virtual Experts to determine how to communicate with the client. This enables Virtual Experts to further interpret and present technical information in an appropriate format, which is understandable for the client. For example, technical information regarding conformance checking can be abstracted and presented to the client in a format such as ‘the building does not comply with the fire safety regulations because it misses several fire exits’. This interpretation of existing technical information can be done using Behaviour Objects supporting different levels of abstraction. Figure 6-16 shows an example where a Behaviour Object interprets technical information related to energy consumption using the floor area to present it in terms of a given scale (efficient or inefficient).

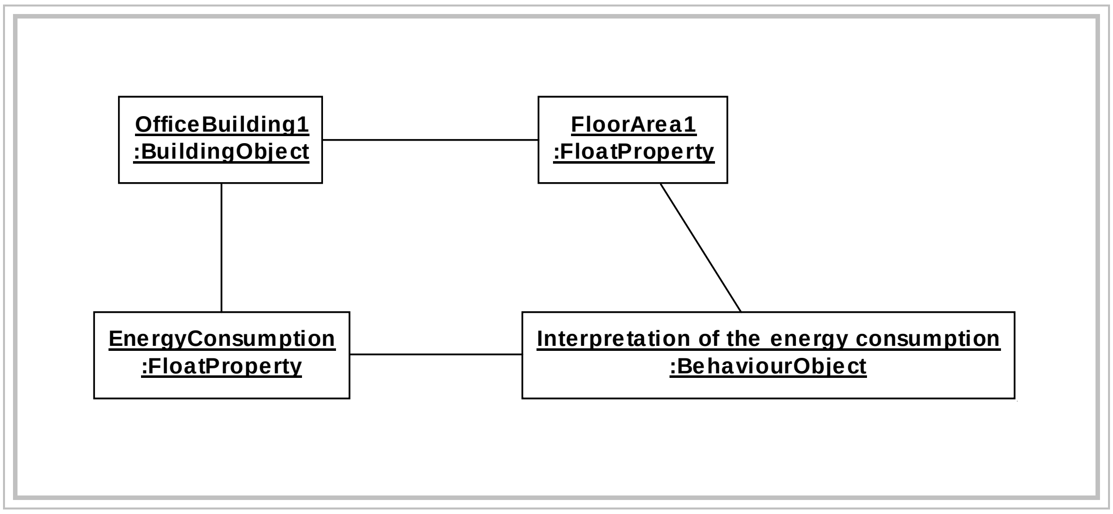


Figure 6-16 UML Object Model of a Behaviour Object interpreting the energy consumption based on the floor area of an office building.

6.6.3 Conclusions

The communication with the client can be supported using Virtual Reality and 3D representations. Virtual Experts can determine how to communicate with the client independently. In addition, the conceptual architecture supports a further presentation of information on different levels of abstraction.

6.7 HOW CAN A DEMAND SUPPORT SYSTEM BE MADE THAT CAN COPE WITH THE UPCOMING NEW GENERATION OF KNOWLEDGE INTENSIVE COMPUTER APPLICATIONS?

This section explains how the DSS can cope with up-coming technologies. One such up-coming technology is the Semantic Web. With the advent of this Semantic Web, a new generation of knowledge intensive computer technology will be supported.

6.7.1 DSS using Semantic Web Technology

Semantic Web technology offers a standardised language for defining and interrelating ontologies. Based on a network of interrelated ontologies, web services can be created that are highly re-usable (see chapter 4 for the analysis of the Semantic Web).

The DSS proposes the creation of Objects of Interest at run-time using an ontology. New ontologies can be used at run-time enabling the creation of new OoI's without changing the program. Therefore, the DSS is able to benefit from the network of ontologies proposed by the Semantic Web. Figure 6-17 illustrates the link between the DSS and a Semantic Web ontology using OWL.

Furthermore, future Semantic Web services can be incorporated in the DSS because of their compliancy with the Semantic Web. Virtual Experts can use these Next Generation web services and become the interface between such a web service and the Project Model by using Knowledge Components.

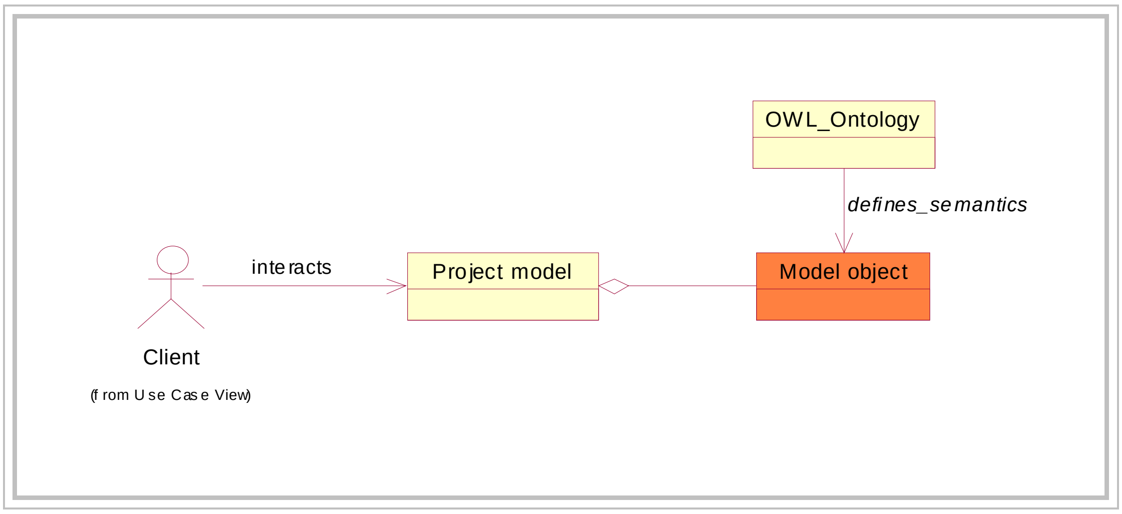


Figure 6-17 UML Class diagram of the DSS using an OWL ontology to create Objects of Interest.

6.7.2 Project-Type-Specific DSS

To determine the parts of the BoCK that the client needs to access, dependents on the type of project, the knowledge position of the client, et cetera. As it is difficult to predict the parts that need to be accessed in advance, the ultimate goal of a DSS is to provide access to the whole BoCK. Evidently, the development of such a system requires a great effort. In order to reduce the amount of knowledge, which potentially can be accessed, different types of DSS can be developed focussing on one type of project (such as office building projects, residential projects) or on one type of client. A project-type-specific DSS can be developed, which can benefit from emerging knowledge intensive applications. In the future, new Demand Support Systems can emerge capable of re-using both the available ontologies and the available Virtual Experts. The evolutionary capability of the DSS points to emergence of a larger and more sophisticated system in the future.

6.7.3 Conclusions

By using Semantic Web technology, the ontology of the DSS can be extended with new ontologies enabling Virtual Experts to reason on new parts of the

BoCK. As the Semantic Web technology supports a distributed infrastructure, it can be used for the evolution of the DSS that in return can support new Virtual Experts. An evolutionary and distributed approach is proposed for the DSS. The evolutionary capability justifies the creation of project-type-specific DSS, which can grow into a more sophisticated system using future web services and future Virtual Experts.

6.8 CONCLUSIONS

This chapter identifies the requirements for a DSS and proposes a conceptual architecture for such a system. In such a system, Virtual Experts proactively use a network of distributed and independent Knowledge-Based Systems in order to infer on the shared Project Model.

Furthermore, the proposed architecture enables the integration of legacy applications and enables the interpretation of technical information at different levels of abstraction for the client.

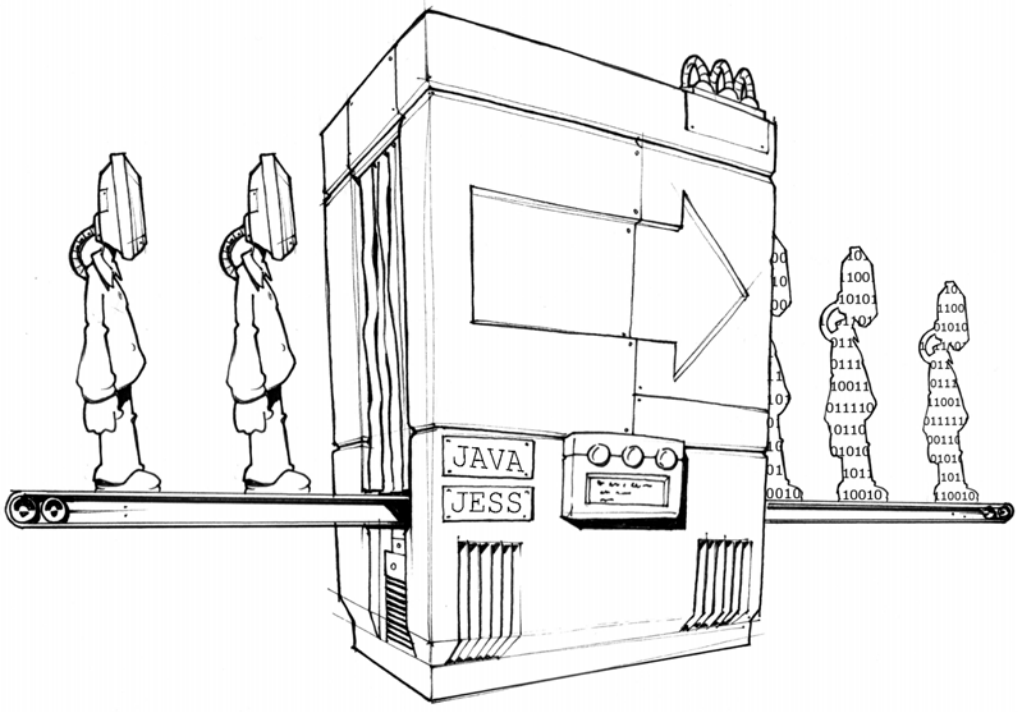
The use of the ontology supports the conceptualisation of Objects of Interests spanning different levels of detail and facilitates flexibility regarding the extension of the DSS.

Compliancy with the Semantic Web technology provides the DSS with capabilities to cope with the upcoming new generation of knowledge intensive applications. The evolutionary approach enables the creation of a project-type-specific DSS that can evolve into a larger and more sophisticated system in the future.

7

Implementation of a Prototype Demand Support System

This chapter discusses the implementation of a prototype Demand Support System based on the conceptual architecture.



Implementation of the Demand Support System including Virtual Experts

7.1 INTRODUCTION

This chapter discusses the implementation of the key components of the prototype DSS. The first section discusses the implementation of the Project Model including the ontology implementation. In addition, this section describes how this Project Model can be shared over the Internet in order to make the Project Model accessible for Virtual Experts. The second section discusses the implementation of a Virtual Expert and explains how a Knowledge Component can be implemented as well as how Patterns can be found in the Project Model. Furthermore, a Knowledge Component editor and the integration of legacy applications are discussed. The last section discusses the implementation of the user-interfaces including the implementation of 2D and 3D presentations of the Project Model.

When implementing a prototype Demand Support System, many choices have to be made such as the type of programming language, design pattern, style of programming, existing components, etc. This chapter opts for a balance between scientific feasibility and the availability of necessary skills. Java is used as the development environment because of its stability and Object Oriented paradigm.

7.2 IMPLEMENTATION OF THE PROJECT MODEL

This section discusses the implementation of the Project Model and the ontology of the DSS, as well as issues related to the sharing of the Project Model over the Internet.

7.2.1 Project Model

The Project Model is implemented using Java Objects with the same name. The Project Model extends from a Java 'Hashtable'[3] enabling it to contain many Model Objects. Both the Model Object and the Project Model implement the Java 'Serializeable' interface, which makes it possible to send the Project Model including the Model Objects over the Internet. Furthermore, every Model Object has a unique ID, which is used as the key for the Hashtable. The unique ID is necessary for the live sharing of the project information over the Internet, which will be discussed in the following section. Figure 7-1 shows the UML class diagram of the implementation of the Project Model.

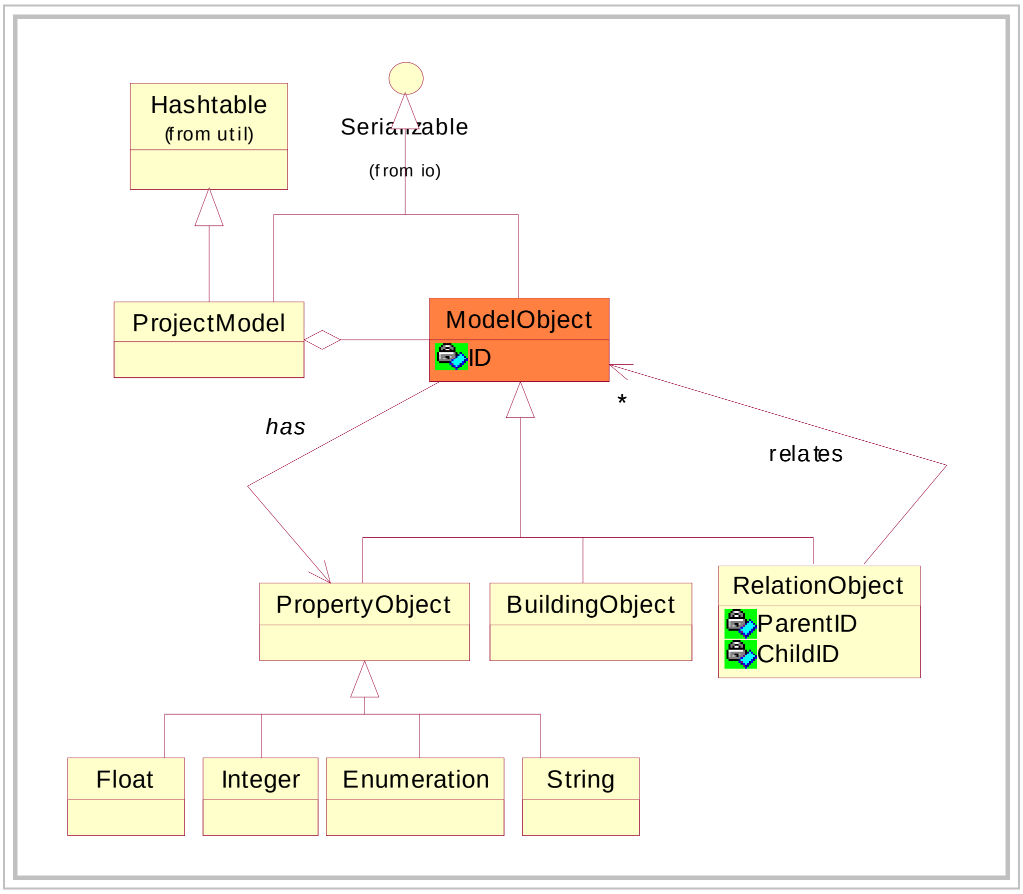


Figure 7-1 A UML Class diagram of the Project Model implemented in Java Objects.

[3] a Hashtable stores the relation between keys and values objects.

7.2.2 Live Sharing of the Project Model

As described in the previous section, Java offers a mechanism for communicating objects over the Internet. Using servlets[4], a server distributing the Project Model containing all the Model Objects can be created. Client-computers can retrieve the Project Model when they connect to this server using a simple URL. With this implementation, sharing the Project Model is possible but hardly suitable for real-time interaction by Virtual Experts. For live sharing of the Project Model, Java offers a very simple-to-use API[5] called Java Shared Data Toolkit [JSDT]. This toolkit enables virtual sharing of objects over the Internet. Serializing the objects in the Project Model enables the communication of every single Model Object over the Internet. The JSDT creates a channel for sending and retrieving modified objects (Figure 7-2).

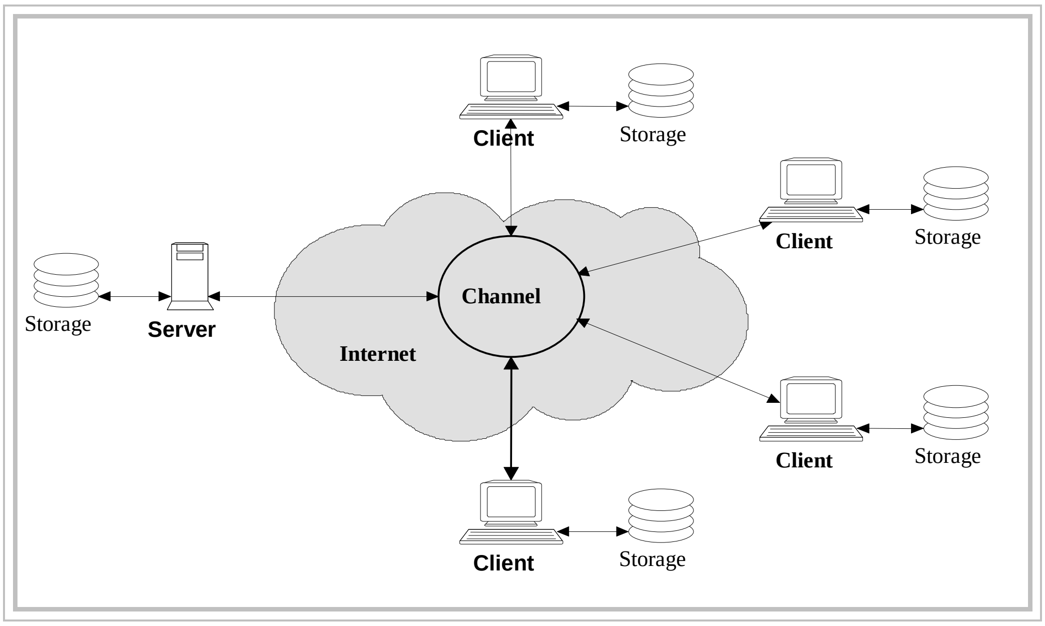


Figure 7-2 Live sharing of the Project Model using the Java Shared Data Toolkit.

One dedicated server provides the continuous availability of the channel. Client-applications can connect to this server in order to retrieve the Project

[4] A server side program for extending the functionality of a web server.

[5] Application programming interface

Model first. Then they connect to the channel to receive up-dates of each Model Object. When the client-computer (e.g. a Virtual Expert) changes a Model Object, this computer will send the updated Model Object over the channel. Every client-computer connected to the channel retrieves this up-dated Model Object. The local copy of the Project Model on each client-computer must be updated with the Model Objects retrieved from the channel. Client-computers use the unique ID's for locating and replacing out-dated Model Objects.

7.2.3 Ontology

A Microsoft (MS) Access database is used to capture the ontology for project-type-specific DSS. A simple scheme is implemented for capturing the definitions of the Objects of Interest. The scheme handles requirement objects separately because these requirement objects are attached to a physical object of interest like building, wall, roof, space, et cetera. Furthermore, these requirement objects are connected to aspects objects, which describe a functional system such as a ventilation system or a heating system.

Two Java classes are implemented to give access to the ontology database. The first Java class, called 'Database', deals with accessing the MS Access database. The second Java class called 'Ontology' implements a high-level functionality on top of the 'Database' class. An editor facilitates the creation of the ontology based on these classes (Figure 7-3).

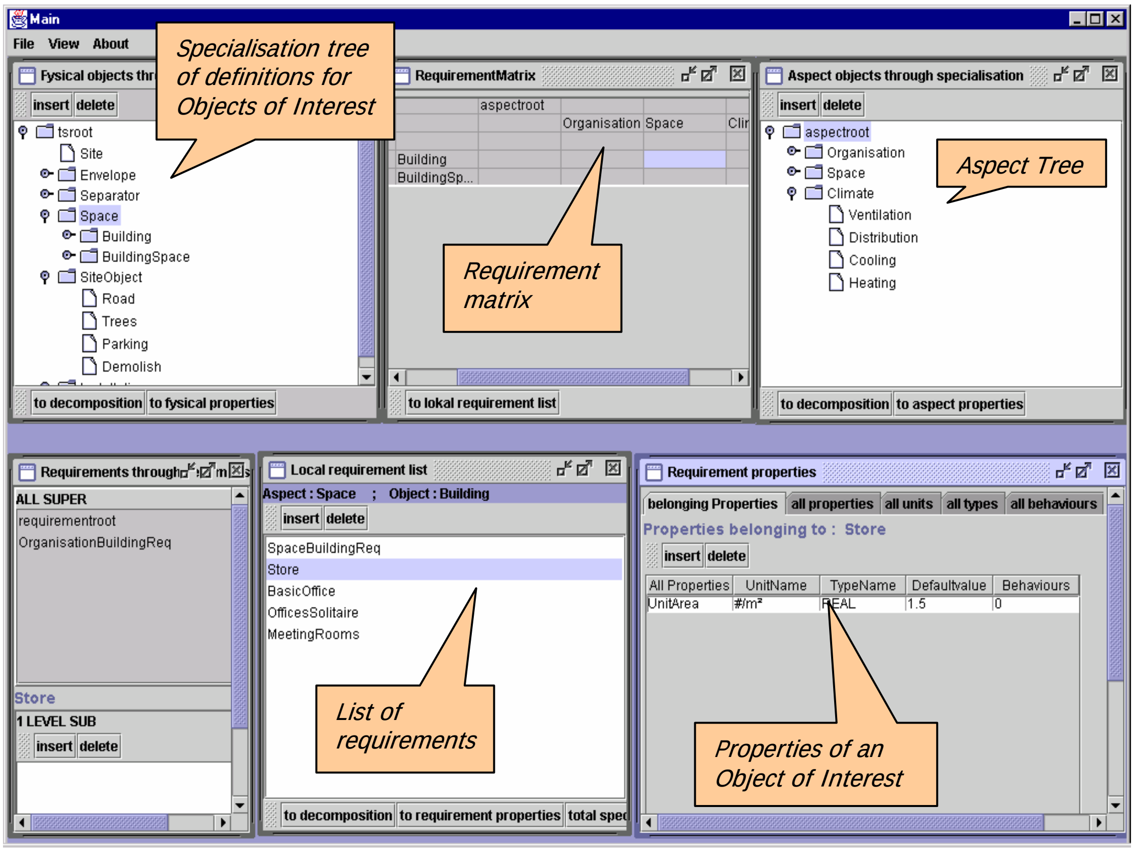


Figure 7-3 A screenshot of the ontology editor.

Another Java Class called the ‘Model Object Factory’ uses the ‘Ontology’ class to create Model Objects according to the specifications provided by the ontology (Figure 7-4).

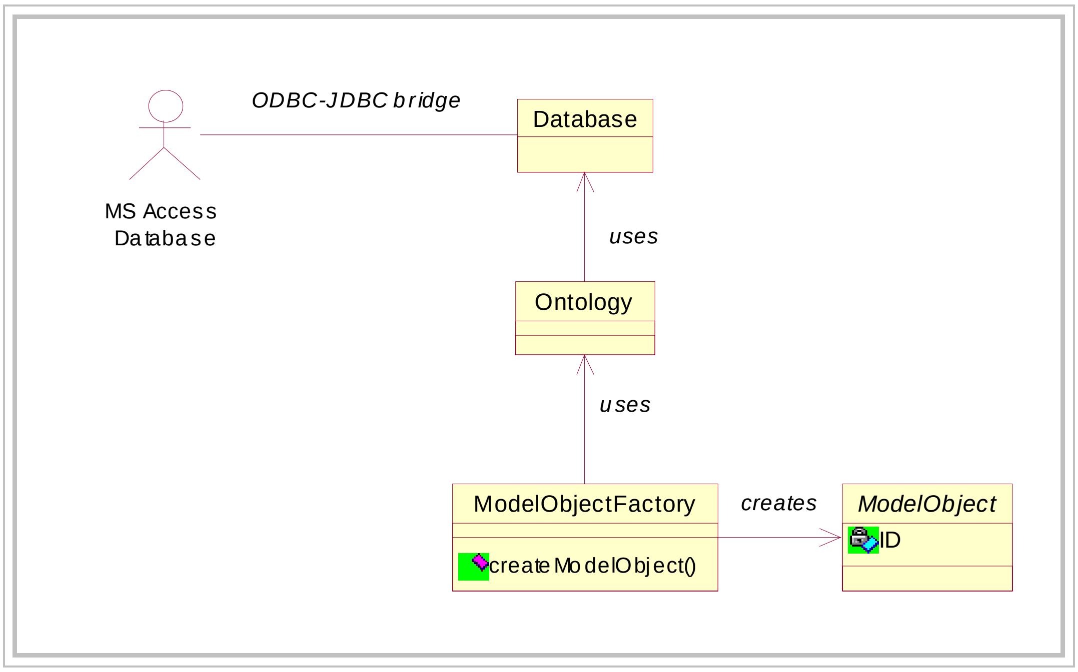


Figure 7-4 UML Class diagram of ModelObjectFactory.

7.3 VIRTUAL EXPERTS

As discussed in chapter six, Virtual Experts need the shared Project Model to search for Patterns (subsets), which are compliant with the conditions specified by the Meta-Knowledge. This section discusses the implementation issues related to the search for Patterns in the Project Model. Furthermore, this section discusses the implementation of Knowledge Components and the access of legacy applications through the Web as well as introducing a Knowledge Component editor.

7.3.1 Searching for Patterns

The Java Expert System Shell (JESS) enables the search for Patterns in the Project Model. It is able to crosscheck facts against conditions in a 'many-to-many' pattern recognition approach. To use this shell, the Project Model is converted into facts and the meta-knowledge is converted into conditions. Every Model Object is translated into a fact with a predefined structure using the following syntax:

Model object <its ID> semantic <its semantic>

Figure 7-5 The syntax for a Model Object

A Relation Object is translated into a fact using the following syntax:

RelationObject <relation ID> semantic < relation semantic> parent <parent ID> child <child ID>

Figure 7-6 The syntax for a Relation Object.

Figure 7-6 illustrates an example of several facts based on Building Objects and Relation Objects.

BuildingObject 1 semantic Building
BuildingObject 2 semantic Storey
RelationObject 3 semantic DecomposeRelation parent 1 child 2

Figure 7-6 An example of Facts based on the Project Model.

Rules are used to search the facts. These rules consist of a conditional part, called the 'Left Hand Side' (LHS) and a conclusion part, called the 'Right Hand Side' (RHS). The Meta-Knowledge is expressed in the LHS. Figure 7-7 is an example of a simple meta-knowledge expressed by a rule.

ModelObject ?building semantic Building
=>
(conclusion ?building)

Figure 7-7 An Example of a rule that can be used to reason on the project information.

The LHS contains a fact with a variable (?building). The JESS is able to find compliant facts. The meta-knowledge becomes more complex when the LHS contains more interrelating facts with more variables (Figure 7-8).

```
ModelObject  ?a semantic Building
ModelObject  ?b semantic Storey
RelationObject ?c Parent ?a Child ?b
=>
(conclusion ?a)
```

Figure 7-8 A rule with a complex Meta-Knowledge.

When a compliant set of facts is found, the rule can be executed enabling the Virtual Expert to create a Behaviour Object using the found pattern (facts).

7.3.2 Knowledge Components

The conclusion part (RHS) of a JESS rule activates a Virtual Expert to create a Behaviour Object. Therefore, the JESS is extended with several new commands for linking the Meta-Knowledge to the Virtual Expert. Furthermore, extra commands are implemented for extra constraints on the LHS (Figure 7-9).

Conditional constraint

```
(defrule DecompBuilding " generated by the kbe "
(BuildingObject ?a&:(eq 0 (instanceOf ?a Building)) semantic ?b)
(Body ?c)
(Loop ?d)
(RelationObject ?e semantic HasShape parent ?a child ?c)
(RelationObject ?f semantic BoundedBy parent ?c child ?d)

=>
```

Activation command

```
(addAction " adding facades to the building" "DecompBuilding"
BuildingObject v1 semantic Façade
RelationObject v2 semantic HasShape parent v1 child ?d
end)
)
```

Figure 7-9 New JESS commands.

Figure 7-9 illustrates a Knowledge Component with two new commands. The first command is ‘InstanceOf’ (the object has to be of a certain type) and adds an extra constraint to the LHS. In this case, the ‘BuildingObject ?a’ has to be of the type Building. The second command is called ‘AddAction’ and adds new Model Objects to the Project Model. In this case, a new Model Object with the semantic ‘Façade’ will be created that includes the relation ‘Has Shape’ between this new Model Object and the existing Loop Object.

7.3.3 Accessing Legacy Applications

Servlets are used to interface between Behaviour Objects and legacy applications residing on a server. In the shared Project Model, the Behaviour Object uses a legacy application interface, which accesses the servlet by using an URL. Furthermore, the legacy application interface sends the necessary input to the legacy application and retrieves the output through the servlet. Figure 7-10 shows a UML Class diagram of this implementation.

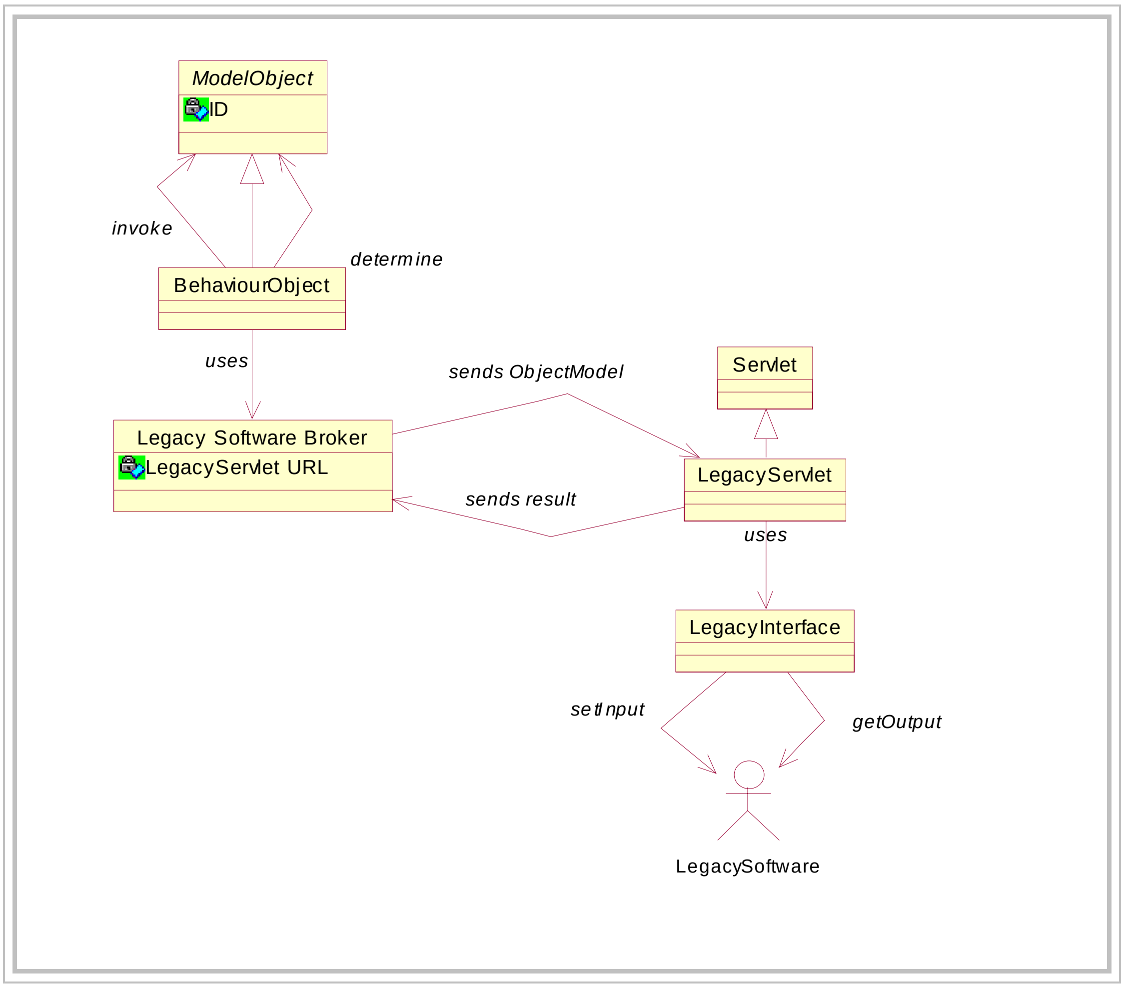


Figure 7-10 UML Class diagram of the implementation of a Behaviour Object connecting to a legacy application through the servlet.

The servlet retrieves the input, which consists of Java objects and deals with the communication with the legacy application in order to send the output results back to the Behaviour Object.

7.3.4 Knowledge Component Editor

Java2D is used to create a Knowledge Component editor. This editor visualises the Knowledge Components and especially the Meta-Knowledge that contains a network of Model Objects (Figure 7-11). The ontology of the DSS is used for providing the definitions of the Model Objects in this network. In this network, constraints can be specified for the semantics, the property values and the cardinality of the relations. Furthermore, this editor supports the creation of knowledge models by using the scripting language Jython.

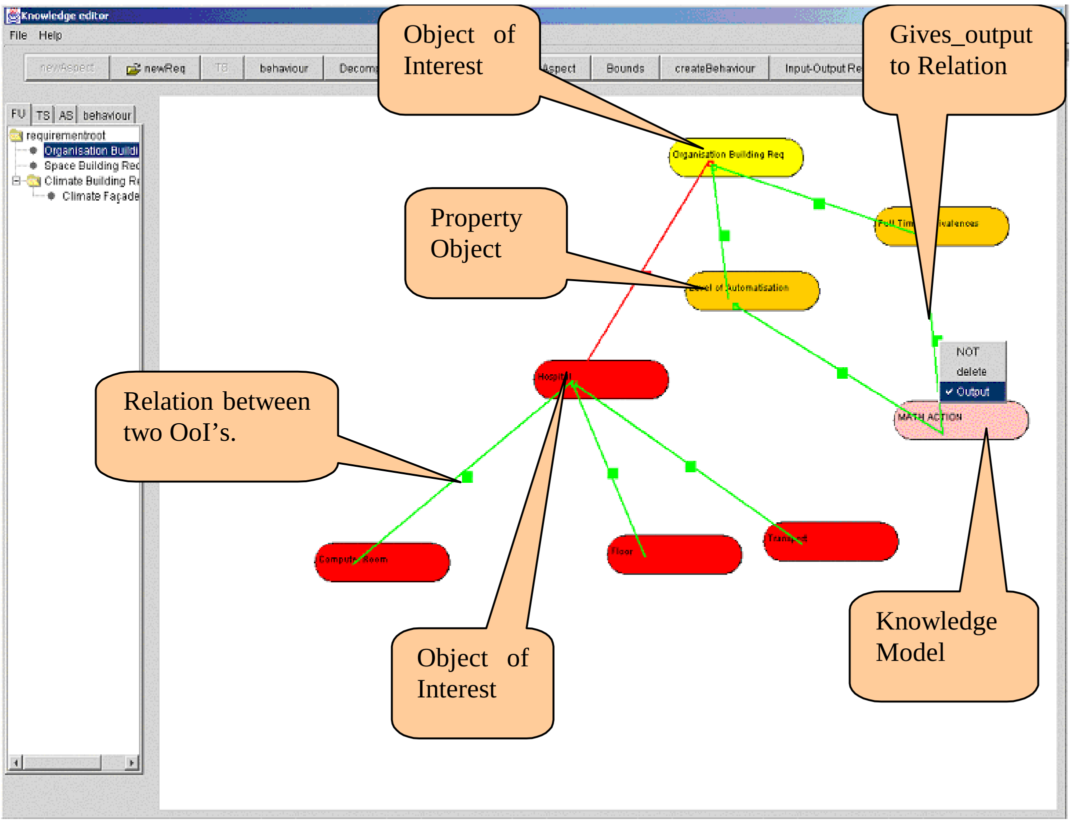


Figure 7-11 Screenshot of the Knowledge Component editor.

7.4 USER INTERFACE OF THE DSS

Two different user-interfaces (2D and 3D) are created that provide standard functionalities for the DSS such as the visualisation of and interaction with the Project Model. Furthermore, this section discusses the user-interfaces developed for and used by Virtual Experts.

7.4.1 2D User-Interface

Java2D is used for representing 2D geometry. OoI's capturing shape related information are defined in the ontology. A 'Mapper Object' uses these OoI's to create the Java2D content, which can be visualised.

7.4.2 3D User-Interface

To deal with the geometry of objects, shape related objects are defined in the ontology. A ‘Mapper Object’ uses these shape related objects to create the 3D content. The 3D content is then visualised using Java3D and Jun4Java (Figure 8-1)

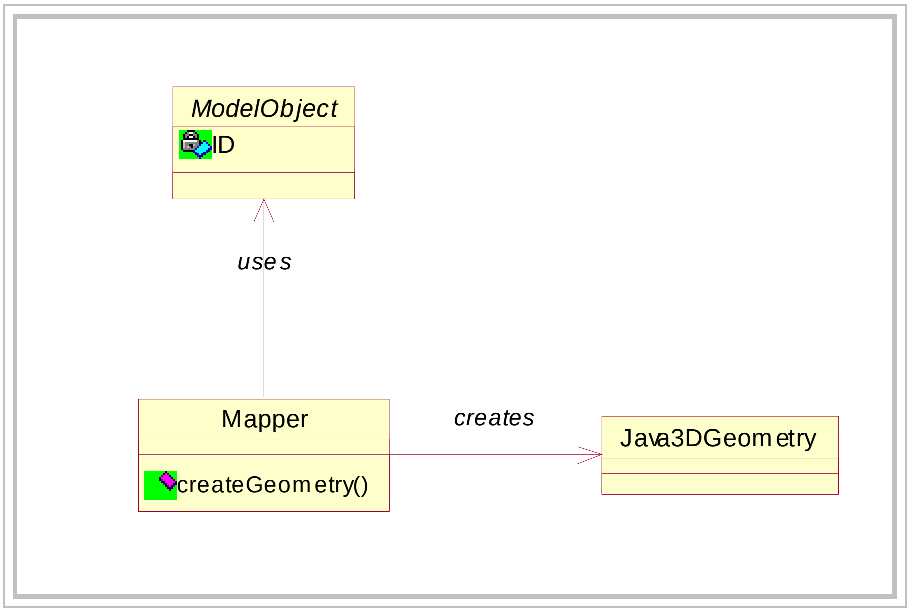


Figure 7-12 UML Class diagram of the ‘Mapper Object’

7.4.3 User-Interface for Virtual Experts

Virtual Experts are able to communicate with the client of the DSS using their own user-interfaces that are enabled by Java classes. Virtual Experts, residing on different web locations, can interface with the client because Java is able to distribute and load Java classes at runtime.

7.5 CONCLUSIONS

The Project Model, Behaviour Objects and the ontology are implemented and can be shared over the Internet. The ontology is open and its content can easily be modified. Consequently, the ontology supports the creation of project-type-specific DSS’s.

Also the other parts of the conceptual architecture are implemented that include Virtual Experts, Knowledge Components and Pattern search mechanisms.

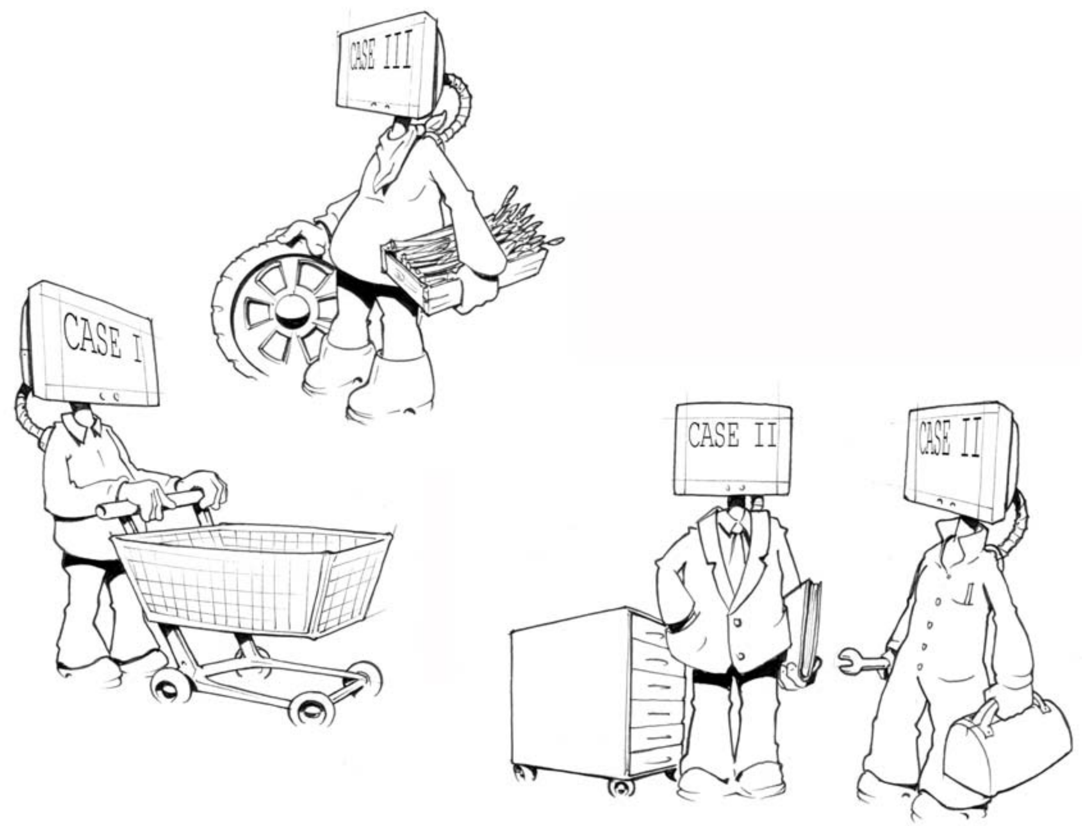
User interfaces visualising shape related objects in 2D and 3D are created in order to support the DSS.

To support the creation of a project-type-specific DSS, an ontology editor and a Knowledge Component editor are created. Project-type-specific Demand Support Systems can be created by using these editors, without having to implement or change the conceptual architecture or the user interfaces.

8

Case Studies

This chapter discusses three case studies that test the usability of the architecture and demonstrate the potentials of the Demand Support System.



Using one architecture, three project-type-specific Demand Support Systems using different Virtual Experts are developed.

8.1 INTRODUCTION

Three different case studies are carried out using the prototype implementation of the Demand Support System. Each case study focuses on a specific type of project and uses its own ontology and Knowledge Components. Using the ontology and the Knowledge Components, a ‘project-type-specific’ Demand Support System can be created. This chapter describes the three prototype Demand Support Systems.

8.2 CASE I: SHOPPING MALL

A shopping mall Demand Support System (DSS) has been created to test the proposed conceptual architecture and to demonstrate the advantages of a DSS during the inception phase of shopping mall projects.

8.2.1 Ontology of a Shopping Mall

The shopping mall ontology contains a Shopping Mall Object that can be decomposed into other objects such as shops, halls and atriums (Figure 8-1). In addition the ontology specifies the default decomposition behaviour of these objects by assigning predefined Behaviour Objects to the properties of these objects. In other words, the ontology contains information pointing to predefined Behaviour Objects that need to be created and assigned when a Model Object is decomposed. For example a predefined Behaviour Object can be attached to the property ‘area’ of a Model Object in the ontology. This Behaviour Object specifies that the value of this property is equal to the sum of all ‘area’ properties of the decomposition Model Objects.

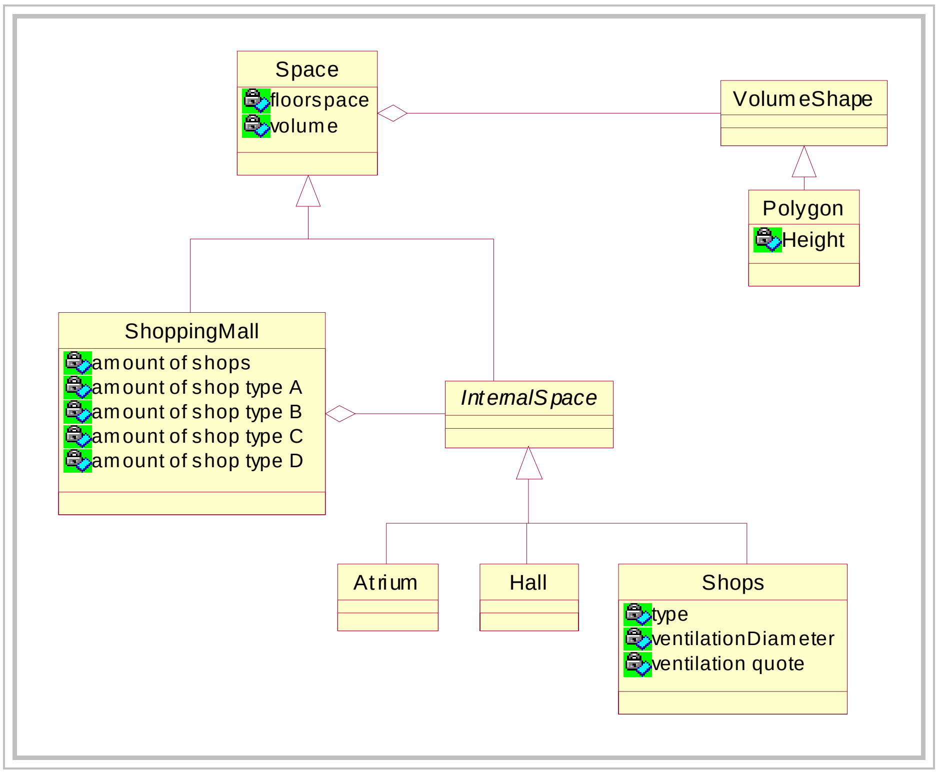


Figure 8-1 A UML Class diagram of the simplified ontology for a shopping mall.

8.2.2 Knowledge for Shopping Malls

Guidelines are available about the number of shops in a shopping mall. Shops can be classified based on their maximum area size. In this example, four types of shops are identified (Figure 8-2). When the total area for the shopping mall is known, as well as the percentage and the area of each shop type, then the number of shops for each shop type can be calculated.

Shop types	Area		Percentage of shop mall
A	75	m^2	30%
B	150	m^2	30%
C	600	m^2	25%
D	1800	m^2	5%
General space	-		10%

Figure 8-2 Shop types in a shopping mall.

It is also possible to calculate the diameter required for the ventilation shaft (Figure 8-3) that is based on the ventilation quote for each shop (Figure 8-4).

Shop area	Diameter ventilation shaft	Diameter ventilation Canal
	n=3	n=5
<250 m^2	Ø 315 mm	Ø 450 mm
250-1000 m^2	Ø 450 mm	Ø 500 mm
>1000 m^2	Ø 500 mm	Ø 650 mm

Figure 8-3 Ventilation information for reaching a certain ventilation quote.

Ventilation quote for regular shops	*n=3*
Ventilation quote for catering industry related shops	*n=5*

Figure 8-4 Ventilation quote for regular and catering industry related shops.

8.2.3 The Shopping Mall DSS

The DSS can be used when information about the shape and the location of a shopping mall is available (Figure 8-5).

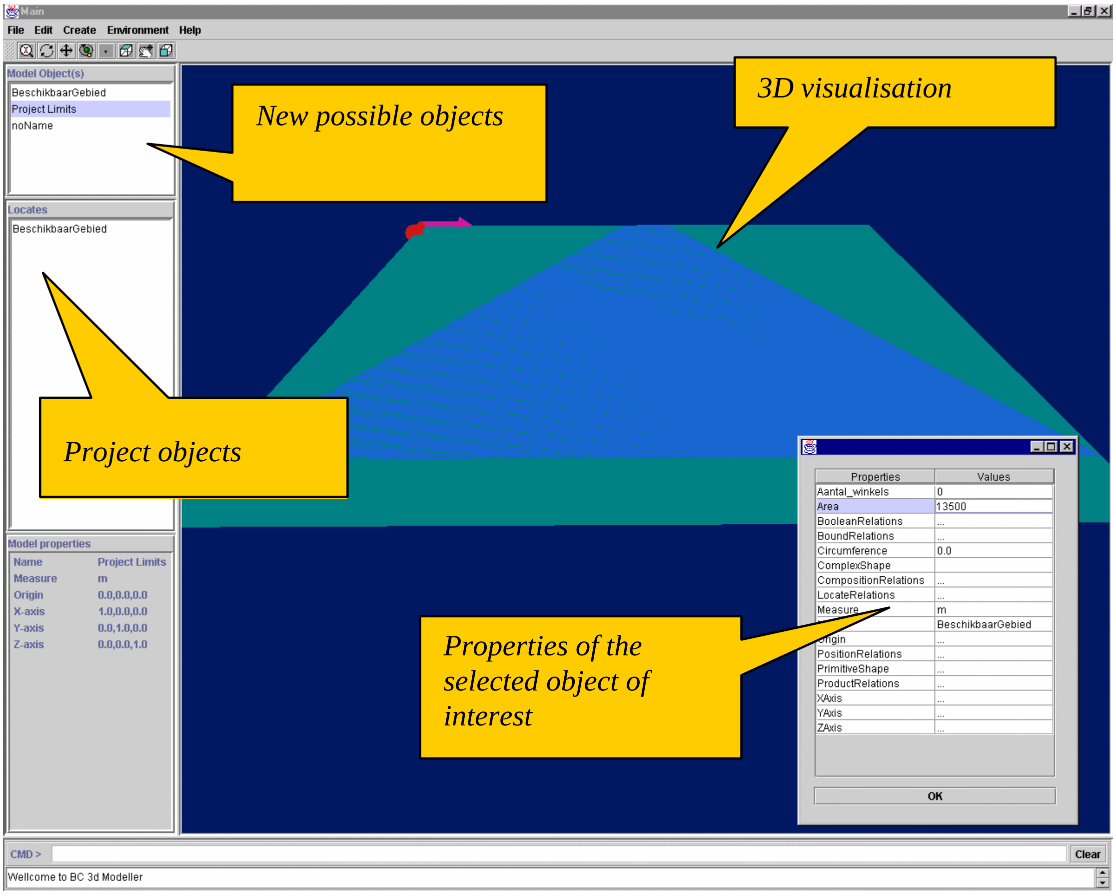


Figure 8-5 A screenshot of the shopping mall DSS showing the available area for the new shopping mall.

Based on the geometry of the shopping mall the number of shops can be calculated (Figure 8-6).

Shop types	Area	Number of Shops
A	**675** m^2	9
B	3375 m^2	22
C	4050 m^2	7
D	4050 m^2	2
General space	1350 m^2	
Total	13500 m^2	40

Figure 8-6 A first estimation of the number of shops in the shopping mall.

Each shopping mall can be decomposed into objects such as halls, atriums and shops. A Virtual Expert uses this project information to add Behaviour Objects to determine the ventilation information for every object such as the ventilation quote and the diameter of the ventilation shaft. For example, a bakery shop can be created of the type B. The Virtual Expert can insert a ventilation shaft automatically, because a bakery is defined in the ontology as a subtype of the catering industry (Figure 8-7). In this figure, a property frame shows that the selected shop has an area of 125 m^2 and a ventilation shaft of 450 mm.

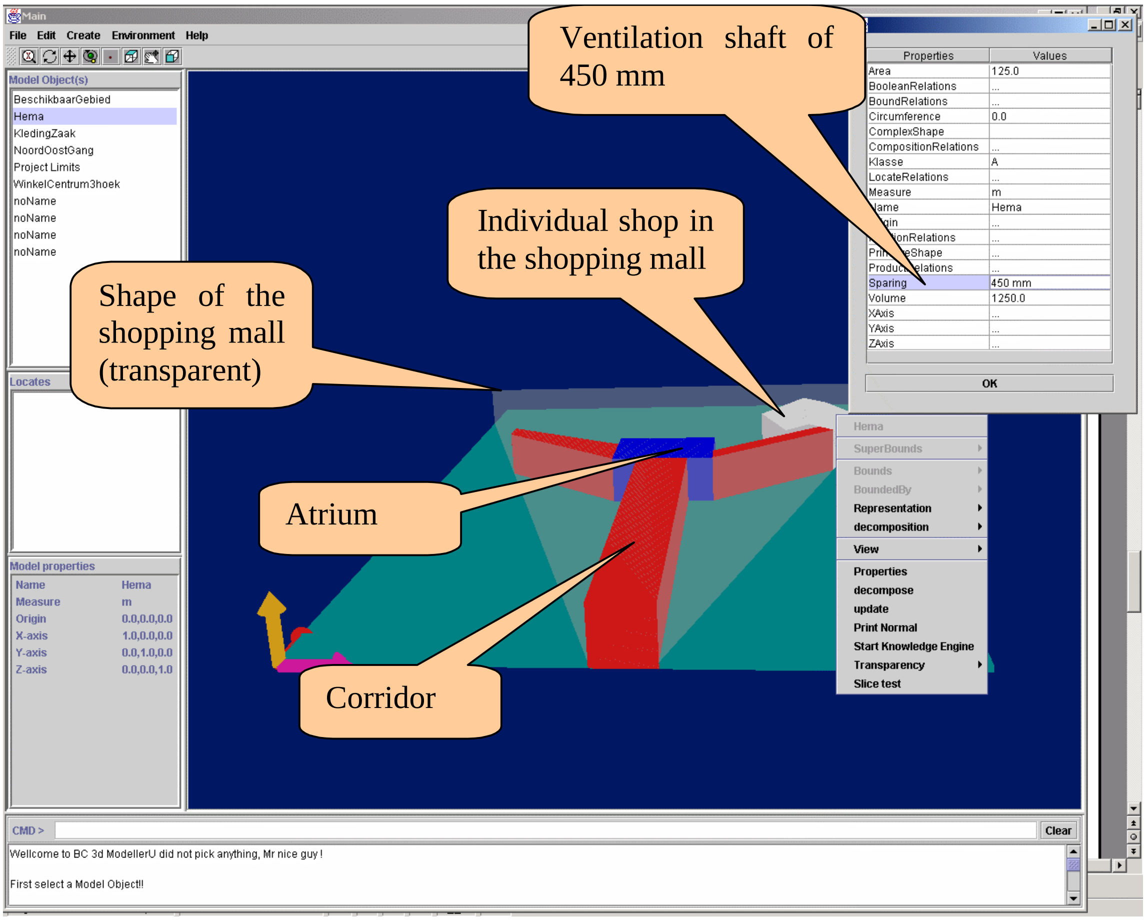


Figure 8-7 A screenshot of the shopping mall DSS.

When the client creates a shaft of for example Ø 315 mm, a Behaviour Object reports a conflict situation because the shaft needs to be Ø 450 mm. Another scenario applies when the client joins two separate shops of less than 150 m2 each, into one larger shop (Figure 8-8). This will result in a different number of ‘type B shops’ (from 22 to 20) and ‘type C shops’ (from 7 to 8). As a result, the total number of shops will be reduced (39 instead of 40). In addition, Behaviour Objects change the necessary ventilation shaft from Ø 450 mm to Ø 500 mm. In Figure 8-8, the property frame shows that the new shop has a ventilation shaft of Ø 500 mm.

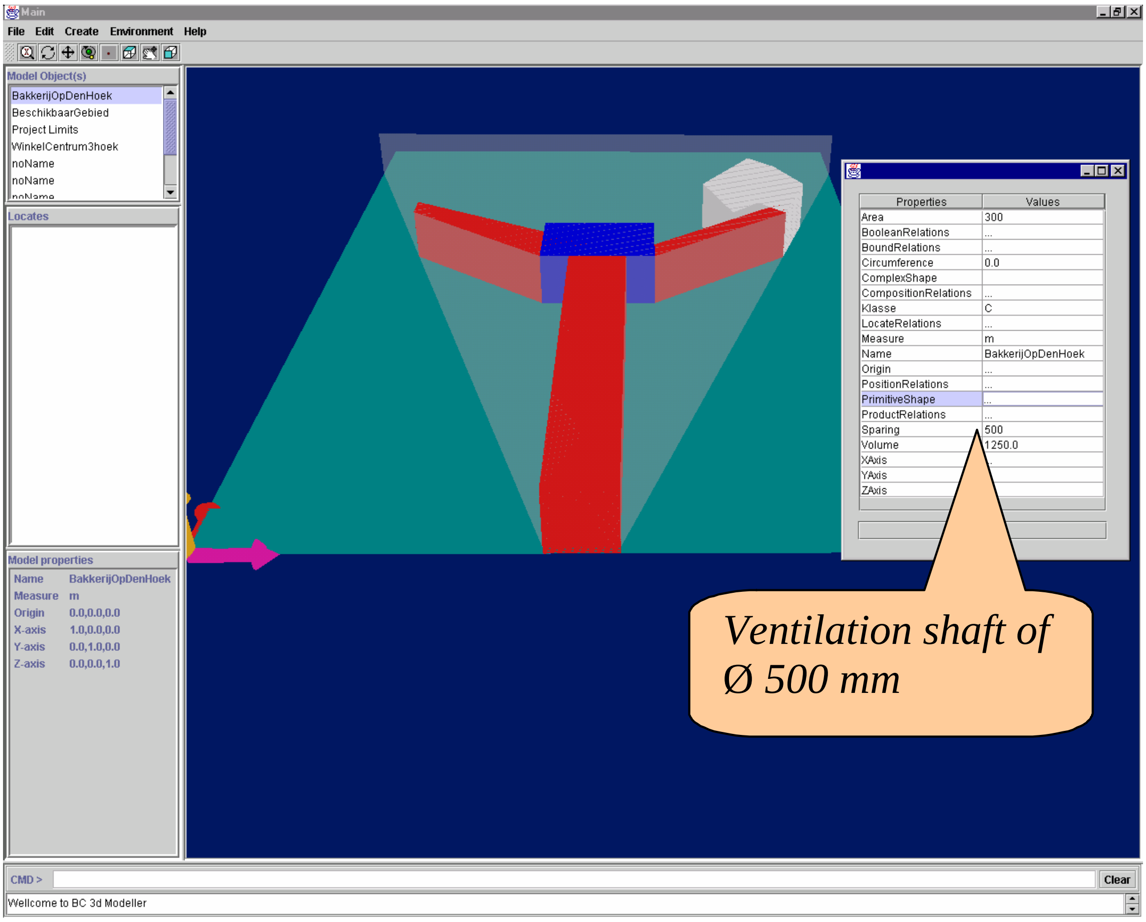


Figure 8-8 A screenshot of the shopping Mall DSS after joining two shops.

8.2.4 Conclusions

Based on a simple ontology and only a limited set of rules, a shopping mall DSS is created without changing the architecture or the user interfaces. Some project-type-specific information can be visualised in the shopping mall DSS. The DSS allows instant demonstration of changes and their consequences.

Several demonstrations of the system have been carried out in practice and have been received positively. The audience could relate to the problems that the system was able to cope with and confirmed the need for such a system.

8.3 CASE II: THE OFFICE BUILDING SIMULATION CASE

The focus of this case study is on simulating heat and energy consumption using a legacy application during the inception phase. In order to use the legacy application, the Virtual Experts will try to generate the necessary information automatically (Guided Extension).

8.3.1 Ontology of an Office Building

The shape model defined in the ontology for this DSS is discussed seperately from the office building ontology.

Office Building Ontology

Three higher-level objects (Physical Object, Requirement Object and Aspect Object) are defined in the ontology (Figure 8-9). Physical Objects describe design solutions such as site, road, building, storey, HVAC installation, outer wall, inner wall, et cetera. Aspect Objects describe systems such as ‘climate system’, ‘heating system’ and ‘structural system’. Requirement Objects describe requirements of space, comfort and such. All Requirement Objects are by definition linked to Aspect Objects specifying the domain of the requirement. In addition, Requirement Objects are also related to Physical Objects describing the design solution on which the requirement can be posed.

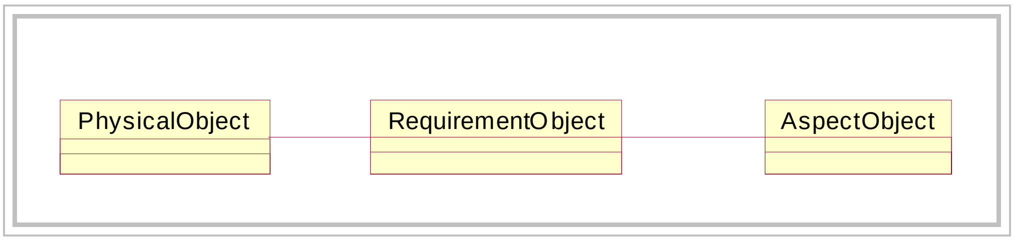


Figure 8-9 Three types of semantic objects in the ontology.

Shape Model

To support the evolution of the shape, a certain consistency between different levels of detail is necessary. Therefore, a shape model is used that uses four types of objects: point objects, line objects, face objects and volume objects. These four types of objects have two types of relations: (1) bounding relations and (2) location relations. Objects can be composed of sets of smaller objects that follow the same classification. Figure 8-10 illustrates the ontology for the shape model.

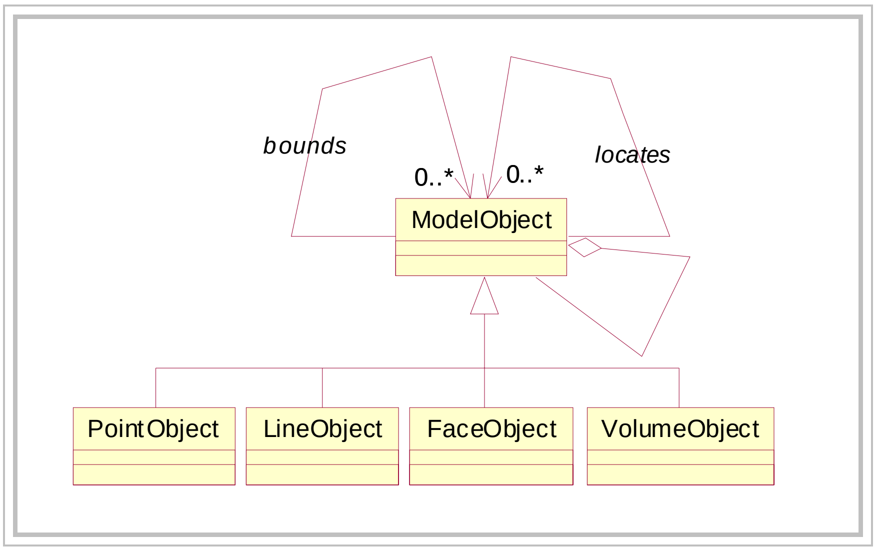


Figure 8-10 Four types of objects Point Object, Line Object, Face Object and Volume Object with 2 types of relations: 'bounds' and 'locates'.

The idea is that all real world objects (Beam, Room, House, Road, Wall, etc.) can be grouped according to these four objects. For example, the class Wall can use the Face Object, the class Road can use the Line Object and the class Building can use the Volume Object.

For this prototype, the development related to the shape of building needs to be supported. Starting with the idea that the shape of a building can be modelled using a Volume Object, all other necessary objects can be generated (e.g. Face Objects, Line Objects and Point Objects). At a certain point, Wall Objects can be related to the existing Face Objects. Later on, a wall may be modelled with a Volume Object. The Face Objects on the higher-level walls can be re-used for the new Volume Object (Figure 8-11).

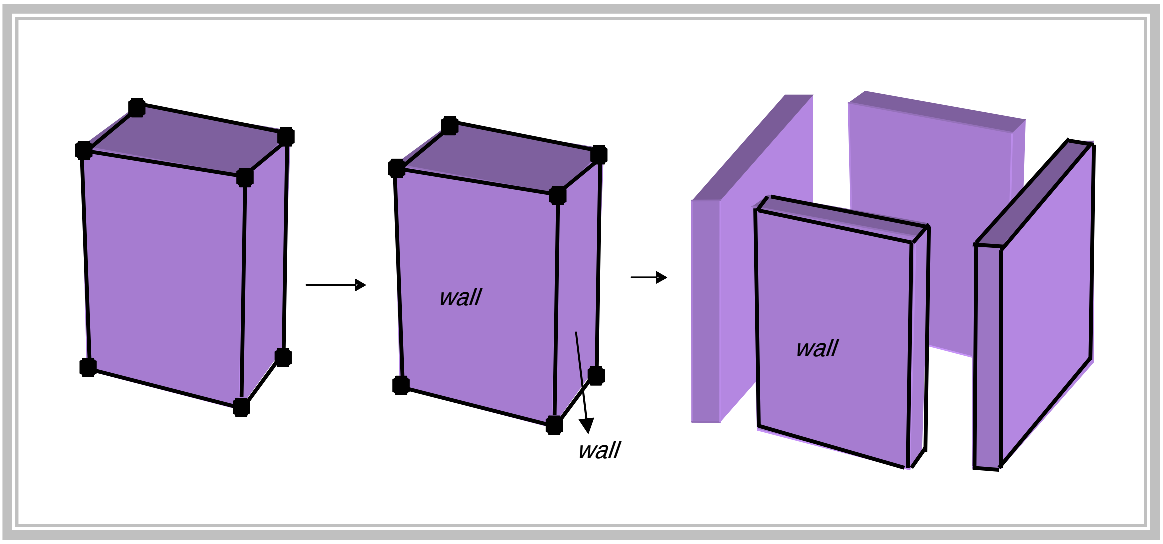


Figure 8-11 Evolution of the shape of a building.

In addition, the DSS facilitates the parameterisation of the shape. For instance, a box can be defined by three parameters length, width and height. Subsequently, a Behaviour Object can be used to influence the Point Objects using these parameters.

8.3.2 Knowledge for Simulating Office Buildings

The Knowledge Component editor is used to capture knowledge for this DSS. Also a legacy application is used.

Knowledge of an Office Building

In this case study, different Virtual Experts are used:

- Facility management Expert capable of generating space requirements based on organizational requirements

- Architect, capable of decomposing a building automatically in stories and walls
- HVAC expert, capable of suggesting, optimising and evaluating building installations

Most knowledge is captured in simple rules. For example, the number of ‘full time employees’ is approximately 70% of the ‘total formation’. Some of these rules use If-Then statements. Figure 8-12 gives an example of such a rule capable of calculating the space requirements.

*Floor area for meeting rooms = 10 m^2 + f * ‘maximum available employees’.*

F= 0.9 when maximum available employees’<200
F = 0.8 when maximum available employees’>200

Figure 8-12 A rule using an If-Then construct for determining the floor area of a meeting room.

In addition, the automatic generation of extra information is implemented. Organisational objects describing the organisation in general, can be automatically decomposed into smaller objects like ‘Reception’, ‘Staff’, et cetera. The same approach has been used for general space requirements which can be decomposed into more specific objects like ‘toilets’, ‘meeting rooms’, ‘offices’ or ‘hallways’.

HVAC Legacy Application

An application called VA114 (Vabi 114) has been used to simulate the energy consumption of office buildings. This application is able to calculate the consumption of electricity, heating and cooling loads, gas usage and temperatures within the building during a certain period. The program is able to consider several installations and is based on a multi-room calculation where interaction is modelled between the rooms. Shadows of the building itself or shadows of surrounding buildings are also taken into account. The necessary input for this program includes detailed design information. Starting from the idea that this information is not available, the most important objects and properties for making an assumption of the energy consumption are identified. Figure 8-13 shows the identified objects and their relations. The communication model will be augmented with extra default values to provide the detailed information needed to run VA114. This means that default values and simple knowledge rules are used in combination with this HVAC model to generate the detailed input necessary for simulations. Figure 8-14 an example of results of such a simulation where score bars are included for easier interpretation of the results.

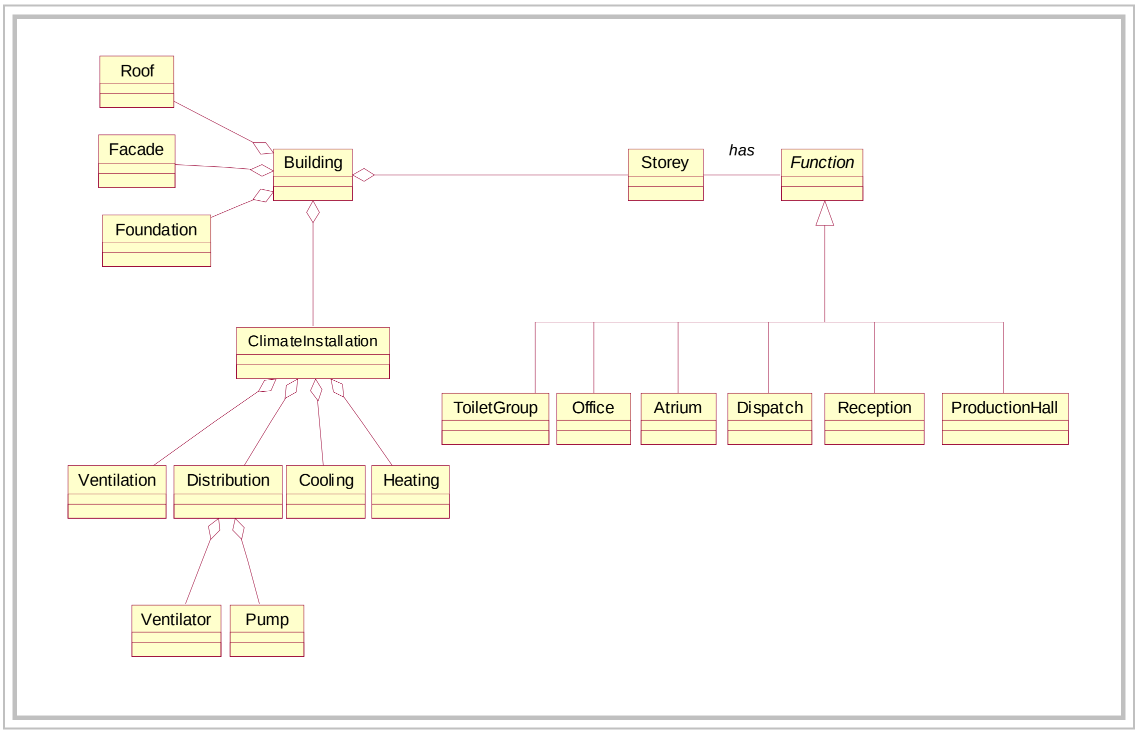


Figure 8-13 UML Class diagram of the simplified communication model for the HVAC simulation.

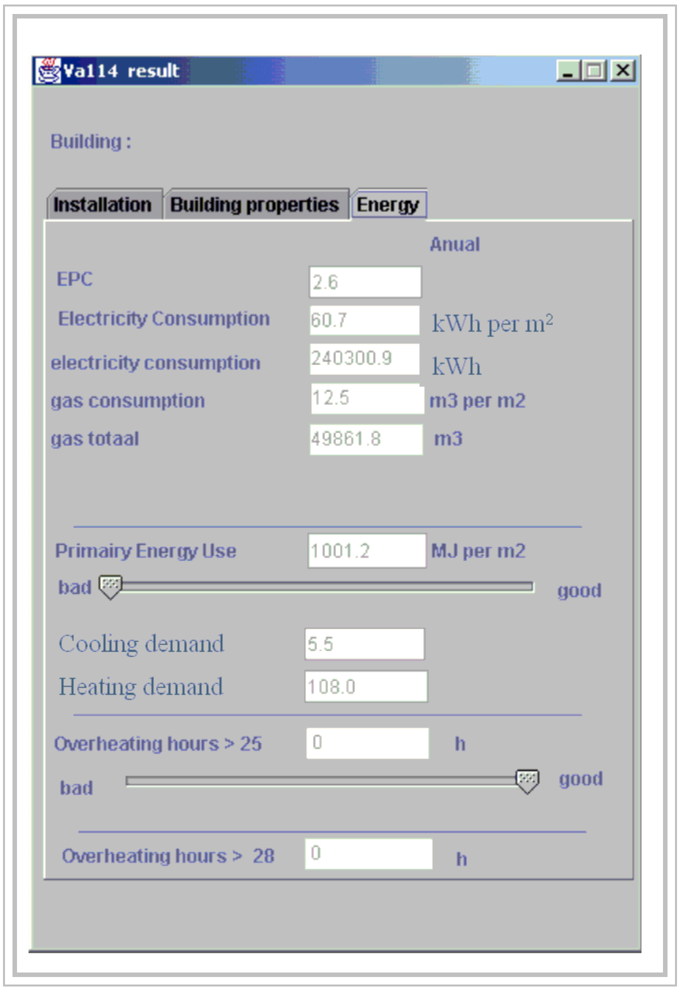


Figure 8-14 Screenshot of the simulation results.

A client may not be able to interpret the results. Therefore, it is necessary to have knowledge rules for interpreting these results. Figure 8-15 gives some examples of rules that can be used to support the interpretation of the simulation results.

Low = 30 kWh per m² Bruto Floor Area
High = 70 kWh per m² Bruto Floor Area

+2	*<=10*
+1	*10 < Electra <= 30*
0	*30 < Electra <= 50*
-1	*50 < Electra <= 70*
-2	*> 70*

Figure 8-15 Rules for interpreting the simulation results.

When the correct information is available in the Project Model, the legacy application can be consulted automatically. The HVAC Expert inserts a Behaviour Object to communicate with the server running legacy application.

8.3.3 The Office Building DSS

This section discusses the user-interfaces of the DSS, the development of the building spanning different levels of detail and the modelling of decisions.

The Requirements User-interface

A user-interface for the input of requirements uses a matrix where each cell represents the correlation of an Aspect Object and a Physical Object. The ontology provides for each cell, the possible Requirement Objects (Figure 8-16). The client can use the system in any order of preference. For example, the client can start with selecting space requirements and then the HVAC requirements, or vice versa.

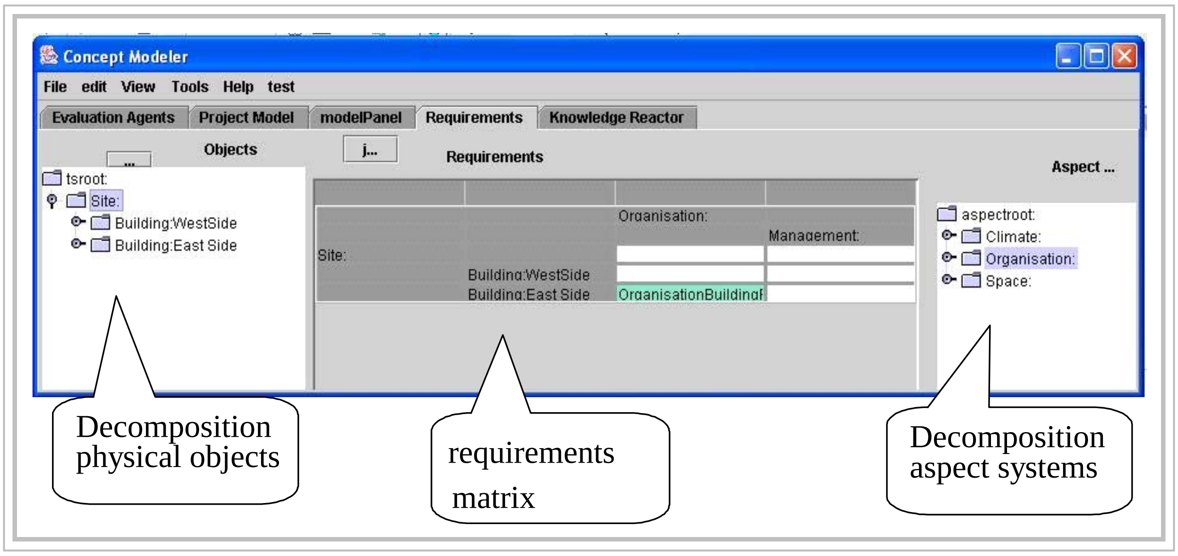


Figure 8-16 Screenshot of the requirements user-interface.

3D User Interface

Design solutions can be visualised and manipulated in a 3D environment. A tree view presents the decomposition structure of these design solutions. Furthermore, a property frame can be retrieved when selecting an object in the 3D environment or in the tree view. Objects for decomposing the selected object present themselves in the user-interface (Figure 8-17).

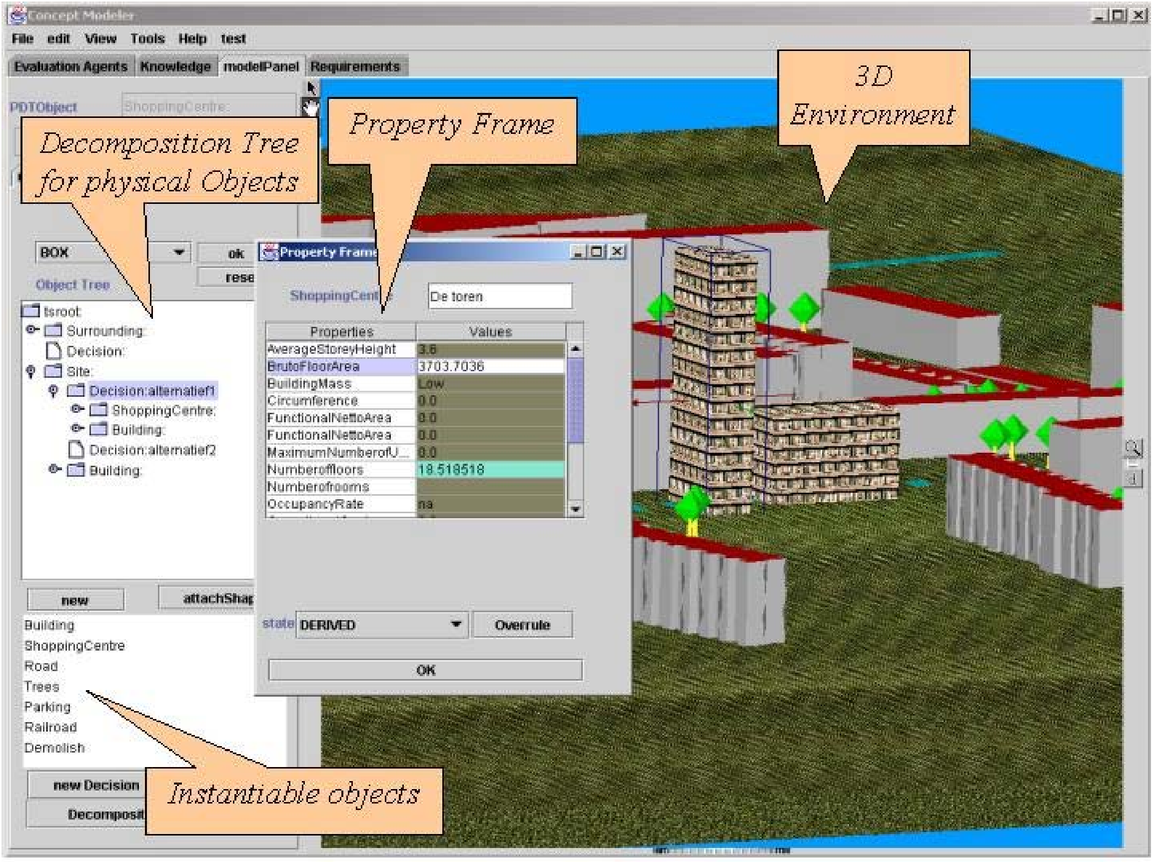


Figure 8-17 Screenshot of the user-interface showing a model of a building with its properties.

Levels of Detail

The building Virtual Expert can decompose Building Objects automatically into walls, floors and roofs (Figure 8-18). When a Building Object uses Volume Object, its Face Objects are recognized as walls, roofs and foundations (Guided Extension). The HVAC Virtual Expert is also able to guide the extension by adding the HVAC installation.

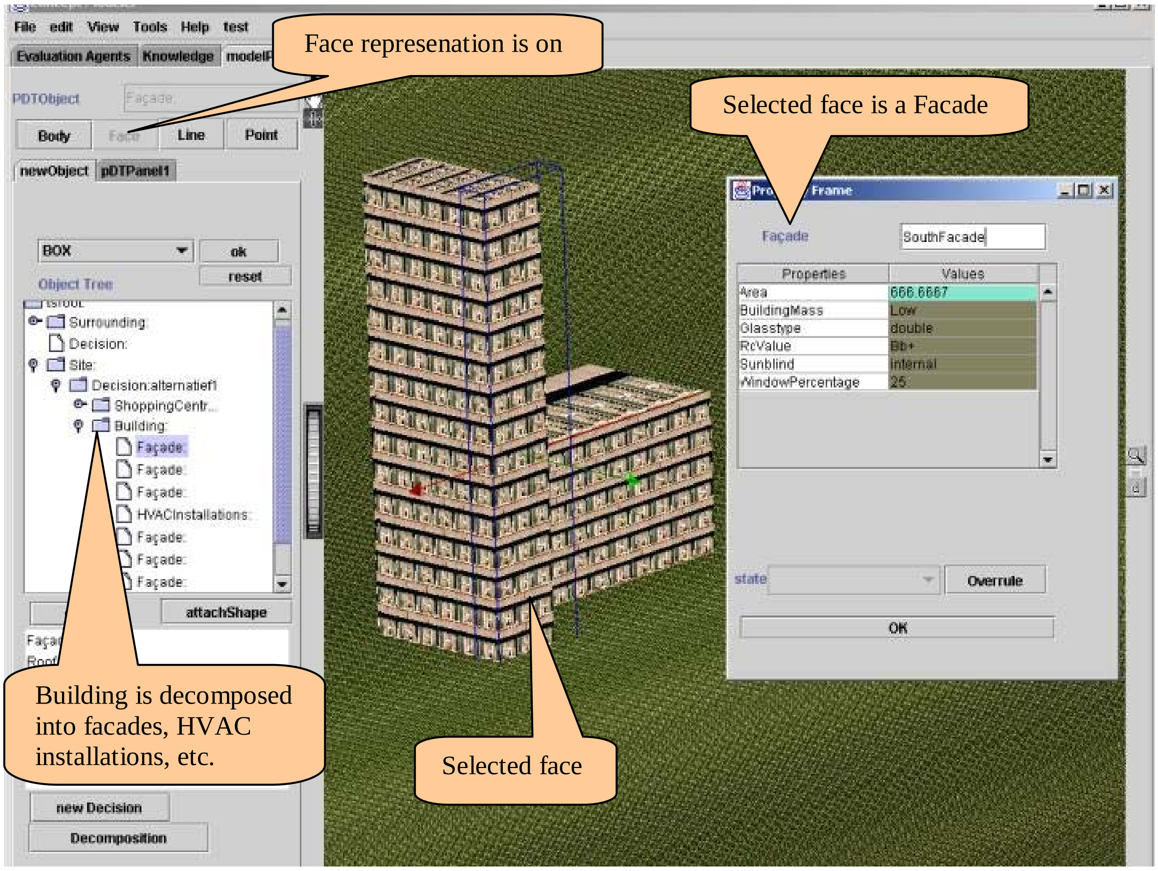


Figure 8-18 Guided Extension of a Building Object.

Multiple decompositions are also possible such as another decomposition of the building into different floors. Figure 8-19 illustrates this example. Behaviour Objects ensure that the storeys of the building are located on top of each other.

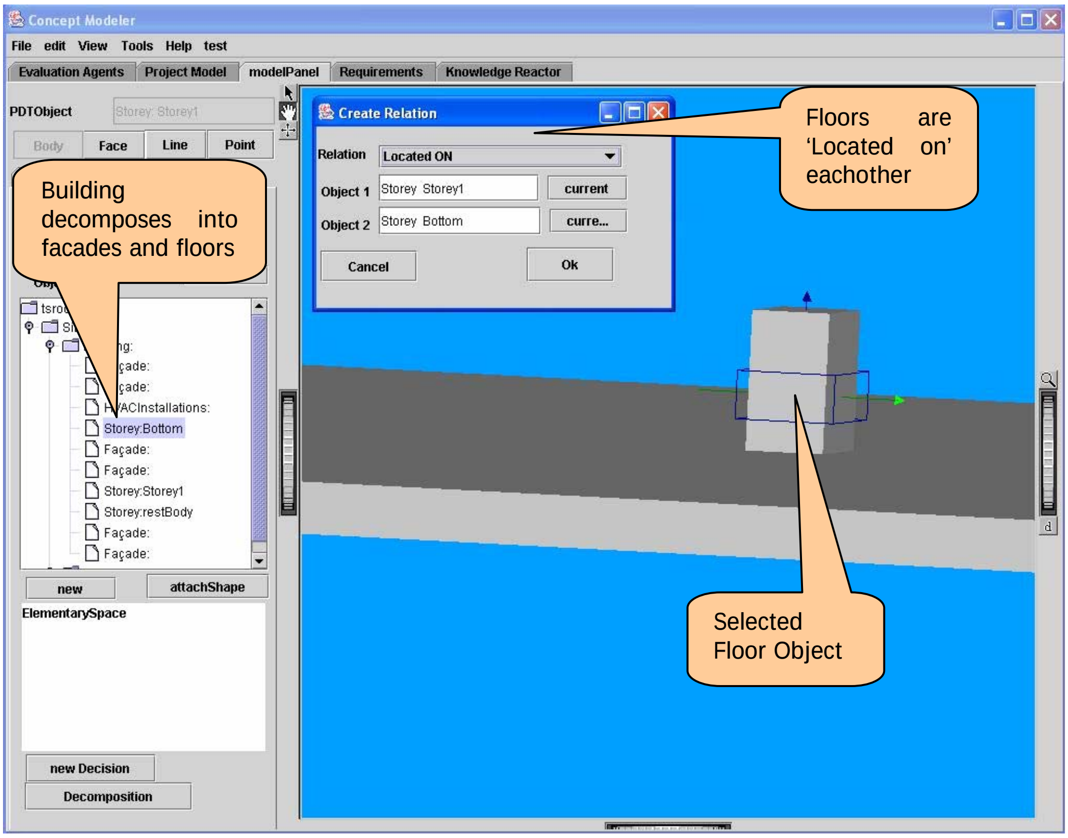


Figure 8-19 Decomposition of a building into several storeys, which are placed on top of each other.

Modelling of Decisions

Decisions can be made on several levels of detail, which is here defined as a choice between different alternatives. The DSS supports this by introducing a Decision Object, which can be turned on and off. On every level of detail, this object can be used to model alternative requirements and design solutions. Decision Objects are not only possible for modelling alternative concept design solutions but also for modelling a different set of requirements. Figure 8-20 illustrates an example where two different decisions are made. The first decision is on the site level and proposes a layout (by demolishing certain buildings). The other decision proposes a different shape for the new building.

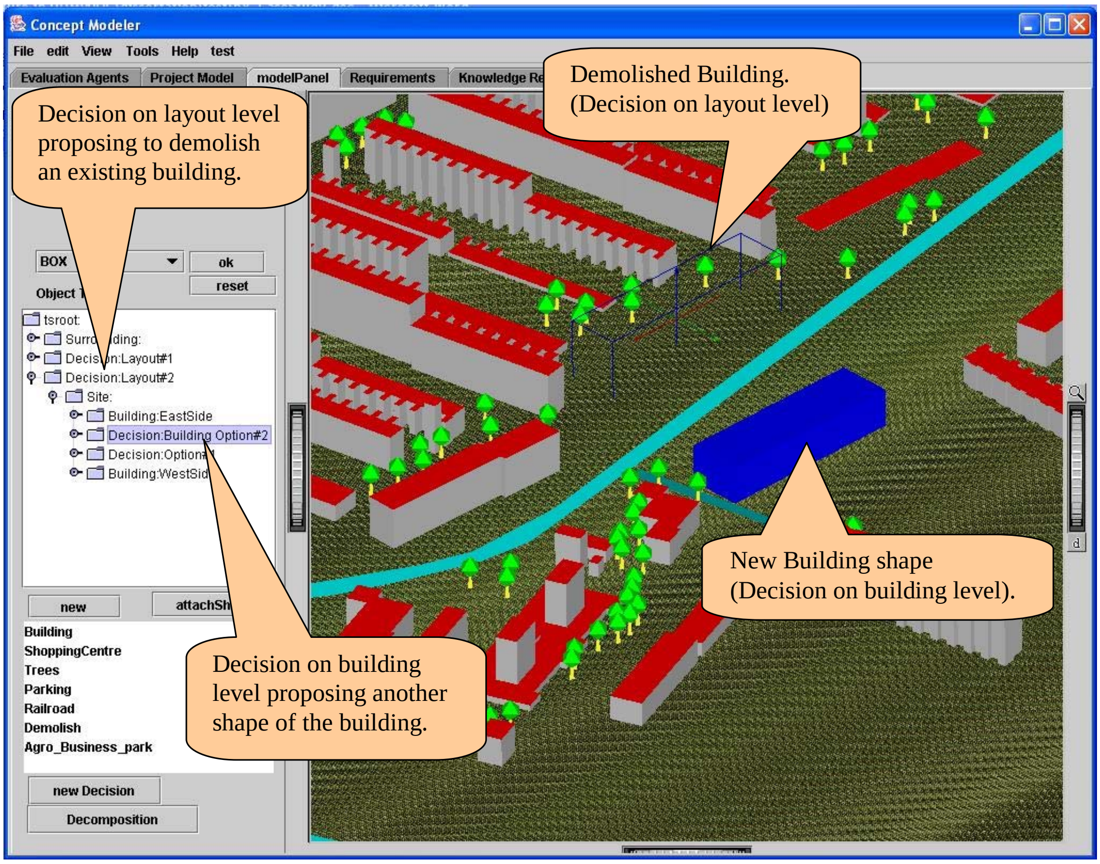


Figure 8-20 Decisions on different levels of detail.

8.3.4 Conclusions

This case study demonstrates the use of three Virtual Experts using knowledge related to facility management, building design and HVAC. Several knowledge representations are used like rules of thumb, If-Then constructs and legacy applications. The legacy application can be accessed automatically using Web services. The DSS is able to consult the legacy application by using Guided Extension as described in chapter 6. The detailed output that this legacy application generates is interpreted making it understandable for the client.

The memory imprint for searching Patterns scales up non-polynomial when inserting new Model Objects. This may form a problem when the Project Model is large. Distribution of the Virtual Experts over multiple computers only solves the problem partly. Therefore, other routines, which are less dependent on memory, are necessary for finding Patterns in de Project Model.

The insertion of Behaviour Objects and the adding new Model Objects was quite rudimentary. Whenever the user specifies a 'Building', the Virtual Experts automatically add installations, storeys, walls, roofs, et cetera. In addition, other Behaviour Objects for calculating volumes, areas, and locations are inserted at the same time. Some Behaviour Objects are necessary for keeping the model consistent and are therefore necessary. Other Behaviour Objects may not be desirable and consequently more control on the insertion of Behaviour Objects is necessary (or more control on the Virtual Experts). The design of Knowledge Components, especially the meta-knowledge offers a lot of freedom which influences *when* the Knowledge Component can be used. In order to get a suitable behaviour, the person that develops this Knowledge Component needs to have some modelling skills.

Regarding the capturing project-specific information, the DSS offers the client a certain degree of flexibility. For example, the client can work simultaneously with different requirements at different levels of detail. Therefore, the DSS is able to follow the design process by supporting the decomposition of the requirements but also by supporting the decomposition of the design solutions.

8.4 CASE III: AGRO BUSINESS PARK

In order to reduce transportation movements in the Netherlands, clustering of agricultural production and distribution facilities becomes important [NDL, 2000]. By forming a cluster, individual agricultural companies can increase their profit, for example by combining their distribution. To study these clusters, a DSS has been developed that is capable of analysing existing and new Agro Business Parks.

8.4.1 Ontology of an Agro Business Park

An Agro Business Park contains several agricultural companies, infrastructure (e.g. roads and parking spaces), terminals and utilities (e.g. water cleaning facilities, waste disposals, etc.). Figure 8-21 illustrates the top level of the ontology used in this case study.

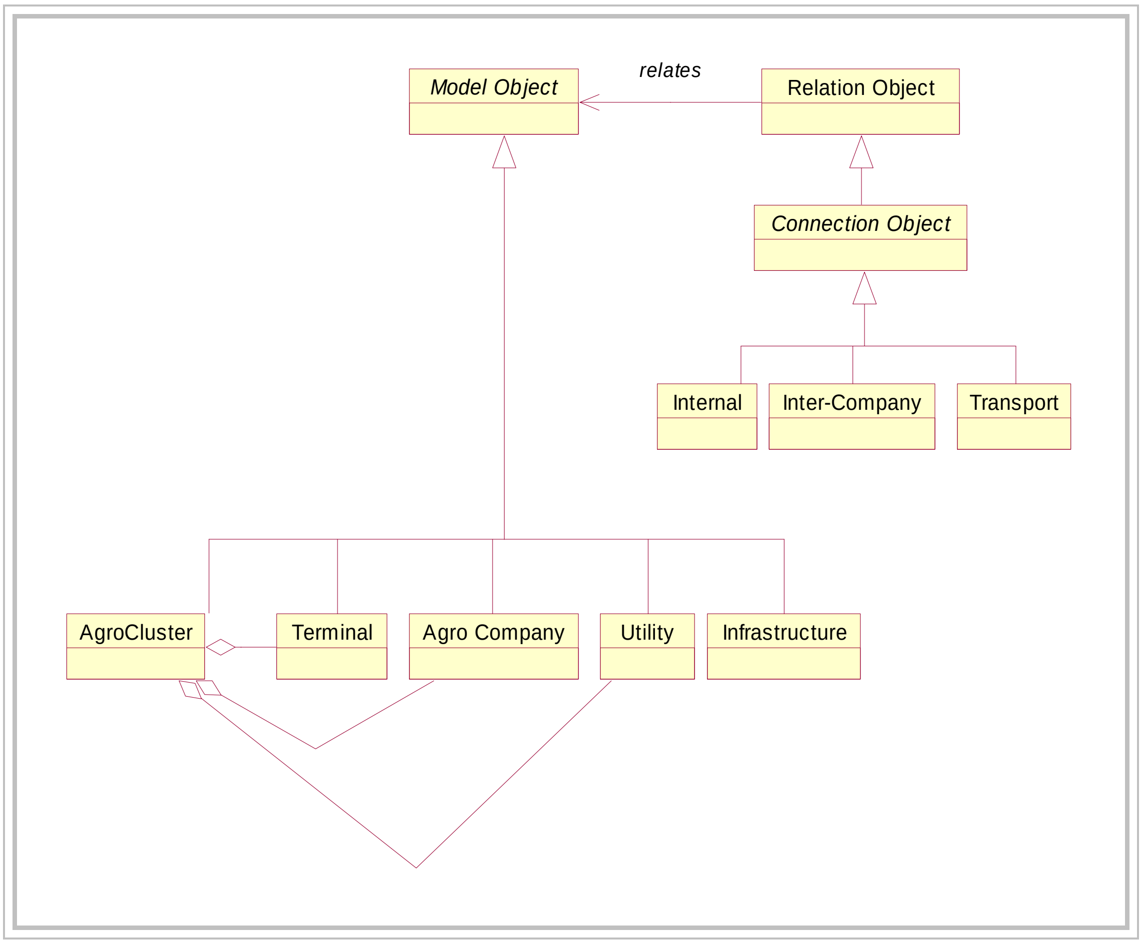


Figure 8-21 A simplified ontology for Agro Business Parks.

All Model Objects have input and output relations with a generic Connection Object enabling relationships between Model Objects. Several types of connections are defined such as Internal Connections, Inter-company Connections, Transport Connections, et cetera. These Connection Objects can be used to describe for example the requirements for arriving and departing transports.

Several types of agricultural companies are identified describing their space requirements. The latter requires Shape Objects such as polygons with height parameters, et cetera. Every Model Object is located by a Layer Object. In addition, every Model Object contains several Layer Objects containing Points for describing the contours of the shape (Figure 8-22). A Layer2D is used for objects where the height is not relevant such as roads and parking lots. A layer3D can be used for buildings, hangers, et cetera.

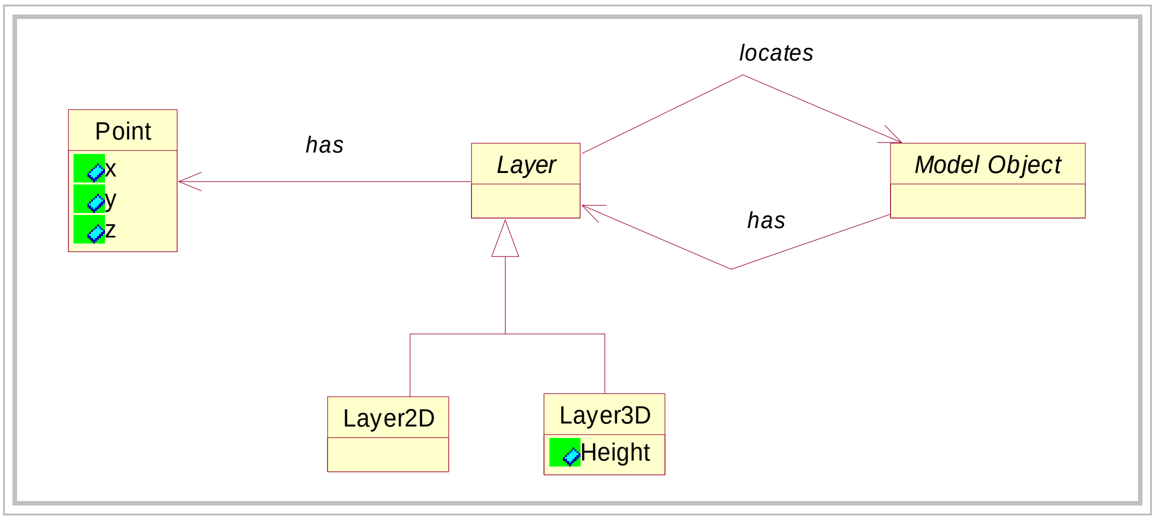


Figure 8-22 Simplified Shape Objects used for the Agro Business Park DSS

8.4.2 Agro Business Park Knowledge

The primary production of an agricultural company is taken into account to calculate the space requirements. For example, for a brewery, the amount of beer produced can provide sufficient information for estimating the space requirements that in turn can be decomposed into spaces for productions, terminals, offices, storage, et cetera.

8.4.3 The Agro Business Park DSS

An Agro Business Park comprises of many different buildings and areas surrounded by various neighbourhoods. Therefore, the shape of existing buildings and infrastructure can be imported in the DSS to allow further modelling of envisaged solutions (Figure 8-23).

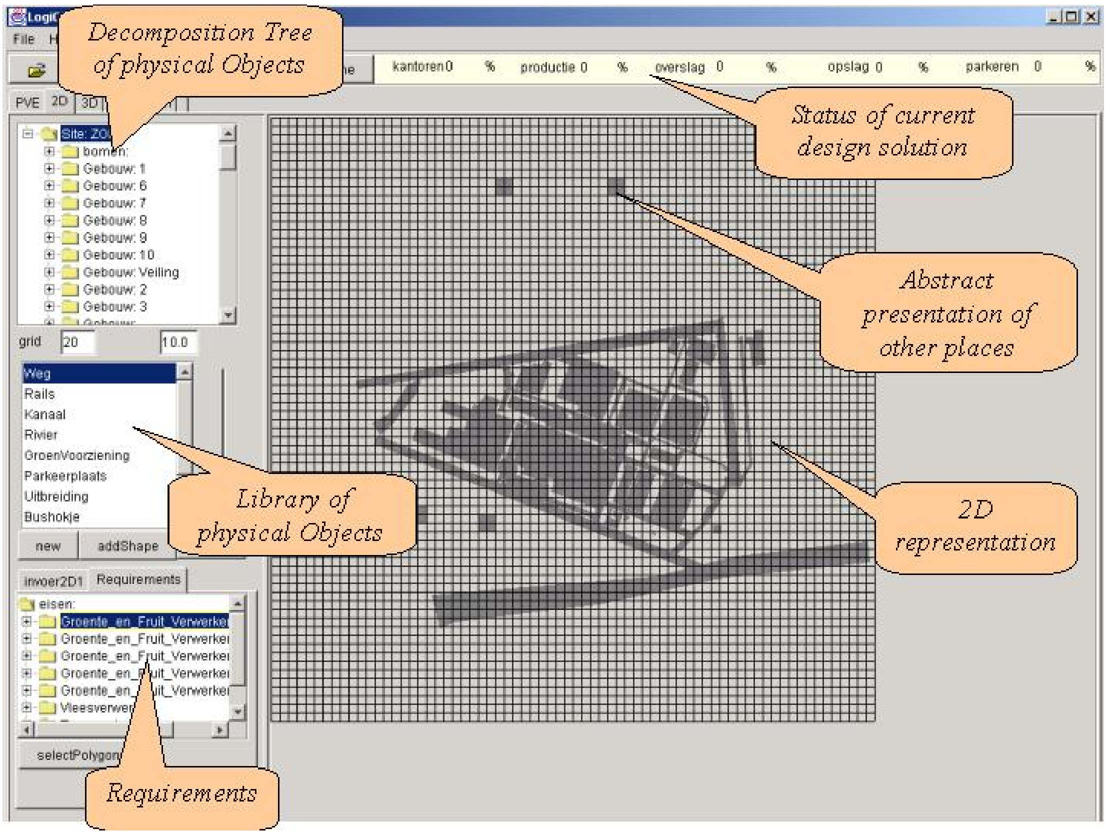


Figure 8-23 Screenshot of the application after importing a DXF spatial layout.

Extra information such as company names, addresses and type of industry can be attached to the objects. The DSS allows the transformation of a 2D model to a 3D model by adding height information (Figure 8-24).

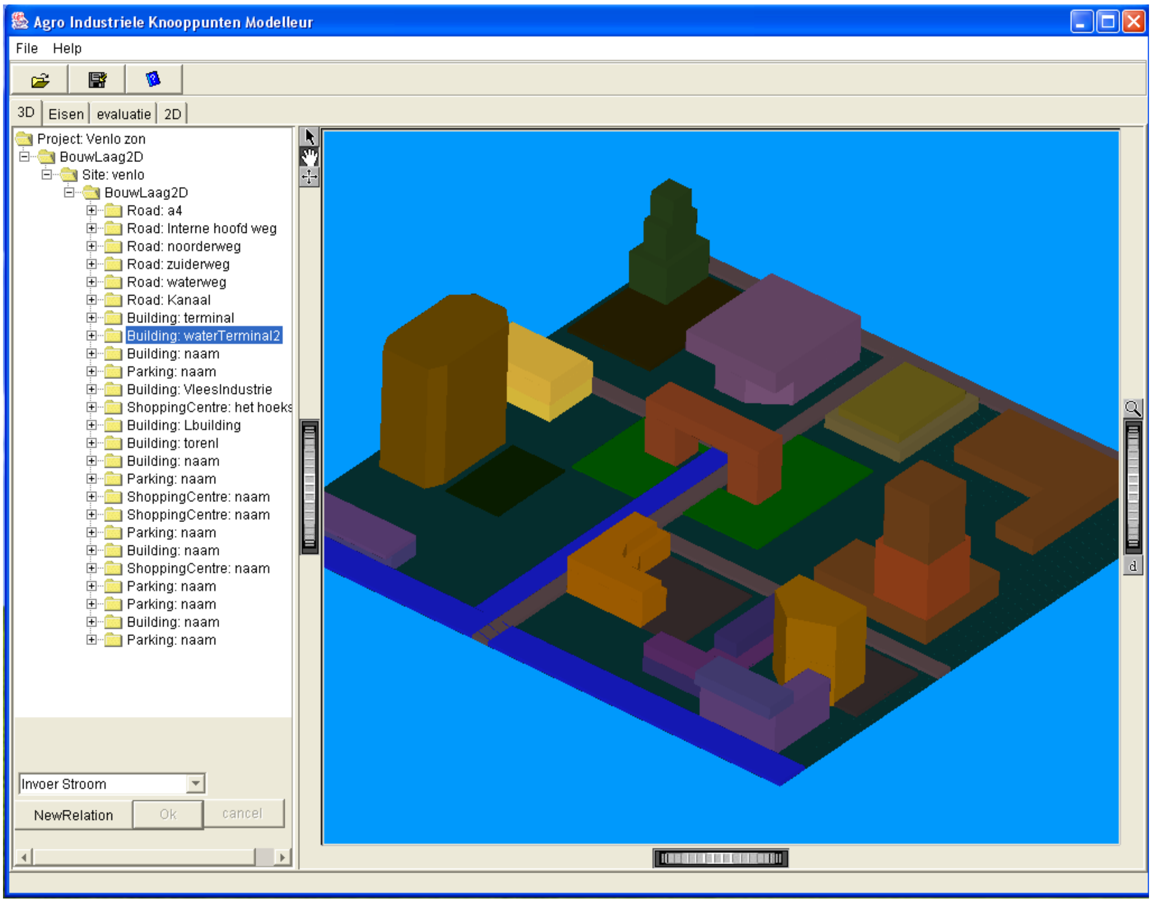


Figure 8-24 3D Representation of an Agro Business Park.

Locations outside of the business park such as harbours and retailers can be modelled using a box on the extremes of the model. This allows connections of the cluster with the outside world.

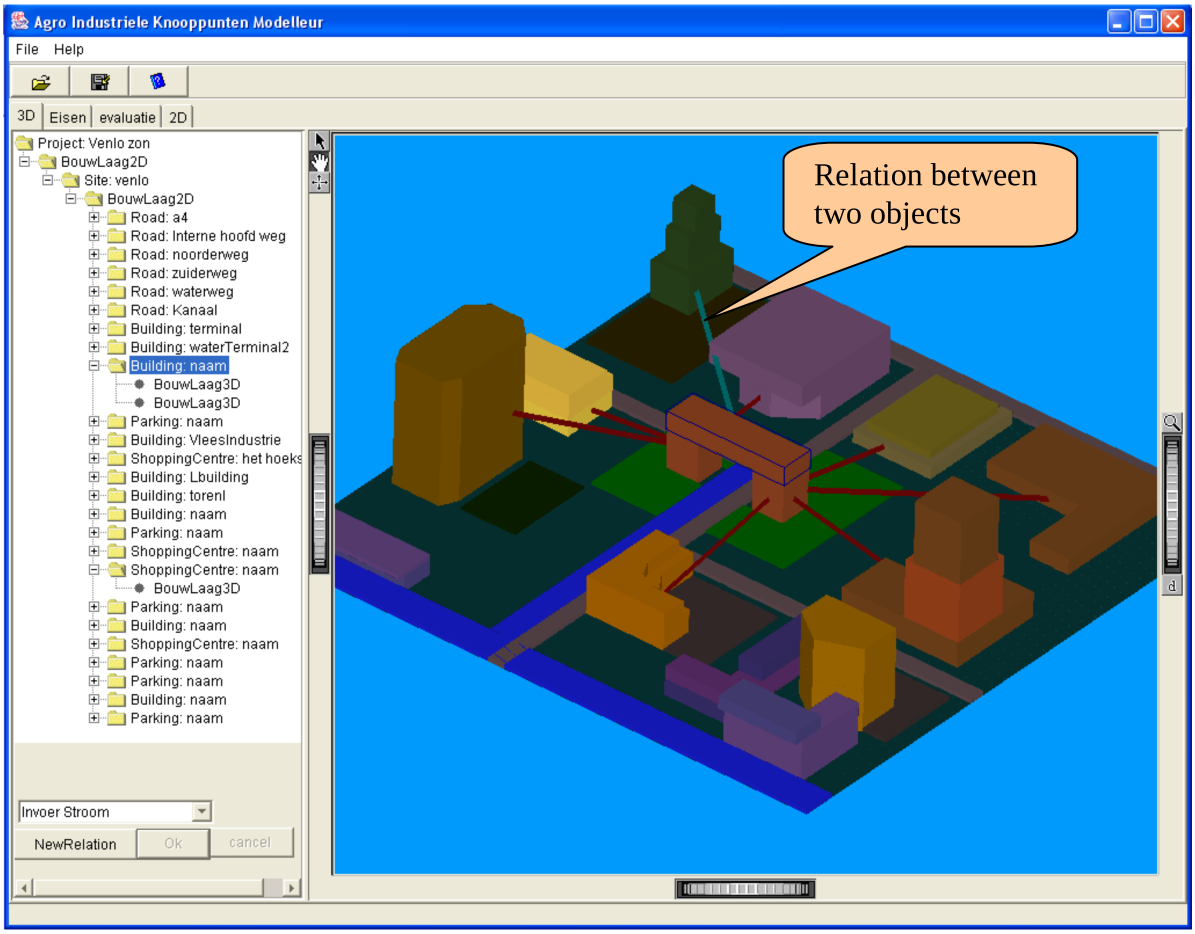


Figure 8-25 The relations between the terminal and other objects in the park

To help the client with decision making, the DSS provides a summary of the characteristics of the solutions in the form of charts presenting available spaces, required spaces, composition of the clusters, et cetera (Figure 8-26).

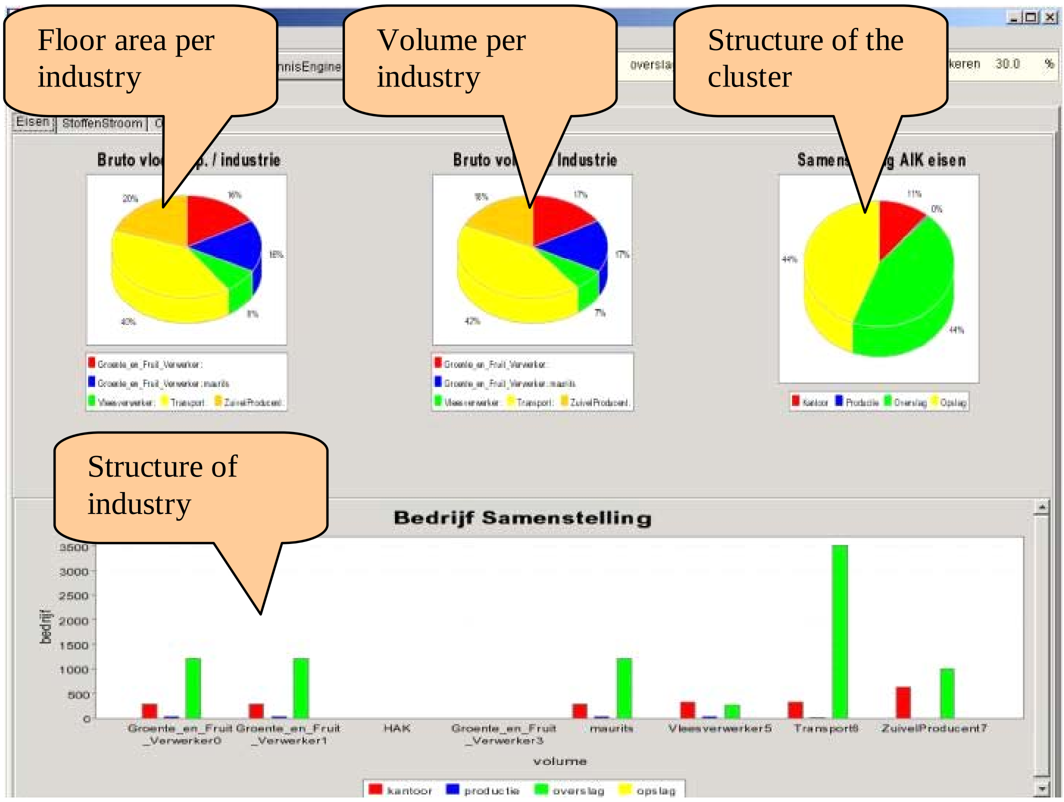


Figure 8-26 A screenshot of charts presenting characteristics of the design solution.

8.4.4 Conclusions

This DSS is able to cope with existing situations by modelling the buildings and their functions. By assigning new functions to new or existing buildings, new layouts can be generated. The DSS keeps track of the use of spaces when functions or buildings change.

The DSS is also able to model connections between (agricultural) companies belonging to the same cluster or from outside. Presenting overviews of these connections supports the discussion for possible cooperation of clusters or companies regarding transport movements, storage capacities, et cetera.

The shape model introduced in this DSS is different than the one used in the previous DSS's. Therefore the implementation of the 3D user interface had to be modified. This simple shape model makes it easier to implement shape related rules.

8.5 CONCLUSIONS

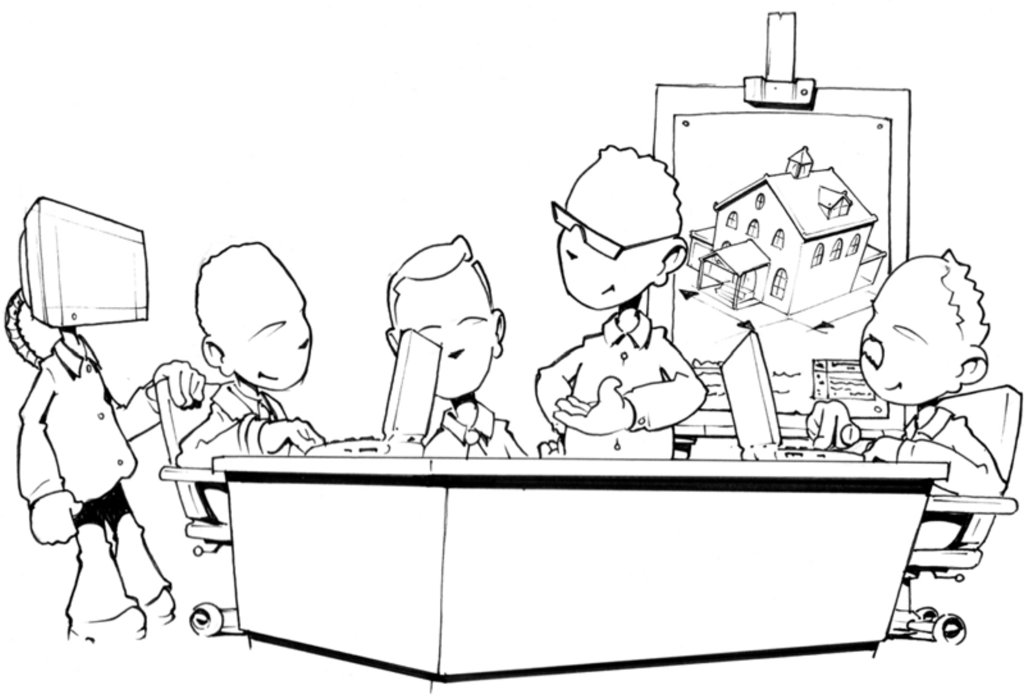
Three project-type-specific Demand Support Systems are developed by creating project-type-specific ontologies and Knowledge Components associated with these ontologies. Each Demand Support System is based on the same conceptual architecture.

The three prototypes demonstrate various features of the proposed DSS such as reasoning with ill-defined information, the use of legacy applications, the use of multiple Virtual Experts, coping with variable input, supporting different levels of detail, et cetera. Furthermore, the prototypes demonstrate the usability of the conceptual architecture and the usefulness of the DSS for the (inexperienced) client in the BC industry during the inception phase.

It seems that the architecture is reasonably suitable for creating Demand Support Systems. However the case studies show only the use of a limited set of Virtual Experts. The second case study uses three Virtual Experts. In this case study a lot of Behaviour Objects are generated which increased the amount of Model Objects rapidly. The rapid growth of the Project Model is not always desirable. In addition, the search for Patterns in larger models proved to be difficult because of the large memory requirements. This search routine seems not very suitable for large models but may be improved by more ‘processor intensive’ routines (versus this ‘memory intensive’ routines).

9 Conclusions

This chapter presents the main conclusions and formulates a number of recommendations.



Supporting the client during the inception phase using a Demand Support System with Virtual Experts is feasible.

9.1 INTRODUCTION

This thesis study started with the presumption that ICT can improve the client's role in a Building Construction process. Clients of the Building and Construction industry are often confronted with products and processes that do not live up to their expectations. Due to the one-of-a-kind nature of the Building and Construction industry the client is usually not able to test the product in advance – the resulting facility *is* the prototype – and therefore the client's expectations may be incorrect due to perception problems. Improving the client's perception requires that the client needs to increase his insight in the consequences of his desires and in the technical possibilities that are available in the market. The client could of course rely on human experts to serve his needs, but if he wants to stay as independent as possible (which often seems to be a good idea) the client, without having to become an expert himself, needs access to the Body of Construction Knowledge (BoCK).

This thesis is an inquiry into the application of Information and Communication Technology (ICT) focusing on providing the client with a suitable access to the BoCK during in the inception phase of a BC project. The analysis of the existing application of ICT in the BC industry shows that clients have no suitable ICT-enabled access to the BoCK during the inception phase. The analysis of state-of-the-art Knowledge Technology shows that a lot of technology is available but hardly any is used in the inception phase.

This thesis proposes a Demand Support System (DSS) comprising of Virtual Experts that give the client a suitable access to the BoCK during the inception phase. Virtual Experts have their own specific knowledge domain and collaboratively support the client. The generic and evolutionary approach of the DSS supports the insertion of new Virtual Experts capable of reasoning on new knowledge domains. This approach enables the creation of relatively small and project-type-specific Demand Support Systems, which can evolve into more sophisticated systems without having to deal with the increasing complexity of the system. The case studies have shown several aspects of a Demand Support System and have shown the potential of these systems in supporting the client in the inception phase. The findings are as follows:

9.2 CLIENT FOCUS

- ICT is able to provide the client with a suitable access to the Body of Construction Knowledge during the inception phase of Building and Construction project.

- The independency between Knowledge Components enables the insertion of new Knowledge Components without having to deal with existing ones. This approach reduces the complexity for extending the system and supports fragmented knowledge elicitation. However this approach requires management of these components (such as the management of conflict situations, trust, certainty, etc.).

- An important process in the inception phase is the search for relationships between the unknown client values and project/product characteristics. This research proposes that Virtual Experts proactively make assumptions for these relations, which can be evaluated by the client.

Besides these conclusions there are also a number of conclusions on the validity of the ICT approach.

9.3 ICT FOCUS

- A Demand Support System as proposed in this thesis is technically viable as demonstrated by the implementation of the system.

- The posterior integration of Knowledge-Based Systems including legacy systems into a Demand Support System is possible as demonstrated by the second case study. Particularly the integration (re-use) of legacy applications is attractive for the BC industry because of its abundance of legacy applications.

- Knowledge-Based Systems that need information, which is not available in the inception phase, can still be used by automatically generating the necessary information.

- Generating the details needed to create input for the application of legacy systems is useful as (1) knowledge abstraction also often means that useful knowledge is lost, and (2) interactions between different knowledge domains often can not be solved on a high abstraction level.

- The separation of the ontology from the Knowledge Components increases the scalability of the Demand Support System by allowing new Knowledge Components to join the system (via Virtual Experts).

- Live sharing of the Project Model is a pre-requisite for a direct proactive support of multiple distributed Virtual Experts.

9.4 RECOMMENDATIONS

The following recommendations can be made:

- Responsibility, (un) certainty and trust will obviously be very important when the client makes certain decisions based on the output of a Demand Support System. Before deploying Demand Support Systems in practice, it is recommended to look deeper into these aspects.

- The usage and impact of these kinds of systems are not dealt with in this thesis. It is recommended to have a look into the impact of such a system on the building process.

- In order to make DSS commercially feasible, it is recommended to start with simple Demand Support Systems focusing on a large market for a specific type of project with a limited set of design solutions such residential renovation or extension projects.

- An evolutionary development of Demand Support Systems is possible by re-using other knowledge intensive applications. Therefore it is recommended to other knowledge intensive application developers make their application web-enabled in order to enable its re-use.

- It is recommended to investigate how to get a desired behaviour of the Project Model as a whole regarding the proactive insertion of Behaviour Objects by autonomous Virtual Experts.

- Various options for Guided Extension could be envisioned. For example generating an IFC compatible database will make it easier to integrate existing IFC compatible applications. Another line of research is to develop and apply typologies (libraries of different solution types that might serve the purpose).

Appendix A. List of Abbreviations

3D	3-Dimensional
AI	Artificial Intelligence
AP	Application Protocol
API	Application Programming Interface
BC	Building and Construction
CAD	Computer Aided Design or Computer Aided Drafting
CIB	Commission International du Batiment.
CTI	Civil Informatics
DAML	DARPA Mark-up Language
DARPA	Defence Advanced Research Projects Agency
DXF	Drawing eXchange Format
GUI	Graphical User Interface
HTML	HyperText Mark-up Language
HTTP	HyperText Transfer Protocol
HVAC	Heating, Ventilation and Air Conditioning
ICT	Information and Communication Technology
ICKT	Information, Communication and Knowledge Technology
IAI	International Alliance for Interoperability
IFC	Industry Foundation Classes
ISO	International Standards Organisation
JDK	Java Development Kit
JSDT	Java Shared Data Toolkit
JVM	Java Virtual Machine
KBS	Knowledge-Based System
KIF	the Knowledge Interchange Format
KR	Knowledge representation
NGI	Next Generation Internet
OIL	Ontology Inference Language
OO	Object Oriented
OoI	Objects of Interest
OOp	Object Oriented programming
OWL	Web Ontology Language
PDM	Product Data Management
PDT	Product Data Technology
PSM	Problem solving method
R&D	Research & Development

RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
STEP	STandard for Exchange of Product model data
UML	Universal Modelling Language
URI	Universal Resource Identifier
URL	Universal Resource Locater
UML	Unified Modelling Language
VRML	Virtual Reality Mark-up Language
WWW	World Wide Web
XML	eXtensible Mark-up Language

Appendix B. Unified Modelling Language Reference

UML stands for Unified Modelling Language. This OO system of notation has evolved from the work of Grady Booch, James Rumbaugh, Ivar Jacobson, and the Rational Rose Corporation. Today, UML is accepted by the Object Management Group (OMG) as the standard for modelling OO systems.

UML defines ten types of diagrams:

- *Class Diagrams*. Class diagrams are the backbone of almost every object-oriented method, including UML. They describe the static structure of a system.

- *Package Diagrams*. Package diagrams are a subset of class diagram, but developers threat them sometimes as a separate technique. Package diagrams organise elements of a system into related groups to minimise dependencies between packages.

- *Object Diagrams*. Object diagrams describe the static structure of a system at a particular time. They can be used to test class diagrams for accuracy.

- *Use Case Diagram*. Use diagrams model the functionality of system using actors and use cases.

- *Sequence Diagrams*. Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

- *Collaboration Diagrams*. Collaboration diagrams represent interactions between objects as a series of sequenced messages. Collaboration diagrams describe both the static structure and the dynamic behaviour of a system.

- *Statechart Diagrams*. Statechart diagrams describe the dynamic behaviour of a system in response to external stimuli. Statechart diagrams are especially useful in modelling reactive objects whose states are triggered by special events.

- *Activity Diagrams*. Activity diagrams illustrate the dynamic nature of a system by modelling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of a system. Typically, activity diagrams are used to model workflow or business processes and internal operation.

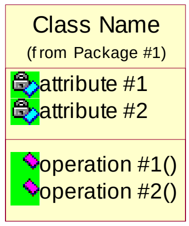
- *Component Diagrams*. Component diagrams describe the organisation of physical software components, including source code, run-time (binary) code and executables.

- *Deployment Diagrams*. Deployment diagrams depict the physical resource resources in a system, including nodes, components and connections.

This part gives a brief overview of the Class Diagram symbols and notations, as used in this thesis.

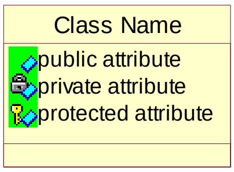
Classes

Classes are represented as rectangles divided into compartments. The name of the class is placed in the first partition and can include an additional package name, the attributes are listed in the second partition, and the write operations (or methods) are listed in the third partition.



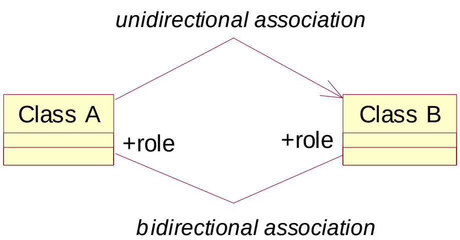
Visibility

Visibility markers are used to signify who can access the information contained within a class. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class.



Associations

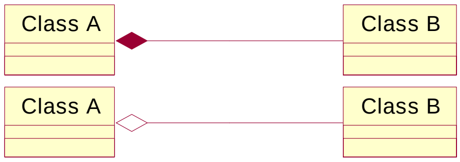
Associations represent static relationships between classes. Association names are placed above, on or below the association line. Association can be bi-directional or unidirectional. Role names represent the way two classes see each other.



Multiplicity

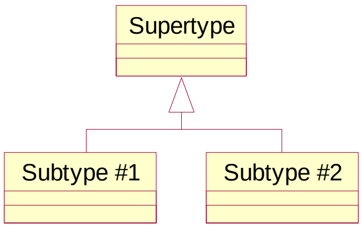
Multiplicity notations are placed near the ends of an association. These symbols indicate the number of instances, which can be associated with one instance of the other class.

Composition and Aggregation



Composition (filled diamond) is a special type of aggregation that denoted strong ownership between class A, the whole, and class B, its part. A hollow diamond represents a simple aggregation relationship in which the 'whole' plays a more important role than the 'part' class.

Generalisation



Generalisation is another name for inheritance or 'as-is' relationship. It refers to a relationship between two classes where one class is a specialised version of another. This relationship is also often referred to as supertype/ subtype relationship.

Vraag Ondersteuning door Virtuele Experts

Ondersteunen van de opdrachtgever in de Bouw gedurende de initiatieffase

1 Introductie

Deze dissertatie beschrijft een onderzoek naar de toepassing van ICT om de opdrachtgever in de Bouw te ondersteunen. Hoofdstuk 1 presenteert een eerste inzicht in de rol van de opdrachtgever. Opmerkelijk is dat de rol van de opdrachtgever tamelijk zwak is omdat hij geen controle heeft over het bouwproces dat onvoorspelbaar kan zijn in termen van tijdsduur, budget en kwaliteit.

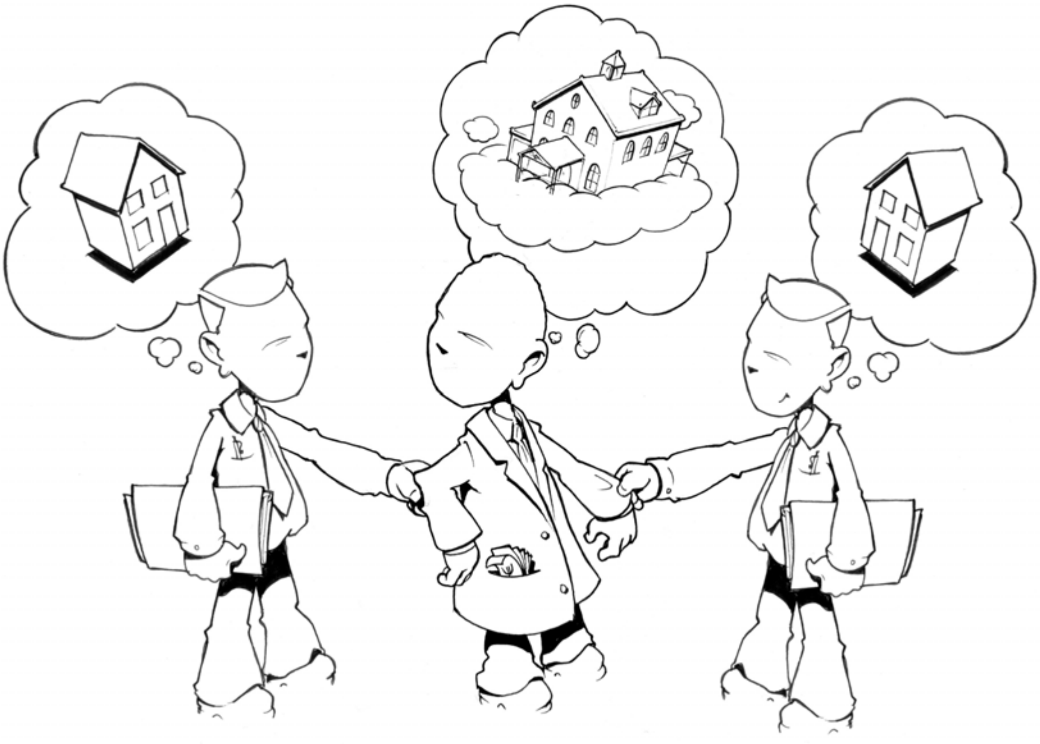
Figuur 1 De opdrachtgever heeft niet voldoende controle over het bouwproces.

Hoofdstuk 1 beschrijft tevens de potentie van ICT voor de bouw als ‘enabler’ en beargumenteert dat ICT de prestatie van de bouw kan verbeteren. Dit in beschouwing genomen, wordt de volgende initiele onderzoeksvraag gesteld:

- Hoe kan ICT de rol van de opdrachtgever in een bouwproces verbeteren?

2 Analyse van de rol van de opdrachtgever in het bouwproces

De opdrachtgever moet een vraag specificeren (bijvoorbeeld een programma van eisen of vragen naar diensten van experts) gedurende het bouwproces. Deze vraag is in dit onderzoek gedefinieerd als het verschil tussen de huidige situatie en de gewenste situatie inclusief het transformatie proces. Wanneer de opdrachtgever geen goede perceptie heeft van de huidige en gewilde situaties loopt hij het risico op ontevredenheid met het eind resultaat.



Figuur 2 Zonder een goed beeld van huidige en gewilde situaties loopt de opdrachtgever het risico ontevreden te worden met het eindresultaat.

Gedurende het project kan de opdrachtgever zijn beeld bijstellen door de aanwezigheid van meer relevante informatie. Deze verandering in het beeld creëert een nieuwe vraag resulterende in extra kosten waarbij delen van de vorige vraag overbodig worden. Vooral gedurende de initiatieffase loopt de opdrachtgever een hoog risico om een incorrecte vraag te stellen omdat relevante informatie niet of onvoldoende aanwezig is. Hoewel veel kennis aanwezig is in de bouw, zijn er enkele redenen aanwezig waarom de opdrachtgever daarvan vrijwel geen voordeel heeft en dus de broodnodige informatie in de initiatieffase moet ontberen. Door de opdrachtgever toegang te geven tot de bestaande bouwkennis tijdens de initiatieffase, zal hij in staat gesteld worden een vraag te ontwikkelen dat de verwachte meerwaarde brengt. Dit onderzoek richt zich op de mogelijk of ICT de opdrachtgever toegang kan geven tot deze bouwkennis zonder de aanwezigheid van experts en zonder dat de opdrachtgever zelf een expert hoeft te zijn.

3 Analyse van de huidige toegang tot de bouwkennis door middel van ICT

Hoofdstuk 3 analyseert de toegang tot de bouwkennis door middel van ICT en concludeert dat de huidige software applicaties niet de juiste toegang kunnen verschaffen voor de opdrachtgever gedurende de initiatieffase omdat:

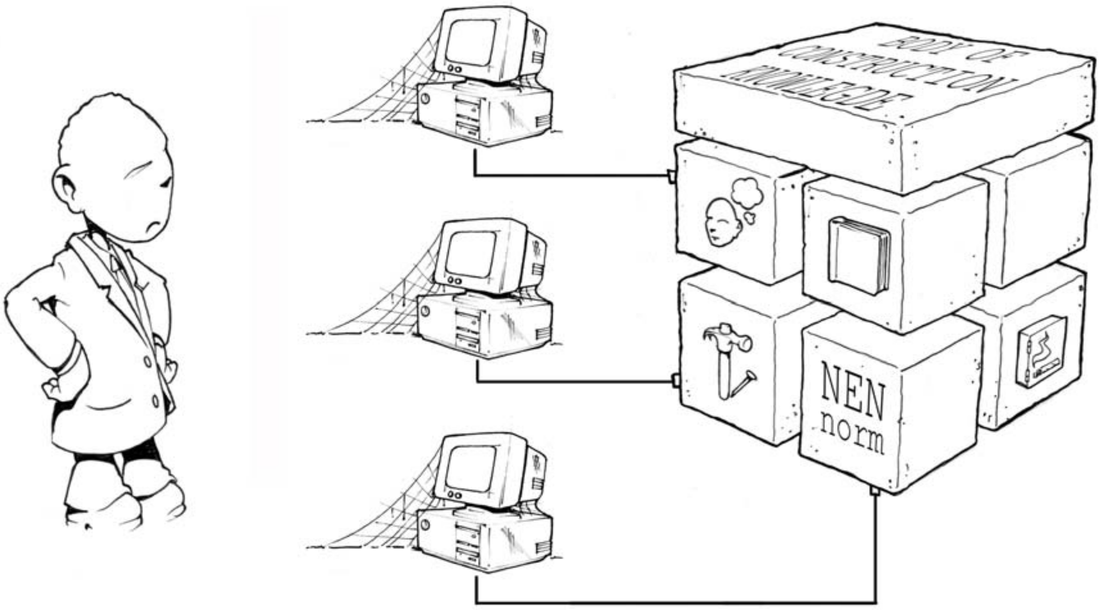
- alleen een klein en specifiek gedeelte van de bouwkennis toegankelijk is met bestaande applicaties. Uitbreiden of integreren van deze bestaande applicaties is lastig omdat de meeste applicaties gesloten systemen zijn.

- de toegang tot de bouwkennis d.m.v. ICT gefragmenteerd is terwijl de opdrachtgever een integrale toegang nodig heeft.

- tot nu toe er geen geschikt mechanisme aanwezig is dat kan omgaan met de aanwezig invoer informatie. De huidige applicaties accepteren alleen invoer informatie op een gefixeerd detail nivo. De aanwezige informatie gedurende de initiatieffase kan gekarakteriseerd worden als schaars en slecht gedefinieerd, zoals incompleet, niet samenhangend, conflicterend, vaag, onzeker en op verschillende detail nivo’s. De meeste applicaties zijn niet inzetbaar gedurende de initiatieffase omdat ze een specificieke set van (gedetailleerde) informatie nodig hebben.

- de meeste applicaties een specifiek werkterrein hebben en dus minder bruikbaar zijn voor de opdrachtgever.

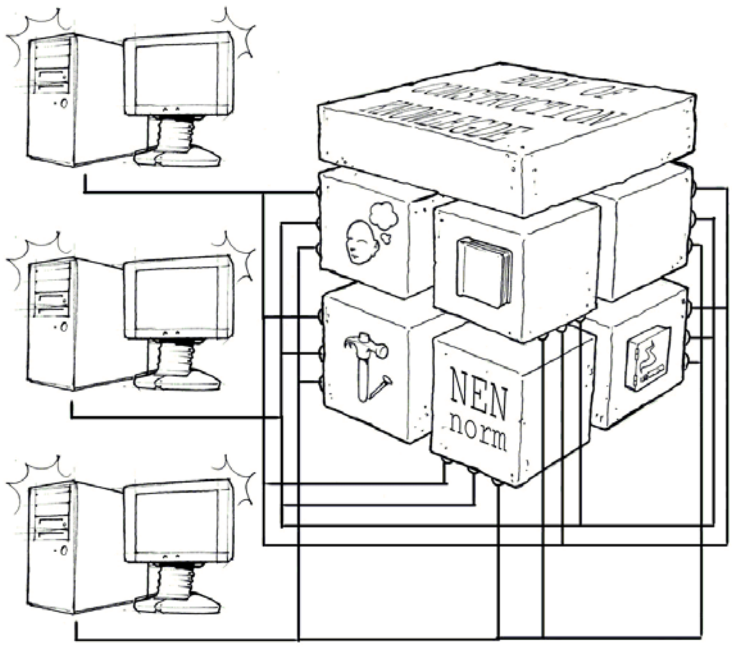
- de kennis opgesloten is in de applicaties en niet zichtbaar of open voor de opdrachtgever of voor de expert.



Figuur 3 De opdrachtgever heeft geen goede toegang tot de gefragmenteerde bouwkennis.

4 Analyse van of state-of-the-art Kennis Technologie

Kennis gebaseerde systemen als onderdeel van kennis technologie biedt nieuwe mogelijkheden om te kunnen redeneren over de bouwkennis zoals het ontsluiten van nieuwe delen van de bouwkennis. Veel verschillende soorten kennis gebaseerde systemen zijn aanwezig met elk z'n voordelen. Zo zijn er kennis gebaseerde systemen aanwezig die kunnen redeneren op basis van schaarse en slecht gedefineerde informatie. Technologie gerelateerd aan bijvoorbeeld het Semantisch Web, agents en Web Services ondersteunt een gedistribueerde benadering voor het ontsluiten van kennis gebaseerde systemen. Echter, een groot aantal (overlappende) benaderingen en implementaties zijn aanwezig met elk z'n voor- en na-delen.



Figuur 4 State-of-the-art kennis technologie biedt veel nieuwe mogelijkheden om te kunnen redeneren over de bouwkennis.

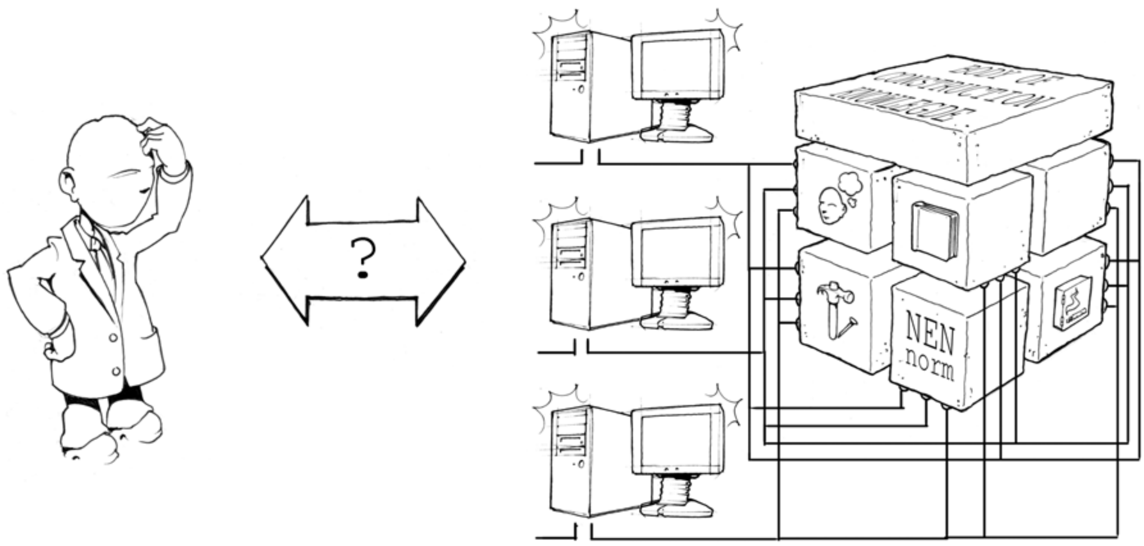
5 Detailleren van de onderzoeksvraag

Om de opdrachtgever te ondersteunen in het bouwproces, heeft hij zo vroeg mogelijk toegang nodig tot de bouwkennis. Daarvoor is een applicatie nodig die de kant toegang verschaft tot de bouwkennis gedurende de initiatieffase om hem zo te ondersteunen met de ontwikkeling van de vraag; een Vraag Ondersteunings Systeem (VOS).

De volgende vijf gedetailleerde onderzoeksvragen zijn boven komen drijven in hoofdstuk 5:

1. Wat zijn de eisen voor een een Vraag Ondersteunings Systeem?
2. Wat is een geschikte conceptuele architectuur voor een Vraag Ondersteunings Systeem?
3. Hoe kan een Vraag Ondersteunings Systeem gebruik maken van legacy software?

4. Hoe kan een Vraag Ondersteunings Systeem onervaren opdrachtgevers ondersteunen?
5. Hoe kan een Vraag Ondersteunings Systeem gemaakt worden dat kan omgaan met de opkomende nieuwe generatie van kennis intensieve computer applicaties?



Figuur 5 Detaillering van de initiele onderzoeks vraag.

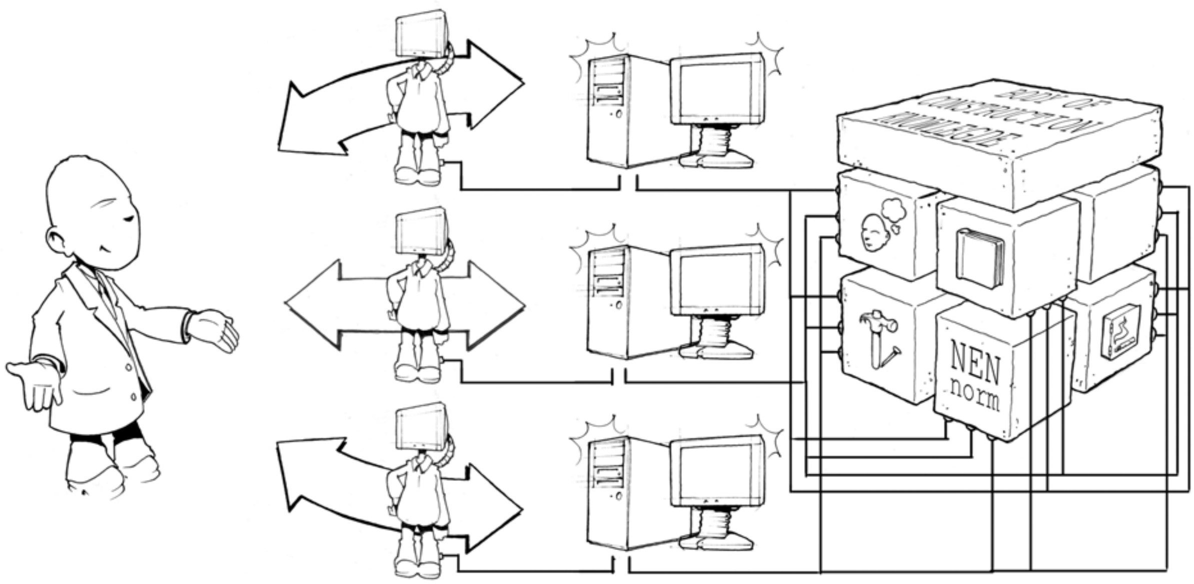
6 Functioneel ontwerp van een Vraag Ondersteunings Systeem gebruikmakende van Virtuele Experts

Hoofdstuk 6 beantwoordt de vijf gedetailleerde onderzoeksvragen uit hoofdstuk 5 en maakt een voorstel voor een conceptuele architectuur voor een Vraag Ondersteunings Systeem (VOS). In deze architectuur gebruiken Virtuele Experts proactief een netwerk van gedistribueerde en onafhankelijke kennis gebaseerde systemen om te kunnen redeneren over het gedeelde Project Model. Hiervoor gebruiken de Virtuele Experts, Kennis Componenten die kennis gebaseerd systemen ontsluiten door deze te relateren aan Patronen in het Project Model.

De voorgestelde conceptuele architectuur maakt integratie met legacy applicaties mogelijk en biedt mogelijkheden om technische informatie te interpreteren naar verschillende abstractie nivo’s opdat het begrijpbaar gemaakt kan worden voor de opdrachtgever.

Het gebruik van een geformaliseerde ontologie ondersteunt de conceptualisatie van Objects of Interest over verschillende detail nivo’s en faciliteert flexibiliteit met betrekking tot de uitbreiding van de VOS. Hierdoor kan een VOS gemaakt worden voor een specific project type met de potentie om door te evolueren.

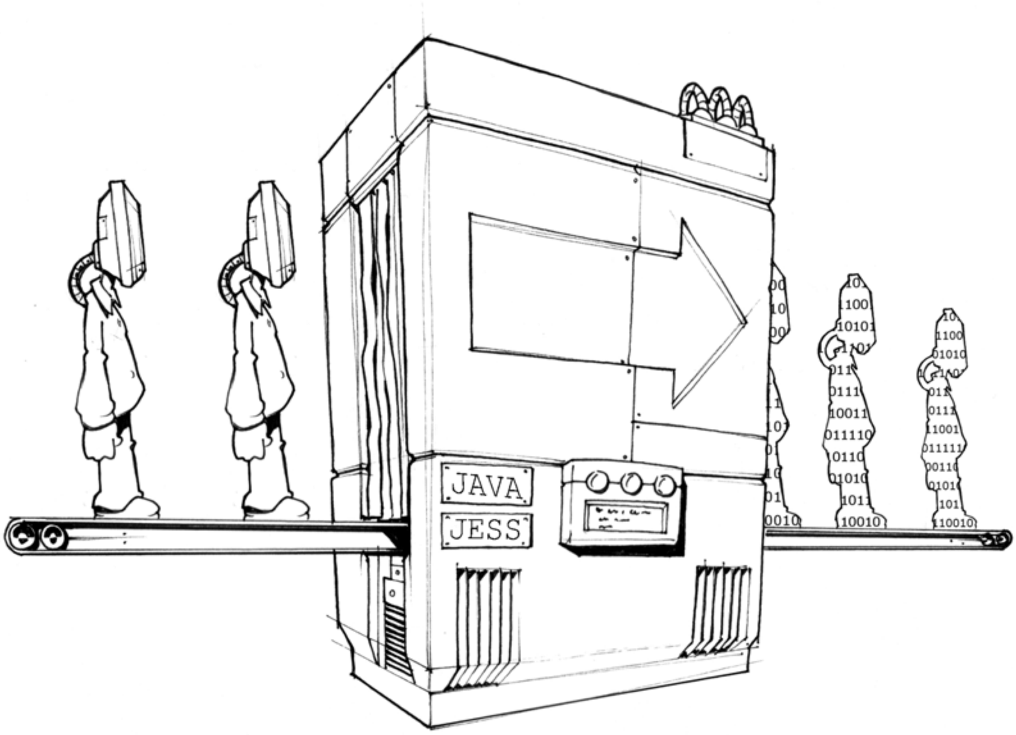
Complementariteit met technologie gerelateerd aan het Semantische Web zorgt ervoor dat de VOS klaar is om het hoofd te bieden aan de opkomende generatie kennis intensieve computer applicaties.



Figuur 6 Virtuele Experts geven de opdrachtgever toegang tot de bouwkennis.

7 Implementatie van een Prototype Vraag Ondersteunings Systeem

Hoofstuk 7 bediscussieert de implementatie van de voorgestelde conceptuele architectuur. Het Project Model, Behaviour Objecten en de ontologie zijn geimplementeerd en kunnen gedeeld worden over het Internet. De ontologie is open en de inhoud kan worden veranderd waardoor de ontologie de ontwikkeling van een VOS ondersteunt voor een specifiek project type. Tevens zijn andere delen van de conceptuele architectuur geimplementeerd zoals Virtuele Experts, Knowledge Components en Patroon herkennings routines.

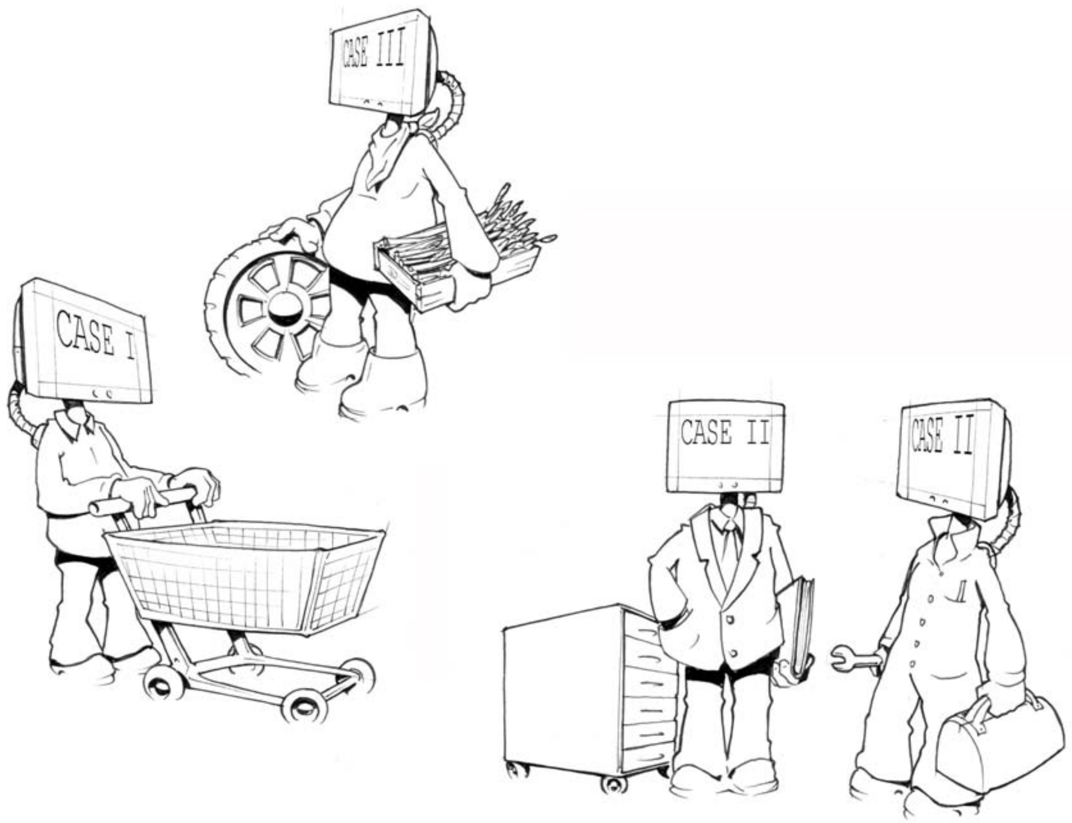


Figuur 7 Implementatie van Virtuele Experts voor een Vraag Ondersteunings Systeem.

User interfaces die vorm gerelateerde objecten in 2D en 3D kunnen weergeven zijn gemaakt om de ontwikkeling van een VOS te ondersteunen. Om de ontwikkeling van een VOS te ondersteunen voor een specifiek project type is er een ontologie editor en een Kennis Componenten editor gemaakt. Met behulp van deze editors is het mogelijk een dergelijke VOS te maken zonder daarbij de conceptuele architectuur of user-interfaces te veranderen.

8 Case Studies

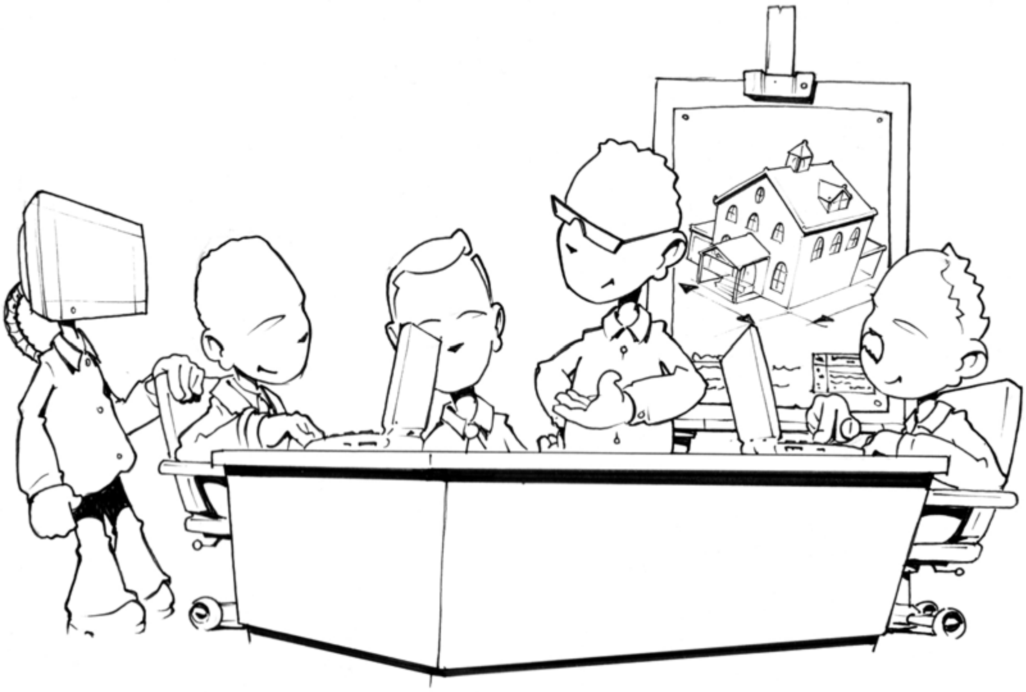
Hoofdstuk 8 demonstreert het potentieel van de conceptuele architectuur en van Vraag Ondersteunings Systemen in drie case studies. Door de ontwikkeling van project-type-specifieke ontologieën en verschillende Virtuele Experts zijn drie Vraag Ondersteunings Systemen ontwikkeld op basis van dezelfde conceptuele architectuur. Deze Vraag Ondersteunings systemen demonstreren verscheidene kenmerken zoals het redeneren op basis van slecht gedefinieerde informatie, het gebruik van legacy applicaties, het gebruik van meerdere Virtuele Experts, het omgaan met variaties in de invoer gegevens, de ondersteuning in verschillende detail nivo's, etcetera. Tenslotte demonstreren the prototypes de bruikbaarheid van de conceptuele architectuur en het nut van een VOS voor de (onervaren) opdrachtgever in de bouw gedurende de initiatieffase.



Figuur 8 Virtuele Experts voor de case studies.

9 Conclusies

Het Vraag Ondersteunings Systeem met Virtuele Experts zoals voorgesteld in dit onderzoek is een poging tot het ondersteunen van de opdrachtgever gedurende de initiatieffase. De huidige staat van Vraag Ondersteunings Systemen is niet productierijp, maar dit onderzoek laat zien dat het concept van Vraag Ondersteunings Systemen kansrijk is om de opdrachtgever te ondersteunen waarmee hij zijn role in de initiatieffase van het bouwproces kan verbeteren.



Figuur 9 Een Vraag Ondersteunings Systeem is kansrijk om de opdrachtgever te ondersteunen om zijn rol in het bouwproces te verbeteren.

Curriculum Vitae

Last name	: Schevers
First name	: Hans Albert Joseph
Date and place of Birth	: 11 March 1975, s-Hertogenbosch, the Netherlands
Nationality	: Dutch
Address	: Lariestraat 25. 5482 LH, Schijndel
Phone	: +31 (0) 73-54 74753
E-Mail	: HansSchevers@hotmail.com

Education:

- MSc studies at the Faculty of Civil Engineering at Delft University of Technology.

- Gymnasium Beekvliet, St. Michielsgestel, the Netherlands. High school: preparatory university education.

Work track:

- **PhD student** at the Faculty of Civil Engineering at Delft University of Technology.

- **Part time Consultant** at TNO Building and Construction Research, Department of Building Process Innovations, Delft, The Netherlands.

References

- Achten, H. (1997), Generic Representations, An Approach for Modelling Procedural and Declarative Knowledge of Building Types in Architectural Design, Ph.D. Thesis, Bouwstenen-publication No. 46 in the series of the scientific press of the Department of Architecture, Eindhoven, Netherlands

- Ang, G., Prins, M., De Ridder, H.A.J., (2003), The added value of architecture in performance based control of Design & Build projects, proceedings of the Joint International Symposium of CIB working commissions, Singapore.

- Augenbroe, G.L.M., (2001), “Building simulation trends going into the new millenium”, proc. 7th International IBPSA conference, Rio de Janeiro, Brazil.

- A.W.T (2000), Bouwen op kennis, rapportage verkenningscommissie Bouw, isbn 90 346 3812x, Den haag, (in Dutch).

- Berners-Lee T., Hendler, J. and Lassila O., (2001) “The Semantic Web,” Scientific Am., vol. 284, no. 5, USA

- Brachman R, Levesque H, eds., (1985), Readings in Knowledge Representation, Los Altos: Morgan Kaufman.

- Brand, Van Den GJW, Nijland, JRP (1999), De vraag centraal ! Innovatief aanbesteden van (rijks) kantoorgebouwen, (in dutch)

- Boelhouwer, B., Schevers, H.A.J. (2003), “Toetsing van bouwontwerpen”, Notitie TNO.

- Chen-Burger (2001), J, Knowledge Sharing and Inconsistency Checking on Multiple Enterprise Models, Informatics Research Report EDI-INF-RR-0037, AIAI, University of Edinburgh.

- Daconta, M.C., Obrst, L.J., Smith, K.T., (2003) The Semantic Web, a guide to the future of XML, Web Services, and Knowledge Management, Wiley Publishing Inc., Indianapolis, Indiana.

- Dado,E,,(2002) ICT-Enabled Communication and Co-operation in Large-Scale On-Site Construction Projects, PhD. Thesis, Delft University of Technology.

- Damski, J. C. and Gero, J. S. (1997), An evolutionary approach to generating constraint-based space layout topologies, in R. Junge (eds), CAAD Future 1997, Kluwer, Dordrecht, p. 855-874.

- Dell'isola, A. J. (1982), Value engineering in the construction Industry, third edition, Van Nostrand Reinhold Company Inc, USA.

- De Ridder, H.A.J., Van Der Klauw, R.A., Vrijhoef, R. (2002) "Het nieuwe bouwen", TNO report 2002-BPI-28, available at: http://www.tno.nl/perskamer/archief/2002/documenten/2002_bpi_028.pdf.

- De Ridder, H.A.J. (1998), Design & Construct of complex civil engineering systems, a new approach to organization and contracts. Delft: Delft University Press.

- De Ridder, H.A.J. (2001), "Dynamic performance control concept of design & construct of complex systems", proceedings CIB World Building congress Performance in Product and Practice, Wellington, New Zealand.

- De Ridder, H.A.J. (2002) "Performance Based Control of Design and Construct Contracts", Proceedings CIB Joint Conference Hong Kong.

- De Ridder, H.A.J., Vrijhoef, R. (2003), Developing a framework for life cycle value management for built facilities, proceedings of the Joint International Symposium of CIB working commissions, Singapore.

- Dullaart W., A. Wichers Hoeth (1998), Opdrachtgevers in beeld, SBR 307, isbn 90-5367-101-3, (in Dutch)

- Egan, J., et al., (1998), Rethinking Construction, Available at: http://www.rethinkingconstruction.org

- Fensel, D., (2002) The semantic web: yet another hip?, Vrije Universiteit Amsterdam, Netherlands.

- Friedl, G.; Trum, H.M.G.J.; Rutten, P.G.S. (2001): Modelling the Design Process: A Process-choreography, Proceedings `International Workshop on Architectural Management`, CIB W096; The University of Reading, Reading, United Kingdom.

- Froese,T., (2003) " Future Directions for IFC-Based Interoperability", PUBLISHED: July 2003 at http://www.itcon.org/2003/17

- Kikuo Fujita, Shinsuke Akagi (1999), Agent-Based Distributed Design System Architecture for Basic Ship Design, Department of Computer-Controlled Mechanical Systems, Osaka University.

- Gielingh, W.F. (1987), General AEC Reference Model for AEC product definition data (version 3), TNO Building informatics, Delft.

- Guida,G., (1994) "Design and Development of Knowledge-Based Systems", Wiley & Sons LtD, Chichester, England.

- Korbijn, A. (1999), Vernieuwing in productontwikkeling, strategie voor de toekomst, Stichting Toekomstbeeld der Techniek, The Hague, (in Dutch).

- Lawson, BR (1982), How designers Think, Architectural Press, London, UK

- Leusen Van M. (1994), A system of types in the domain of residential buildings, TU Delft, PhD thesis Delft University of Technology.

- Melody, W.H. (2002), The Triumph and Tragedy of Human Capital: Foundation Resource for the Global Knowledge Economy, TU-Delft. Available at http://www.lirne.net/about/papers/TriumphTragedyHC.pdf.

- NDL (2000), "European Logistics, positioning European Logistics concepts in the changing businessenvironment of transparency supply and demand networks", vereniging Nederland Distributieland (NDL), the Netherlands

- Nederveen, van G.A, (2000) Object Trees: Improving Electronic Communication between Participants of Different Disciplines in large-scale Construction Projects, Ph. D. thesis, Delft University of Technology, Netherlands.

- Newell, A, (1982) The Knowledge Level, International journal of Artificial Intelligence no18.

- Rezgui, Y., Body, S, Bouchlaghem, D, Hassanen, M., Cooper, G,Barret, P., Austin, S., (2001), A proposed IT-Based Approach for Managing the Construction Brief Effectively, Proceedings of the CIB-W78, IT in construction, South Africa

- Rijn Van T., Jagbeck A, Karstila K., “Concurrent Engineering in the Tendering Process of Building & Construction”, Proceedings of the CIB Working Commission W 78, Information Technology in Construction Conference, Stockholm, Sweden, 3-5 June, 1998

- Ozsariyildiz, S.S. PhD thesis, Delft University of Technology, Forthcoming

- Porter, M.(1985) Competitive Advantage, creating and sustaining superior performance, , Macmillan inc, New York, usa.

- Reniers, M.G.C.E., De Ridder, H.A.J. (2003) “Proposed method of analysing dissatisfaction based on arbitrary cases”, CIB student chapters Innovation in Construction and Real Estate, Hong Kong.

- Rutten, P.G.S., Trum, H.M.G.J.,(2000), ‘On the innovation of buildings’, Technical University of Eindhoven, The Netherlands.

- Michael Sannella (1992), The SkyBlue Constraint Solver, Tech Report 92-07-02, University of Washington, Computer Science and Engineering department.

- Schevers, H.A.J. (1999), Hospital Inception Support System, master thesis, Delft University of Technology.

- Schevers, H.A.J., and Tolman, F.P. (2000). “Supporting the Inception Stage of Building Projects with Instant Value versus Cost Evaluations” Proceedings of the CIB78 Conference, Reykjavic Iceland.

- Scott, W., Cook. S.C. (2003) An architecture for an intelligent requirements elicitation and assessment assistant, proceedings of 13th annual international INCOSE symposium,

- Sowa, John F., (2000), Knowledge representations, Pacific Grove Brooks/Cole, USA.

- Speel, P.H. & Aben, M. (1997); Applying a library of problem-solving methods on a real-life task, International Journal Human-Computer Studies 46

- Spekkink, D., (2003), Visie ICT en Bouw, SBR publication, The Hague.

- Spekkink, D., (2002), Next Generation Scenario ICT in de Bouw, available at: http://www.minez.nl/publicaties/pdfs/02I05.pdf (in Dutch).

- Spekkink, D., Van Wijk, M. (1998), Bouwstenen voor het PvE, Stichting Bouw Research, Rotterdam, (in Dutch).

- Studer, V.R. Benjamins, and D. Fensel, (1998), Knowledge Engineering: Principles and Methods. Data & Knowledge Engineering 25.

- Tolman F.P., M.R. Beheshti, (2003), College Dictaat 4270, Tu Delft (in Dutch)

- Tolman. F.P., Ozsariyildiz S. and Schevers, H. (1999). “Information Modelling to Support Inception and Early Conceptual Design of Large-Scale Engineering Projects”, International Journal of ICT, Volume 7, No. 2

- Tolman, F.P., (1999), Product modelling standards for the building and construction industry: past, present and future, Journal Automation in Construction, the Netherlands.

- Tolman F.P., (1998), Bouwinformatica: ontwerp en constructies, Delft University of Technology, Netherlands, (in dutch)

- Tolman F.P., (2002), Informatie- en Kennistechnologie in het Bouwproces. Delft University of Technology, Netherlands, (in dutch)

- Vrijhoef, R., De Ridder, H.A.J.(2003), “Developing a Value-Price-Cost leverage model for integrated value chain and life cycle management of built facilities”, CIB student chapters Innovation in Construction and Real Estate, Hong Kong.

- Wel, van der G.J., (2002), Design Assistant, master thesis, Delft University of Technology.

- Weiss, G. (editor), 2000, Multi agent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press, USA.

- Willems, P. (1998) Conceptual modelling of structure and shape of complex civil engineering projects, PhD thesis, Leidschendam.

- Woestenenk, K., (2000), Implementing the Lexicon for Practical Use, Proceedings of CIT 2000, Iceland.

- Zeisel,J., (1981), Inquiry by Design, Cole publishing company, California, USA.

Internet Resources

[ArchiCAD] http://www.graphisoft.com/products/archicad/

[ADT] http://www.autodesk.com/products/ADT

[DOORS] http://www.telelogic.com/

[Catalyze] http://www.steeltrace.com/

[CICA] http://www.cica.org.uk/

[Cobrite, n.d.] IT Tools and Support for Improved Construction Briefing, available at: http://www-staff.lboro.ac.uk/~cvmahh/cobrite-intro.htm (2003).

[CommonKADS] http://www.commonkads.uva.nl

[CONCUR] Concurrent Design and Engineering in Building and Civil Engineering , available at http://cic.vtt.fi/projects/concur (2003).

[DOE-2] http://www.doe2.com

[eConstruc, 2002] eCommerce and eBusiness in the European Building and Construction Industry: Exploiting the Next Generation Internet, website, www.econstruct.org, 2002

[EDT] Estimation Desktop, http://www.estimatingdesktop.com.

[Gruber, 1993] Tom R. Gruber, website, http://www.ksl.stanford.edu/kst/what-is-an-ontology.html, 1993

[ICCI] Innovation co-ordination, transfer and deployment through networked Co-operation in the Construction Industry, available at http://icci.vtt.fi/ (2003).

[IES4d] http://www.ies4d.com/

[Java] www.java.sun.com

[Java2D] http://java.sun.com/products/java-media/2D/

[Java3D] http://java.sun.com/products/java-media/3D/

[JESS] http://herzberg.ca.sandia.gov/jess/

[JSDT] http://java.sun.com/products/java-media/jsdt/

[Jun4Java] http://www.sra.co.jp/people/nisinaka/Jun4Java/index_en.html

[Kubus Stabu] http://www.kubusinfo.nl

[KIF] knowledge Interchange Format, website, http://logic.stanford.edu/kif/kif.html

[MS ACCESS] http://www.opengl.org/

[Motta, 2000] http://kmi.open.ac.uk/projects/ibrow/Publications/Motta_pkaw00.pdf

[OntoKnowledge] http://www.ontoknowledge.org

[OpenGL] http://www.opengl.org/

[Principia] Principia Cybernetica Web,avaialble at http://pespmc1.vub.ac.be/ (2003)

[PSIB, n.d.] Proces en systeeminnovatie in de bouwsector, Available at: http://www.psib.nl/

[RDF, 1999] Resource Description Framework (RDF) model and syntax specification, W3C Recommendation, 1999, http://www.w3.org/tr/rec-rdf-syntax/

[Roadcon, n.d.] Strategic Roadmap towards Knowledge-Driven Sustainable Construction, available at: http://www.roadcon.org/ (2003).

[Servlets] http://java.sun.com/products/servlet/

[TRNSYS] http://www.trnsys.com

[UDDI] http://www.uddi.org

[VABI] http://www.vabi.nl