10 MODELLING THE FIRST BUILDING LIFE CYCLE STAGES

Ir. H. Schevers and Prof. Ir. F.P. Tolman
Delft University of Technology, Faculty of Civil Engineering and Geosciences Stevinweg 1, P.O. Box 5028, 2600 GA Delft, the Netherlands

h.a.j.schevers@ct.tudelft.nl and f.tolmaln@ct.tudelft.nl

Abstract

This paper discusses the preliminary results of a research project that uses Product Data Technology (PDT) and Knowledge Technology (KT) for supporting the first building life cycle stages. Within this project a prototype implementation has been made providing a Virtual Reality environment where technical buildings in the early project phases (inception and early design) can be simulated. The system supports the creation and evaluation of concept solutions. These evaluations consist of feasibility studies on aspects like energy consumption, durability and costs. The evaluations are immediately available and provide more insight on the concept design. The basis for the implementation is a building model integrated with formalised knowledge. The paper discusses how early project data can be captured with PDT and how knowledge can be used to support the early building life cycle phases.

Keywords: *product data technology, knowledge technology, Inception phase*

INTRODUCTION

This paper discusses the first results of a research project focusing on support for the inception and early design stages of building projects. The research concentrates on the iterative design process where requirements and concept designs are developed. Within this process, design decisions determine the course and outcome of the project. The impact of the design decisions on the course of the project becomes smaller when the project evolves (see figure 1). Supporting the decision process using ICT in early project phases has been recognized earlier and different projects have been started on this subject [CoBrITe] [Seed] [Ozsariyildiz & Tolman 1998].

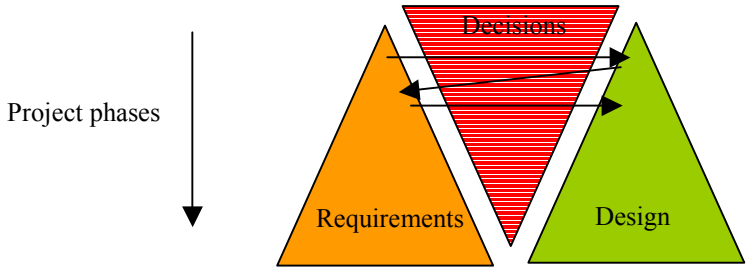

Figure 1- the iterative design process of the development of requirements, synthesis and analysis of design alternatives.

CoBrITe tries to improve the information in the briefing process through more efficient and effective use of existing and emerging IT that can support both the client as well as designers. The SEED project has similar goals and is developing a Software Environment to support the early phases in building Design. My colleague in Delft, Saban Ozsariyildiz, is also working on a thesis in this area. In his approach Saban implements a Functional Unit – Technical Solution decomposition [Gielingh 1988] of a project and adds knowledge rules to guide the instantiation and decision making process, mainly focusing on cost aspects.

Supporting the early design phases in such a way that the quality of the decisions improves, is beneficial for the whole project. To improve the quality of the decisions support for developing the requirements and concept designs should be complemented combination with support for evaluating the concept designs. Incorrect project decisions are the result of uncertainty and inaccuracy of information. Also other reasons for making incorrect decisions are present:

- The project is done in an environment where many restrictions apply such as crowded environments (like the Netherlands). Many governmental restrictions are present to protect the existing situation. Even different authorities guard their own interest. Probably neither all restrictions nor empowered authorities are present at the early stages. If early design decisions will lead to a situation where restrictions are applicable is hard to predict.
- When the early phases are part of a tendering process, the available time is limited. Aspect evaluations in order to make the correct decision are not always ready or available in time. Especially when changes are made until the last minute.
- When the project is complex, evaluations can be too simple and thus incorrect. Even lack of information or lack of relevant criteria can result in incorrect early design decisions. With complex projects it is not apparent in the early phases which properties are relevant.

- Re-use of knowledge gathered in previous projects is not stored in a neutral format and is not easy accessible.

This paper describes the preliminary results of a research project into ICT support for the early design phases. Two prototype implementations are made. One implementation was a concept implementation to investigate applicability of a Virtual Reality based Graphical User Interface (GUI) [Schevers 1999] [Schevers & Tolman 2000]. The second prototype is a concept implementation based on techniques explained in the next section of this paper.

SOLUTION CONCEPT

Basically the idea is to apply the principles of Product Data Technology (PDT) integrated with knowledge technology (KT). With Virtual Reality (VR) based GUI's a virtual project environment can be created. Within this environment it is possible to explore the boundaries of the project by developing requirements and concept solutions that fit within this environment. The values of the concept solutions can be tested by multi aspect evaluations. Having support in developing the requirements and the concept design in combination with direct feedback on multi aspects, actors within the project can orientate and make the correct decisions for the real project. To have more insight on this solution concept, a prototype implementation has been made just to create a better view on such an environment. This prototype does not deal with the real problems of creating such an environment but gives a good idea of how the environment must work [Schevers & Tolman 2000]. In this prototype it is possible to model existing hospital situations and create a concept solution by demolishing existing buildings and create new buildings. Having some knowledge about hospitals implemented, the size of the new buildings can be calculated in order to meet the requirements. Furthermore knowledge about costs could be made available which resulted in a direct cost feedback.

REQUIREMENTS FOR PRODUCT DATA TECHNOLOGY IN THE EARLY PHASES

PDT is a technology to describe products in a computer interpretable language. The idea is to describe only the relevant aspects and properties of the product. In the scope of this research, the product model has to describe a (technical) building with all relevant aspects for the early project phases (the inception phase and the early design phase). This means the building model must be able to model the relevant information in these phases by capturing and storing all relevant information. 'The building design process produces design information that naturally grows in quantity as the design evolves' [Rivard & Fenves 2000]. This information has a range of uncertainty and vagueness.

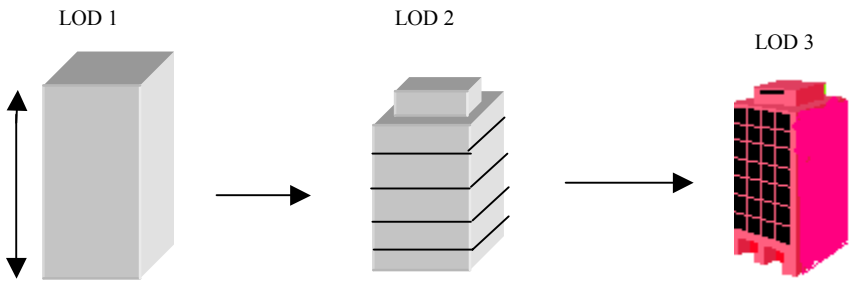

Figure 2 shows the design evolution for a building during the early design process. The initial design ideas are vague but during the project more and more information about the design becomes available.

Figure 2 illustrates the evolution of a building (design concept) from a rough concept idea to a more detailed design. *The product model must support this design evolution*. Furthermore the model must support multi aspects evaluations of a building project like structural engineering aspects, thermo related aspects, cost aspects etc. All disciplines have their own view on the building. *This means the product model must integrate multiple views*. Furthermore all relevant project information must be captured. For example (new) client's requirements and the design concepts must be captured not only in terms of initiating new instances or setting certain values, but also in terms of new concepts and data structures. *Therefore the product model must be flexible regarding extensions and semantics*. For supporting the early life cycle stages of a building, support is needed for the requirements and concept designs (see Solution Concept). Knowledge Technology and the product model must interact. *This means the product model must be integrated with KT*. International efforts like AP225 from Standard for the Exchange of Product model data [STEP] and the Industry Foundation Classes [IFC] International Alliance for Interoperability [IAI] are focussing on later stages of the design and on data sharing and exchange. For the data-exchange, protocols like SDAI and file exchange formats like STEP-physical file format have been developed. Unfortunately these efforts do not explicitly support the design evolution and are based on static building representations. This means that the representation is not really suitable for knowledge integration.

A SIMPLE PRODUCT MODEL FOR THE EARLY PHASES

For supporting the design evolution several mechanisms are available. One important example is the Intension-Extension mechanism. Basically the Intension-Extension orders information with increasing meaning, i.e. from abstract meaning to detailed meaning. An example of the Intension-Extension dimension in Building Construction is the following ordering:

- Participant
- Architect
- John Doe, Architect / Studio 2000, Architect / Leslie Hunt, Architect

Participant is the most abstract way to refer to the people that together realise a project. Participants play different roles: Architects, Engineers, Subcontractors, etc. The notion of types of types (as in Participant) is sometimes called a power type. Architect, like Engineer, Contractor and such are instances of Participant (it are participants). And John Doe, Studio 2000 and Leslie Hunt are instances of Architect (it are architects).

This idea can be generalised and used to describe a building project in the Inception stage. Notions that play a role are for instance: Circulation Space, Structural System, Space Separation Structure, Foundation, and such. Each of these notions has an Extension in the Design model that has instances in the Project DB. Circulation Spaces are Staircases and Lift shafts for vertical transport and Corridors, Halls, and such for horizontal transport. Another mechanism for supporting the design evolution is the dynamic properties mechanism. With this mechanism it is possible to further specify a certain object during its lifecycle in the project. Using the dynamic properties it is possible to add new properties or to remove existing properties at 'run-time'. The last mechanism for supporting the design evolution is the Levels of Detail (LOD) mechanism. In figure 2, a building in three different LOD is displayed. Figure 3 illustrates the building evolution in product data.

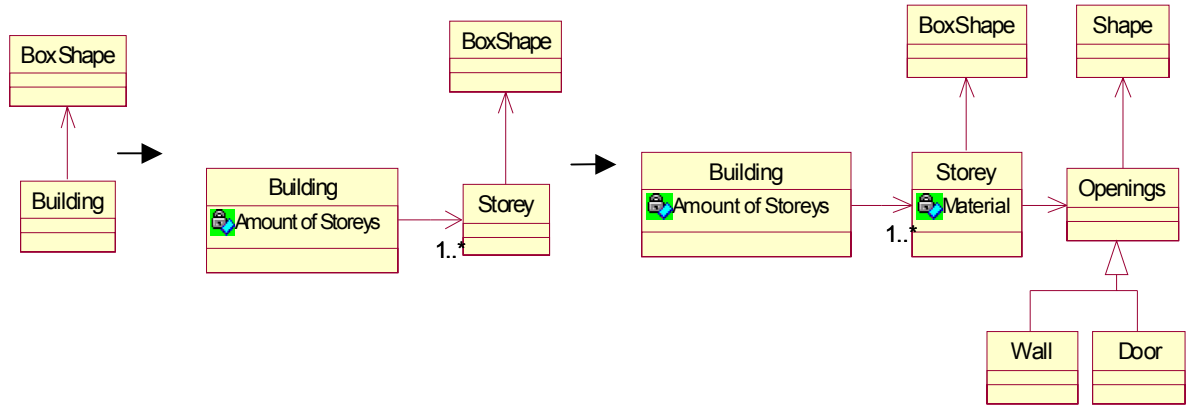

Figure 3 shows the evolution of the building design in terms of product data. First the building is modelled with simple data structures. During the design process the building model becomes more complex.

How many LODs are relevant is specific for each project. Predefining a product data structure for certain levels of detail for buildings is not suitable. This makes the LODs a dynamic part in the product model. The dynamic LODs provide a flexible approach of modelling the development of project information.

Using a generic product model where semantics are included instead of using a predefined schema, improves the flexibility of the model. Within this approach a taxonomy database containing a dictionary with semantic concepts and relations to properties and other semantic concepts is necessary. Within this database the relations can specify Intension-Extension dimension, default properties, decomposition structures, etc. The data in the database can be used as basis for applying knowledge technology.

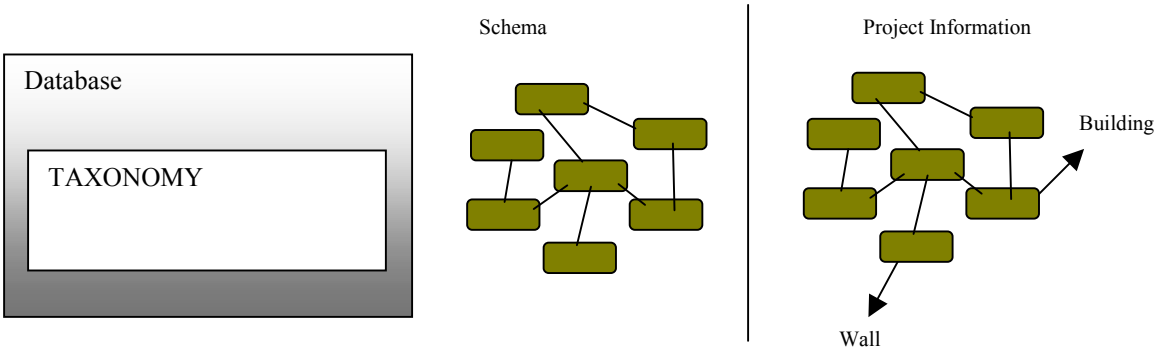

Figure 4 shows an abstract layout of this modelling approach. The schema with concepts out of the taxonomy database is used for creating new objects that contain the specific project information. Relations have been predefined in order to assure this project model does not become an unorganised pile of semantics. Connecting objects with the predefined relations organises the objects. The semantic of the relations can be predefined or also retrieved from the taxonomy database. With these relations, it is possible to bring structure to the objects. The meaning of this structuring is totally based on the own perception of the modeller and has no influence on the behaviour of the model. This means that relations are just connections between objects in order to structure the model.

KEEPING THE MODEL CONSISTENT.

The main problem of the approach described is how to keep such a model consistent? The key for keeping the model consistent is the relation object. The meaning of the relations has an impact on the state of the objects and thus affects the model in its behaviour. Therefore the meaning of the relations must be identified and implemented. For identifying the correct behaviour of a relation, the concepts of the objects have to be taken into account. This means the behaviour of a certain relation is dependent of the meaning of the objects. For example, when a Building in LOD 1 has decomposition relations with storey spaces in LOD 2, increasing the storey height of the storey spaces will affect the height of the building. This means, the model must increase the building height in order to keep the whole model consistent. In this particular case the building height becomes a derived property. This behaviour of the model can be *identified* by a building product data structure. The building data structure contains a Building Object with a LOD relation with storey spaces. Having identified this situation, Behaviours have to be implemented in order to assure the consistency of the model. For the actual implementation of Behaviours, many different approaches are possible. Simple Procedural routines can be used but also KT approaches as Neural Nets and Genetic Algorithms. These Behaviours operate within the product model.

In the prototype implementation several simple procedural behaviours are implemented, for example, a ‘derived property’ behaviour. This behaviour responds when a property of the same type in one of the decomposed objects is changed (see figure 5).

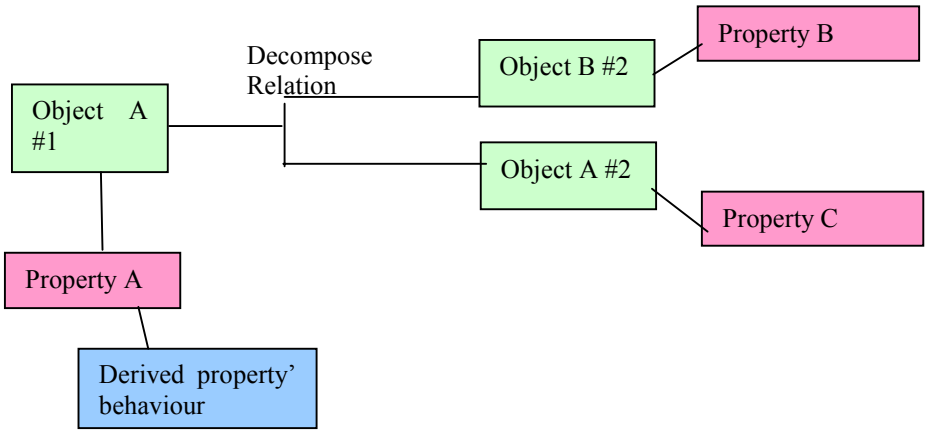

Figure 5- object model with behaviour attached to a property object-

The 'derived property' behaviour calculates new values of the property by adding all property values of the composition objects. Such behaviour is of course not attached to every property within the model because this behaviour is not always desirable. The behaviour is dependent on the semantics of the property and the object, which is connected with the property. Therefore the semantics must be identified and checked to see if such behaviour is desirable. Not only the semantics determine the desired behaviour, also the state of the property values determines the desirability of a behaviour. These property states can be for example indifferent, desirable, or demanded, etc. If certain behaviour is desirable the behaviour can be attached to the correct object. For identifying which behaviour is wanted in which situation, a pattern recognition mechanism within the product model is needed. With this approach, it is possible to create a library of behaviours that are automatically inserted when a certain product data pattern occurs. Having all behaviours identified and implemented in combination with all relevant patterns, the model can be kept consistent.

KNOWLEDGE COMPONENTS

The prototype implementation called Concept Modeller [CM] has the goal to support the early project phases. As described earlier in this paper, requirements support, design support and evaluation support is necessary. Knowledge Technology is used to create this support. Because the product model approach is flexible, the knowledge system must be able to cope with this flexibility. This means the knowledge must be able to identify certain product data structures before it can be used. The product model pattern recognition is used to identify when certain knowledge can be used. Basically the idea is to capture the knowledge gathered and used in earlier projects and store this knowledge in a knowledge database. Using the key product data structure as pattern when the knowledge was used, the knowledge can be re-used when the same pattern occurs in other projects (see figure 6). Connecting the knowledge components to the product model allows good trace ability when knowledge used. Also with this approach knowledge components can be created with patterns containing other knowledge components. This means knowledge components can overrule, adjust, etc existing knowledge components. With this approach, it is possible to store and re-use knowledge in components in a flexible manner.

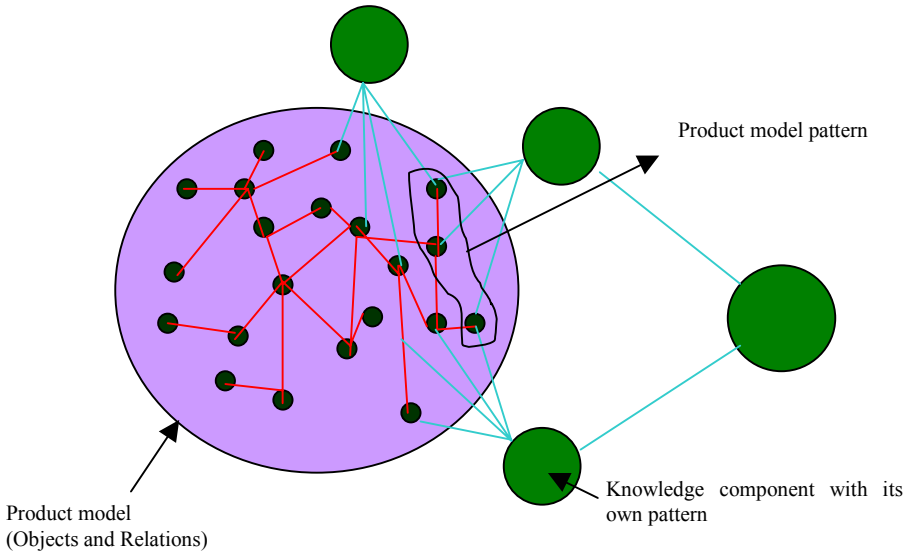

Figure 6 –pattern recognition for executing the knowledge components-

How the knowledge is implemented is not described. Basically every representation can be used from simple rules to neural nets or existing programs. Simple examples of this approach are for example property search mechanisms based on concept objects. This mechanism searches default properties for instantiated objects. A more complex example has been implemented which contains a set of procedural knowledge components.

PROTOTYPE IMPLEMENTATION

An implementation has been made based on the described approach. A simplified taxonomy database has been created which contains concepts and default properties. A simple product data schema has been created which can be instantiated using semantics out of the taxonomy database. With this implementation it is possible to add and remove new properties at 'run-time'. The high level layout of the implementation is illustrated in figure 7.

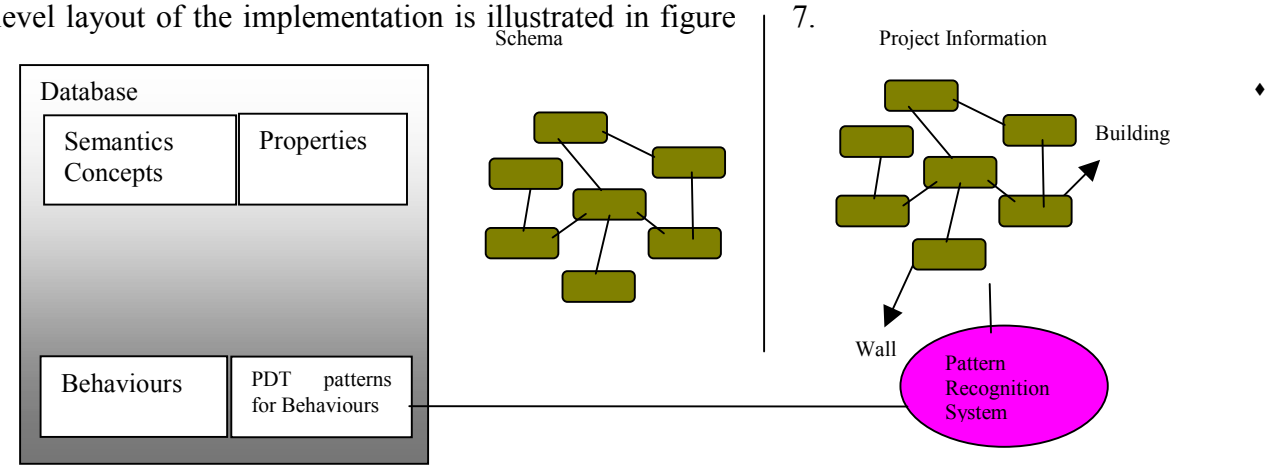

Figure 7- Implemented approach for prototype of the Concept Modeller-

As described in this paper the schema contains objects, predefined relations and a skeleton for behaviour objects. Having implemented behaviours stored in a database with PDT patterns, a Product Data pattern recognition system has been used to insert behaviours automatically. For the pattern recognition the Java Expert System Shell [http://herzberg.ca.sandia.gov/jess] has been used. Jess is a rule base system based on a Rete algorithm to process rules and to solve many-to-many matching problem. The top level of the schema in figure 7 is displayed in figure 8.

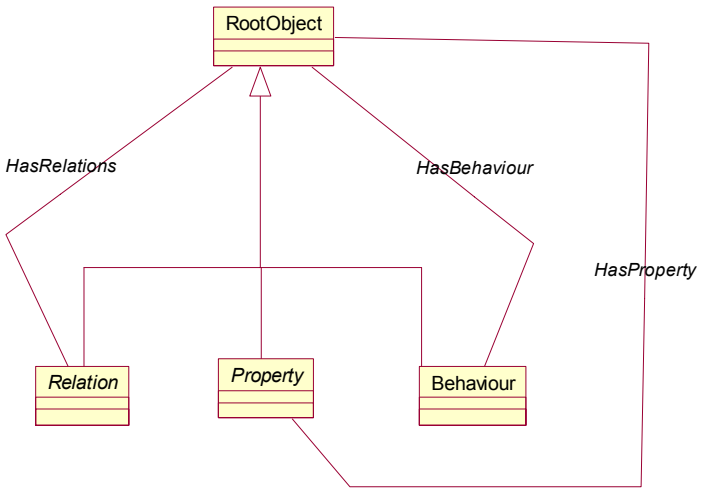

Figure 8- the top-level part of the schema-

In this figure a RootObject can be instantiated with a semantic concept from the taxonomy database. Furthermore this object can have relations with the Objects: Relation, Property and Behaviour. These last three objects are subtypes of the RootObject. This creates a lot of flexibility: With this model relations can be connected to properties and vice versa. Even Behaviour objects can have behaviours, etc. With this implementation, further efforts have been made aiming at office buildings.

In large projects dealing with office buildings, the clients requirements starts with identification of an organisation structure followed by the spatial requirements. The prototype provides some basic functionality to capture these organisation structures and the functional masses. Choosing from a library of requirements (library window in figure 9) the user is able to create a specific set of requirements (project requirements window in figure 9). Within the library, most defined objects have default properties. In figure 9 a property frame displays the default properties of the *OfficeOrganisation* Object.

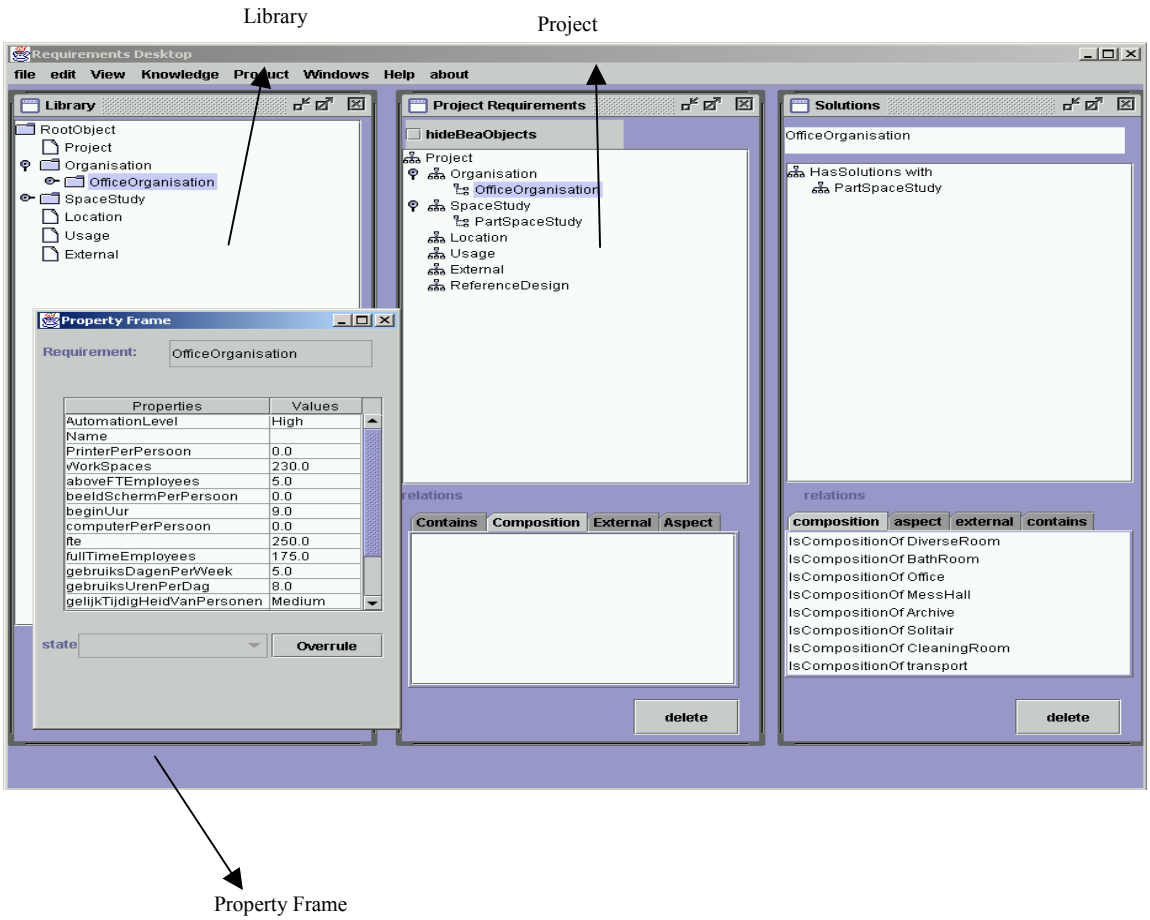

Figure 9 –screenshot of the requirements desktop-

The Requirements are ordered using a template or a view of the project data. At the moment the requirements are viewed following a standard clients requirements list. This means the project tree contains first a decomposition tree of the organisation followed by a decomposition tree of the functional spaces and at the end a decomposition tree of the design itself. In the current implementation all other requirement types can be attached to the functional spaces or the design itself. Knowledge components are implemented for supporting the requirements. Default office organisations are present and from these organisations, functional space requirements can be generated.

For displaying the concept design itself, a new desktop has been implemented (figure 10). Within the 3D environment of this desktop, objects like building, wall, installation, parking area, etc can be instantiated with a certain shape. The shape is visualised and the properties of the objects can be displayed in a similar frame as in figure 9. Connecting the Space study objects in figure 9 to a building object in figure 10 by creating a relation between these objects, the system automatically adjusts the size of the building object to match the Space study.

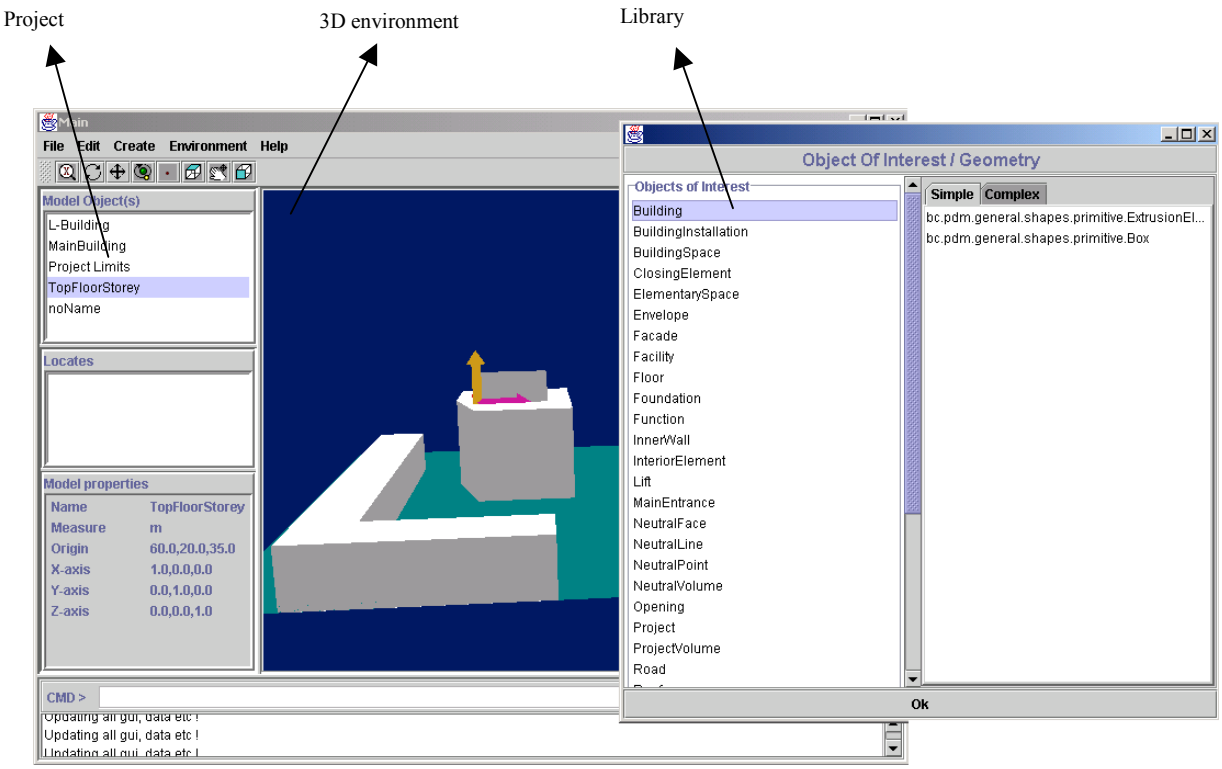

Figure 10- screenshot of the concept design desktop-

Now it is possible to select a building in the concept design desktop and calculate the HVAC aspects. As result of this HVAC calculation, the system will display figure 11. Changing the properties, the shape of the buildings or changing the walls or roofs, will have of course a different HVAC result. The results will also suggest a certain installation for the building. These installations can be added to the buildings.

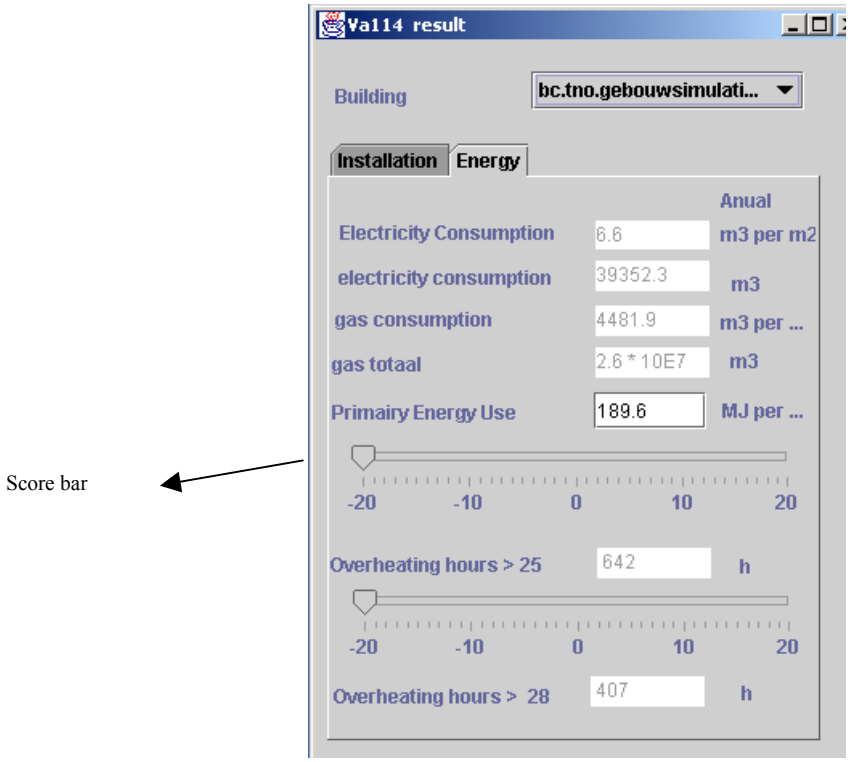

Figure 11 -HVAC result frame-

CONCLUSIONS

This paper discussed a PDT and KT approach for supporting the early life cycle phases of a building project. Using VR as Graphical User Interface, the idea is to concurrently support the development of the requirements, concept design and the evaluation of the concept design, focusing both on costs and values (values are performances expressed in money). An implementation of such a system has been made filled with data and knowledge of office building projects. As an example, the prototype concept modeller supports on-the-fly evaluation of HVAC performance characteristics during the development of the building model. Other evaluations will be added in a later stage. The prototype contains some simple features of an open product data model approach and a first integration with knowledge has been made. The current implementation contains only a simple knowledge base and the product model supports only a limited set of the needed features. In order to support the early life cycle bigger knowledge bases are necessary. Problems, which occur when scaling this system, are not yet dealt with. For example: structuring the knowledge components, conflict situations, etc. Also the current implementation of the taxonomy database is oversimplified and needs improvement.

More information on http://cti030.citg.tudelft.nl

REFERENCES

[COBrITe] Construction Briefing Information Technology. http://www-staff.lboro.ac.uk/~cvmahh/cobrite-intro.htm

[SEED] Software Environment to support the Early phases in building Design. http://seed.edrc.cmu.edu

[Ozsariyildiz & Tolman 1998], Ozsariyildiz S, Tolman F.P., "IT Support for the Very Early Design Stages of Building and Civil Engineering Projects", Proceedings of CIB78 Conference, 1998

[Tolman & Ozsariyildiz 1998] Tolman. F.P. and Ozsariyildiz, S. *"A Product Modelling Approach to Competitive Tendering in the Building and Construction Industries"*, Proceedings of the 2nd International Conference on Concurrent Engineering in Construction, 1998

[Tolman et all 1999] Tolman. F.P., Ozsariyildiz S. and Schevers, H., "Information Modelling to Support Inception and Early Conceptual Design of Large-Scale Engineering Projects", International Journal of ICT, Volume 7, No. 2., 1999

[Ozsariyildiz & Tolman 1999] Ozsariyildiz, S. and Tolman, F.P., "First Experiences with an Inception Support Modeller for the Building and Construction Industry", Proceedings of CIB78 Conference., 1999

[Gielingh 1988] Gielingh W.F., General AEC Reference Model (GARM), TNO Building and Construction Research Report, 1988

[Rivard & Fenves 2000] Rivard H., Fenves S. J., A Representation for Conceptual Design and Buildings, Journal of computing in civil engineering, 2000

[Schevers 1999] Schevers, H.A.J, Conceptual Design Support for Technical Buildings, Technical University Delft, Master thesis., 1999

[Schevers & Tolman 2000] Schevers, H. and Tolman, F.P., "Supporting the Inception Stage of Building Projects with Instant Value versus Cost Evaluations" Proceedings of the CIB78 Conference. , 2000